



# THESE

En vue de l'obtention du

## DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*  
Discipline ou spécialité : *Informatique*

---

Présentée et soutenue par *Bashar KABBANI*  
Le *29/10/2015*

**Titre :**  
*Gestion unifiée et dynamique de la sécurité:  
Un cadre dirigé par les situations*

**Title:**  
*Dynamic Security Management:  
A Unified Framework oriented by Situations*

---

### JURY

**Président :** *M. David CHADWICK, Professeur, University of Kent, Kent, UK*

**Rapporteurs :** *Mme. Maryline LAURENT, Professeur TELECOM & Management SudParis, Paris*  
*M. Frédéric CUPPENS, Professeur TELECOM Bretagne, Rennes*

**Examineurs :** *M. Abdelmalek BENZEKRI, Professeur à l'Université Toulouse III, Toulouse*  
*M. Romain LABORDE, Maître de conférences à l'Université Toulouse III, Toulouse*  
*M. François BARRERE, Maître de conférences à l'Université Toulouse III, Toulouse*

---

**Ecole doctorale :** *ED MITT : Domaine STIC : Réseaux, Télécoms, Systèmes et Architecture*

**Unité de recherche :** *IRIT-UMR 5505*

**Directeur(s) de Thèse :** *M. Abdelmalek BENZEKRI*

**Co-Encadrants :** *M. François Barrère*  
*M. Romain Laborde*



# ACKNOWLEDGEMENTS

---

PhD thesis comes with continues collective and collaborative research work. It is a long productive process that needs patience, passion, organization and determination. Thus, being thankful is a virtue needs to be kept alive and therefore, being expressed out loud.

This thesis work was the fruit of four years of intensive efforts and beneficial collaborations. It has been accomplished on behalf of the Institute of Researches in Informatics at Toulouse (**IRIT**) and welcomed by the team of **SIERA** (Integration of Services and Networks Administration). The thesis would not see the light without being financed by the European project **PREDYKOT** (Policy REfined DYnamically and Kept On Track) of **ITEA2**. For all these reasons, it is my honor and great pleasure to acknowledge the people who have had a significant influence on me in order to succeed in publishing and achieving this scientific work.

First, I would like to sincerely thank Professor **Marline LAURENT** of TELECOM & Management SudParis and Professor **Frédéric CUPPENS** of TELECOM Bretagne, who accepted to be the examiners of this thesis. I am also grateful to Professor **David CHADWICK** for accepting to be the president of the jury during my thesis defense. I appreciate the time they spent on reviewing the dissertation, and in consequence, all their valuable and precious comments that helped improving the quality of the dissertation.

It is with no doubt that I am deeply appreciative to my supervisor and my two advisors and would like to express my gratitude to their various and generous contributions through the last four years:

Prof. **Abdelmalek BENZEKRI**, for his directions to enhance the quality of my work and the presentation of my research publications. I learned several managerial competences from him. I should also thank him for resolving my administrative issues and helping me focus on developing my thesis work.

Dr. **François BARRERE**, for being supportive and helpful at each aspect outside and inside the office whenever it is feasible. He was very helpful in discussing my research ideas and revising my English writing. He also played a key role in managing the financial aspects of my contract and business travels.

Dr. **Romain LABORDE**, for tremendously supporting me during my PhD journey. Romain's passion, guidance, and discipline have been indispensable to my growth as a scientist and as a person over the past four years. I am especially grateful to Romain for his devotion to his students' education and success.

I see myself blessed and lucky for having those three academicians in my life who equipped me with the necessary skills and capabilities that will help me go through my future adventures.

An extraordinary work environment that the SIERA team has provided to me through the last four years featured with its welcoming, support, humanity and consultancy. I would like to thank: Emmanuel Lavinal, Thierry Desprats, Michelle Sibilla and Philippe Vidal, for being supportive and leaving me to remember your encouraging smiled faces further after. I am also grateful to Julien Broisin, Daniel Marquié, and Michel Galindo and big thank to Bruno Roussel for his confidence on me giving the opportunity to help students in their professional career projects and for refreshing the atmosphere with his nice and funny comments whenever he visits me at the office.

I would not forget my colleagues and friends for their support and help by being generous with their ideas and suggestions to advance this work. I appreciate their availability to listen to my preparations and repetitions of the scientific disseminations I published. Particularly, I thank Ahmad Samer Wazan, Audrey Moui, Antoine Toueir, Hicham El Khoury, Ibrahim Yonis Omar, Ludi Akue, Marwan Cheaito, Maud Bouteillier, and Messaoud Aouadj.

Through the European Project PREDYKOT, I had the honor to collaborate with several persons on behalf of their partners in the industry and academia from 2011 to 2014. I would like to thank all those who participated in the success of the project and supported my research work. I would like to dedicate a special and a warm thank to Mr. Thierry Winter for managing the project. As a special acknowledgement, I would like to mention the names of those whom I had the opportunity to discuss and exchange ideas: Mikel Uriarte, Oscar López, Jordi Blasi, and José Luis Eguiguren (from Nextel), Harri Takala (Net Man Oy), Hadi Ghanbari (University of Jyväskylä), Nora Cuppens (Telecom Bretagne), Jarkko Holappa (Nixu Oyj).

I would not definitely forget the work experience I spent in Ireland. I would like to sincerely thank Dr. Edward Curry for accepting me in his DGSIT team in 2012/2013. I would like also to thank all the family of DERI; as they fascinatedly follow the cooperation and the harmony of a family. A special thanks to my friends and colleagues in Galway: Ahmad Tarek, Gofran Shukair, Souleiman Hasan, Wassim Derguech, Sonya Abbas, Fadi Maali, Islam Hassan and Lukasz Porwol. Thanks for the convivial moments that we shared with the HMI team, especially Philippe Palanque, Marco, Eric, Célia, and Camille.

Finally, I put every effort to acknowledge all the individuals who contributed to the success of my journey directly or indirectly with all hopes that I have not missed anyone who deserves to be mentioned.



# إهداء كشكر وتقدير

---

إنّ أتمن أنوار الشكر التعبير بحميدة ...

أهديكي لمن اشتاقت روحي ونفسي لها ... لي روح أمي نجلاء طيارة

لي أحكم إنسان قابلية ... لي روح جدي عبد القادر طيارة

*To my mother and grandfather souls ...*

أهديكي لي عائلتي التي وقفت بجانبتي وشمعتني وتحملت عناء النجاح ...

لي عائلتي - أبي فاخر قباني وأمي نخللة طيارة, أخي طارق, وأختي راما

لي جيتي خالوتي وخالي وأولادهم ... لي كل من في عائلتي قباني وطيارة

*To my parents and all Kabbani and Tabbara families ...*

لي أجباني إصوتي رفعتي وصحبي ... ولي كل من قلبه نبض نبضة تمننت لي الخير والنجاح

*To all my friends and beloved ones ...*

لي بلدي الحبيب ... سوريا

*To my beloved country ... Syria*

إيكم جميعا أجبكم وأهديكم نجاحي, شكري وتقديري

بشار قباني



# ABSTRACT

---

A Security Management System (SMS) connects security requirements to the technical application domain. On the one hand, an SMS must allow the security administrator/officer to translate the security requirements into security configurations that is known as the *enforcement process*. On the other hand, it must supply the administrator/officer with monitoring features (SIEM, IDS, log files, etc.) to verify that the environments' changes do not affect the compliance to the predefined security requirements known as the *monitoring process*.

Nowadays, guarantying security objectives requires a human intervention. Therefore, the SMS enforcement process is disconnected from the monitoring process. Thus, an SMS cannot dynamically guarantee that security requirements are still satisfied when environment behavior changings are observed. As part of the European project PREDYKOT, we have worked on closing the management loop by establishing a feedback on the dynamic behavior, captured from the environment, to impact the enforcement process. As a result, expressing and applying a dynamic security policy will be possible.

However, many policy expression and enforcement approaches exist currently. Each security management solution is dedicated to some specific issues related to authorization or to system/network management. Each solution provides a specific policy language, an architectural model and a management protocol. Nevertheless, closing the management loop implies managing both authorizations and system/network configurations in a unified framework.

Our contribution tackles the following three main issues:

- ◆ **Feedback:** The monitoring process captures the highly dynamics of the behavior through events. However, each event is not semantically associated with other events. We propose to get more semantics about behavior's changings thus introducing the concept of "*situation*" to be dealt with in security management applications. This concept aggregates events and links relevant security requirements, relevant behavior changes, and relevant policy rules. A new management agent, called the *situation manager*, has been added. The latter is responsible for the management process of the situations lifecycle (situation beginning and ending, etc.). We implement this software module using the complex event processing technology.
- ◆ **Policy Expression:** We propose to specify dynamic security policies oriented by situations. By doing so, the expression of the security policy rules becomes simpler to understand, easier to write and closer to the business and security needs. Hence, each relevant situation orients automatically the policy evaluation process towards a new dynamic decision that doesn't require updating the policy rules. We apply the attribute-based expression approach because of its ability to represent everything through attribute terms, which is a flexible way to express our dynamic policy rules.
- ◆ **Enforcement Architecture:** we propose a unified and adaptive architecture that supports situations-oriented policies enforcement. We choose to build an event-driven architecture. Exchanging management messages in terms of events allows our architecture to be independent from the management protocols. Thus, it covers in a unified way authorizations as well as configurations management approaches considering both provisioning and outsourcing policy control models. In addition, management agents are adaptable and can be upgraded dynamically with new management functionalities.

Our framework has been implemented and is compliant with the OASIS XACMLv3 standard. Finally, we evaluated our contributed according to four different scenarios to prove its generic nature.

## Keywords

*Policy-Based Management* ↔ *Security Policy* ↔ *Situation Management* ↔ *ABAC/XACML*  
*Security Management* ↔ *Adaptability* ↔ *Dynamic Authorization* ↔ *Configuration Management*



# RÉSUMÉ

---

Les systèmes de gestion de la sécurité (SGS) font le lien entre les exigences de sécurité et le domaine d'application technique. D'un côté, le SGS doit permettre à l'administrateur sécurité de traduire les exigences de sécurité en configurations de sécurité (appelé ici le processus de déploiement). De l'autre, il doit lui fournir des mécanismes de supervision (tels que des SIEM, IDS, fichiers de logs, etc.) afin de vérifier que l'état courant du système est toujours conforme aux exigences de sécurité (appelé ici processus de supervision).

Aujourd'hui, garantir que les exigences de sécurité sont respectées nécessite une intervention humaine. En effet, les processus de déploiement et de supervision ne sont pas reliés entre eux. Ainsi, les SGS ne peuvent garantir que les exigences de sécurité sont toujours respectées lorsque le comportement du système change. Dans le cadre du projet européen PREDYKOT, nous avons tenté de boucler la boucle de gestion en intégrant les informations sur le changement de comportement du système et en les injectant dans le processus de déploiement. Cela permet de faire appliquer des mesures de sécurité dynamiques en fonction des changements de comportement du système.

Toutefois, il existe diverses approches pour exprimer et mettre en œuvre des politiques de sécurité. Chaque solution de gestion est dédiée à des problématiques de gestion des autorisations ou à celles des configurations de sécurité. Chaque solution fournit son propre langage de politique, son propre modèle architectural et son propre protocole de gestion. Or, il est nécessaire de gérer à la fois les autorisations et les configurations de sécurité de manière unifiée.

Notre contribution porte principalement sur trois points :

- ◆ **Le retour d'information de supervision :** Le processus de supervision capture le comportement dynamique du système au travers d'événements. Chaque événement transporte peu de sens. Nous proposons de considérer non pas les événements individuellement mais de les agréger pour former des situations afin d'amener plus de sémantique sur l'état du système. Nous utilisons ce concept pour relier les exigences de sécurité, les changements dans le système et les politiques de sécurité à appliquer. Un nouvel agent, appelé gestionnaire de situations, est responsable de la gestion du cycle de vie des situations (début et fin de situation, etc.) Nous avons implanté cet agent grâce à la technologie de traitement des événements complexes.
- ◆ **Expression de la politique :** Nous proposons d'utiliser le concept de situation comme élément central pour exprimer des politiques de sécurité dynamiques. Les décisions de sécurité peuvent être alors automatiquement dirigées par les situations sans avoir besoin de changer la règle courante. Nous appliquons l'approche de contrôle d'accès à base d'attributs pour spécifier nos politiques. Cette approche orientée par les situations facilite l'écriture des règles de sécurité mais aussi leur compréhension. De plus, ces politiques étant moins techniques, elles sont plus proches des besoins métiers.
- ◆ **L'architecture de gestion :** Nous présentons une architecture de gestion orientée événement qui supporte la mise en œuvre de politiques de sécurité dirigées par les situations. Considérer les messages de gestion en terme d'événements, nous permet d'être indépendant de tout protocole de gestion. En conséquence, notre architecture couvre de manière unifiée les approches de gestion des autorisations comme des configurations (obligations) selon les modèles de contrôle de politiques en externalisation comme en approvisionnement. De plus, les agents de gestion sont adaptables et peuvent être dynamiquement améliorés avec de nouvelles fonctionnalités de gestion si besoin.

Notre cadriciel a été complètement implanté et est conforme au standard XACML v3.0 d'OASIS. Enfin, nous avons évalué la généricité de notre approche à travers quatre scénarii.

## Mots-clés

*Gestion à base de politique ↔ politique de sécurité ↔ gestion des situations ↔ surveillance des situations ↔ adaptabilité ↔ autorisation dynamique ↔ ABAC/XACML ↔ gestion des configurations ↔ gestion de la sécurité*



# TABLE OF CONTENTS

---

## PART I: CONTEXT & PROBLEMATIC

<b>GENERAL INTRODUCTION</b>	<b>1</b>
<b>I. RESEARCH QUESTIONS</b>	<b>3</b>
<b>II. THESIS CONTRIBUTION</b>	<b>4</b>
<b>III. DISSERTATION OUTLINE</b>	<b>5</b>
<b>IV. DISSERTATION STRUCTURE</b>	<b>6</b>
<b>V. PUBLICATIONS ARISING FROM THESIS WORK</b>	<b>6</b>

## PART II: STATE OF THE ART

### CHAPTER 2: SECURITY POLICY & MANAGEMENT POLICY EXPRESSIONS 9

---

<b>I. SECURITY EXPRESSIONS</b>	<b>11</b>
I.A. EXPRESSING POLICY USING MODELS	11
A.1. Historical view on models	11
A.2. RBAC and its family	13
A.3. Models influenced by RBAC	14
A.4. Usage Control model	15
A.5. Attribute-Based Access Control	16
I.B. EXPRESSING POLICY USING SECURITY LANGUAGES	17
B.1. XACML	18
B.2. PERMIS	21
B.3. WS-POLICY	22
<b>II. CONFIGURATION MANAGEMENT</b>	<b>24</b>
II.A. EXPRESSING MANAGEMENT POLICIES THROUGH LANGUAGES	25
A.1. CIM/PCIM-SPL DMTF management solution	26
A.2. IETF PIB	29
A.3. PDL	30
A.4. The Ponder Policy Specification Language	31
A.5. NETCONF configuration language	33

### CHAPTER 3: SECURITY & MANAGEMENT ARCHITECTURES 38

---

<b>I. GENERAL SECURITY MANAGEMENT ARCHITECTURE</b>	<b>39</b>
I.A. A LOOK AT EXISTING ARCHITECTURES	41
<b>II. ARCHITECTURES WITH MANAGEMENT OBJECTIVES</b>	<b>43</b>
II.A. SNMP-BASED ARCHITECTURE	43
II.B. POLICY-BASED MANAGEMENT	45
II.C. PBM COMMUNICATION PARADIGMS	47
II.D. DMTF WBEM ARCHITECTURE	50
II.E. PONDER ARCHITECTURE	52
II.F. CONCLUDING MANAGEMENT ARCHITECTURES	52
<b>III. ARCHITECTURES WITH SECURITY OBJECTIVES</b>	<b>54</b>
III.A. AAA ARCHITECTURE	54
III.B. OASIS XACML ARCHITECTURE	56
III.C. CONCLUDING SECURITY ARCHITECTURES	58

## **PART III: CONTRIBUTION**

<b>INTRODUCTION TO PART III</b>	<b>61</b>
---------------------------------	-----------

---

<b>CHAPTER 4: EXPRESSION OF A DYNAMIC SECURITY MANAGEMENT POLICY</b>	<b>65</b>
--	-----------

<b>I. EXPRESSING THE POLICY WITH FEEDBACK</b>	<b>70</b>
I.A. TERMINOLOGY	71
A.1. Situation	72
A.2. Context	75
A.3. Decision	76
I.B. METHODOLOGY: THE SCD POLICY ENGINEERING	77
B.1. SCD Policy Specification	77
B.2. SCD Policy Modeling	78
B.3. SCD Policy Expression	80
<b>II. THE FEEDBACK IMPLEMENTATION</b>	<b>82</b>
II.A. SITUATION MANAGEMENT	82
II.B. COMPLEX EVENT PROCESSING	84
II.C. ADAPTABLE SITUATION MANAGER	89
C.1. Situation Design & Dynamic Configuration	90
C.2. Situation Identification	94
C.3. Situation Preservation	98
<b>III. REPRESENTING THE SCD POLICY-EXPRESSION</b>	<b>99</b>
III.A. DYNAMIC XACML POLICY ORIENTED BY SITUATIONS	99

---

<b>CHAPTER 5: UNIFIED &amp; ADAPTIVE ARCHITECTURE</b>	<b>106</b>
---	------------

<b>I. INTERRELATIONS IMPLEMENTATION</b>	<b>108</b>
I.A. THE SM-PDP RELATIONSHIP	108
I.B. THE PDP-PEP RELATIONSHIP	109
B.1. Providing the Relevant Contexts	109
B.2. Asynchronized Security Management Decisions	109
B.3. Synchronized decisions requests/responses	110
I.C. MASTER PEP CONCEPTUAL ARCHITECTURE	110
I.D. THE PEP-SM RELATIONSHIP	112
<b>II. THE IMPLEMENTATION OF A UNIFIED ARCHITECTURE</b>	<b>113</b>
II.A. AN EVENT-DRIVEN ARCHITECTURE	113
A.1. Agents Implementations	116
A.2. Inter-Agents Communications	117
A.3. Managing Agents Exchanges	120
<b>III. SCD UNIFIED &amp; ADAPTIVE ARCHITECTURE</b>	<b>125</b>
III.A. THE SCD UNIFIED ARCHITECTURE	125
III.B. ADDING THE ADAPTABILITY FEATURE	127
B.1. Defining Adaptability Notions	127
B.2. Current XACML PEPs	128
B.3. A Dynamically Adaptable XACML v3 PEP	129
III.C. ADDING ADAPTABILITY TO AGENT IMPLEMENTATIONS	134



<b>CHAPTER 6: PROOF OF CONCEPT:THREE DIFFERENT SCENARIOS</b>	<b>139</b>
<b>I. 1<sup>ST</sup> SCENARIO: BREAK THE GLASS</b>	<b>140</b>
I.A. MODELING BTG SITUATIONS	140
I.B. REPRESENTING SCD POLICY	141
I.C. IDENTIFYING BTG SITUATIONS	142
I.D. SCD ARCHITECTURE	144
D.1. Event Sensors	144
D.2. BTG Manager	145
D.3. PI System	145
<b>II. 2<sup>ND</sup> SCENARIO: WORKFLOW MANAGEMENT</b>	<b>147</b>
II.A. MODELING WORKFLOW MANAGEMENT SITUATIONS	148
II.B. REPRESENTING SCD POLICY	148
II.C. IDENTIFYING WORKFLOW MANAGEMENT SITUATIONS	150
II.D. SCD ARCHITECTURE	151
<b>III. 3<sup>RD</sup> SCENARIO: JOB SUBMISSION</b>	<b>154</b>
III.A. MODELING THE JOB SUBMISSION SITUATIONS	155
III.B. REPRESENTING SCD POLICY	156
III.C. IDENTIFYING THE JOB SUBMISSION SITUATIONS	157
III.D. SCD ARCHITECTURE	158
D.1. Event Sensors	159
D.2. Job Submission Clients	159
D.3. Job Executer	159
<b>IV. FRAMEWORK EVALUATION</b>	<b>162</b>

## **PART IV: CONCLUSION AND FUTURE WORK**

<b>GENERAL CONCLUSION</b>	<b>172</b>
<b>I. RESEARCH QUESTIONS</b>	<b>173</b>
<b>II. FUTURE WORK</b>	<b>174</b>
II.A. SECURITY MANAGEMENT SPECIFICATION	174
II.B. SECURITY MANAGEMENT ARCHITECTURE	175

# Table of Figures

---

FIGURE 1.1: TODAY'S GENERIC ARCHITECTURE OF SECURITY MANAGEMENT SYSTEM.....	1
FIGURE 1.2: TODAY'S DETAILED VISION OF SECURITY MANAGEMENT .....	2
FIGURE 1.3: PREDYKOT INNOVATION AS GENERIC ARCHITECTURE.....	3
FIGURE 1.4: DISSERTATION PLAN.....	6
FIGURE 2.5: HISTORY OF SECURITY MODELS WITH SECURITY AND MANAGEMENT LANGUAGES .....	10
FIGURE 2.6: RBAC MODEL (ANSI INCITS 359, 2004).....	13
FIGURE 2.7: UCON MODEL AND ITS PROPERTIES (ZHANG, PARISI-PRESICCE, SANDHU, & PARK, 2005).....	16
FIGURE 2.8: XACML POLICY VERSION 3.0, OASIS STANDARD, 2013. ....	19
FIGURE 2.9: THE X.509 PMI RBAC POLICY AND ITS SUB-POLICIES .....	21
FIGURE 2.10: CIM METAMODEL SCHEMA .....	26
FIGURE 2.11: DETAILED SUBSET OF THE CIM POLICY SCHEMA v2.38.0.....	27
FIGURE 2.12: EXTRACT FROM THE CIM SYSTEM SCHEMA v2.42.0.....	28
FIGURE 2.13: NETCONF ARCHITECTURE (CHADHA & KANT, 2008, P. 168).....	34
FIGURE 3.14: GENERAL ARCHITECTURE OF THE SECURITY MANAGEMENT.....	39
FIGURE 3.15: STATE DIAGRAM FOR PASSIVE ARCHITECTURES.....	40
FIGURE 3.16: STATE DIAGRAM FOR ACTIVE ARCHITECTURES .....	41
FIGURE 3.17: SNMPV2 AGENT-MANAGER RELATIONSHIP.....	44
FIGURE 3.18: SNMP TECHNICAL ARCHITECTURE.....	44
FIGURE 3.19: POLICY-BASED MANAGEMENT ARCHITECTURE (RFC 2753). ....	45
FIGURE 3.20: RELATIONSHIPS OF PEP-PDP/LPEP, PDP-PEP/LPDP AND PDP/PEP.....	46
FIGURE 3.21: ONE-TO-MANY RELATIONSHIP USING THE PBM ARCHITECTURE. ....	47
FIGURE 3.22: THE PBM COMMUNICATION PARADIGMS OF OUTSOURCING AND PROVISIONING.....	48
FIGURE 3.23: OUTSOURCING AND PROVISIONING TECHNICAL ARCHITECTURES .....	50
FIGURE 3.24: WBEM ARCHITECTURE .....	51
FIGURE 3.25: INTEGRATED MANAGEMENT USING WBEM AND SNMP FOR CONFIGURATIONS PROVISIONING.....	51
FIGURE 3.26: PONDER ARCHITECTURE .....	52
FIGURE 3.27: GENERAL MANAGEMENT ARCHITECTURE .....	53
FIGURE 3.28: AAA GENERIC ARCHITECTURE (RFC 2903) .....	55
FIGURE 3.29: THREE AAA PDP/PEP INTERACTION MODELS: (A) AGENT, (B) PUSH, (C) PULL (RFC 2904).....	55
FIGURE 3.30: XACML PBM ARCHITECTURE.....	57
FIGURE 3.31: LITERATURE GENERAL VIEW OF ACCESS CONTROL ARCHITECTURES .....	58
FIGURE III.32: COMPARING PREDYKOT WITH THE CURRENT POLICY LIFE-CYCLE MANAGEMENT .....	62
FIGURE III.33: LINKING PREDYKOT ARCHITECTURE WITH NEXT TWO CHAPTERS .....	63
FIGURE 4.34: SITUATION AS A CORE CONCEPT.....	73
FIGURE 4.35: MODELING SITUATIONS USING STATE DIAGRAMS .....	79
FIGURE 4.36: MODELING THE SITUATIONS OF 1R1 BUILDING.....	79
FIGURE 4.37: GENERAL VIEW ON THE ARCHITECTURE OF THE AMIT SITUATION MANAGER. ....	83
FIGURE 4.38: GENERAL VIEW OF THE CEP ARCHITECTURE.....	86
FIGURE 4.39: SITUATIONS AND COMPLEX-EVENTS RELATIONSHIP IN EVENT STREAM PROCESSING (ESP). ....	87
FIGURE 4.40: OUR SM ARCHITECTURE.....	90
FIGURE 4.41: GENERAL DESIGN FOR A SITUATION. ....	90
FIGURE 4.42: INTERRELATIONSHIP BETWEEN SITUATIONS AND ENTITIES.....	91
FIGURE 4.43: CONTEXT MANAGER CONFIGURATION FILE .....	93
FIGURE 4.44: SM CONFIGURATION FILE .....	94
FIGURE 4.45: STREAM PROCESSING USING CEP RULES. ....	95
FIGURE 4.46: SITUATION IDENTIFICATION TECHNIQUE.....	95
FIGURE 4.47: SEQUENCE DIAGRAM OF THE INTERACTIONS BETWEEN CONTEXT AND SITUATION MANAGERS.....	97
FIGURE 4.48: SITUATION PRESERVATION PROCESS.....	98
FIGURE 4.49: OASIS XACML v3.0 A SECURITY MANAGEMENT POLICY ORIENTED BY SITUATIONS .....	100
FIGURE 4.50: CONCLUDING THE POLICY EXPRESSION CONTRIBUTION.....	104

<i>FIGURE 5.51: VIEW ON THE NECESSARY IMPLEMENTATIONS TO APPLY OUR SCD POLICY</i> .....	107
<i>FIGURE 5.52: PEP AGENT AS A SENSOR</i> .....	109
<i>FIGURE 5.53: PEP AGENT AS AN ACTUATOR</i> .....	110
<i>FIGURE 5.54: PEP AGENT AS A SMART NODE</i> .....	110
<i>FIGURE 5.55: CONCEPTUAL ARCHITECTURE OF A MASTER PEP</i> .....	111
<i>FIGURE 5.56: THE MISSING PUZZLE PIECE OF OUR MASTER PEP ARCHITECTURE</i> .....	112
<i>FIGURE 5.57: CONCEPTUAL EXAMPLE OF A UNIFIED ARCHITECTURE</i> .....	113
<i>FIGURE 5.58: TOPIC-BASED ENTERPRISE SERVICE BUS</i> .....	115
<i>FIGURE 5.59: IMPLEMENTATION OF THE THREE AGENT TYPES</i> .....	116
<i>FIGURE 5.60: THE MASTER PEP IMPLEMENTATION</i> .....	117
<i>FIGURE 5.61: THE OUTSOURCING SCD POLICY CONTROL MODEL IMPLEMENTATION</i> .....	118
<i>FIGURE 5.62: THE PROVISIONING SCD POLICY CONTROL MODEL IMPLEMENTATION</i> .....	118
<i>FIGURE 5.63: THE HYBRID SCD POLICY CONTROL MODEL IMPLEMENTATION</i> .....	119
<i>FIGURE 5.64: OUR AVAILABLE TOPICS FOR THE MESSAGES BUS</i> .....	122
<i>FIGURE 5.65: EXAMPLE ON THE MANAGEMENT OF EXCHANGES</i> .....	124
<i>FIGURE 5.66: THE FULL VIEW ON THE SCD UNIFIED ARCHITECTURE</i> .....	125
<i>FIGURE 5.67: MESSAGES EXCHANGED BETWEEN THE PEP AND THE PDP</i> .....	128
<i>FIGURE 5.68: RESPONSES SENT BY THE PDP</i> .....	129
<i>FIGURE 5.69: OUR ADAPTIVE PEP ARCHITECTURE</i> .....	132
<i>FIGURE 5.70: EXTRACT OF THE OBLIGATION EXPRESSION FOR THE “AUDITING” FUNCTIONALITY</i> .....	133
<i>FIGURE 5.71: PROCESS FOR UPGRADING THE PEP</i> .....	134
<i>FIGURE 5.72: UPGRADING AGENTS WITH ADAPTABILITY</i> .....	134
<i>FIGURE 5.73: STATE DIAGRAM REPRESENTING OUR DSMS ENFORCEMENT LOGIC</i> .....	136
<i>FIGURE 5.74: CONCLUSION VIEW ON OUR SCD UNIFIED ARCHITECTURE</i> .....	137
<i>FIGURE 6.75: STATE DIAGRAM OF THE BTG MANAGEMENT IN HEALTHCARE INFORMATION SYSTEMS</i> .....	141
<i>FIGURE 6.76: TECHNICAL IMPLEMENTATION OF THE BTG PROTOTYPE BASED ON OUR FRAMEWORK</i> .....	144
<i>FIGURE 6.77: WORKFLOW MANAGEMENT IN THE VO SCENARIO FOR THE VIVACE PROJECT</i> .....	148
<i>FIGURE 6.78: STATE DIAGRAM OF THE WORKFLOW MANAGEMENT</i> .....	148
<i>FIGURE 6.79: TECHNICAL IMPLEMENTATION OF WORKFLOW MANAGEMENT BASED ON OUR FRAMEWORK</i> .....	151
<i>FIGURE 6.80: STATE DIAGRAM OF THE JOB SUBMISSION IN GRID-COMPUTING ENVIRONMENT</i> .....	155
<i>FIGURE 6.81: TECHNICAL IMPLEMENTATION OF THE JOB SUBMISSION PROTOTYPE BASED ON OUR FRAMEWORK</i> .....	158
<i>FIGURE 6.82: RULE 1: TESTING 200 REQUESTS WITH PERMIT PERMISSION</i> .....	162
<i>FIGURE 6.83: RULE 5: TESTING 200 REQUESTS WITH DENY PERMISSION</i> .....	162
<i>FIGURE 6.84: BTG-END BUNDLE: READY TO APPLY</i> .....	163
<i>FIGURE 6.85: BTG-START BUNDLE: READY TO APPLY</i> .....	163
<i>FIGURE 6.86: LOGGING BUNDLE: READY TO APPLY</i> .....	164
<i>FIGURE 6.87: LOGGING BUNDLE: DOWNLOAD &amp; APPLY</i> .....	164
<i>FIGURE 6.88: BTG-END BUNDLE: DOWNLOAD &amp; APPLY</i> .....	164
<i>FIGURE 6.89: PERFORMANCE CONTROL POINTS TO MEASURE PROCESSING TIME FOR OBLIGATIONS GOING TO SM</i> ..	165
<i>FIGURE 6.90: SITUATION: BTG GRANTED</i> .....	165
<i>FIGURE 6.91: SITUATION: NORMAL</i> .....	166
<i>FIGURE 6.92: PERFORMANCE CONTROL POINTS TO MEASURE THE IDENTIFICATION TIME OF SITUATIONS</i> .....	166
<i>FIGURE 6.93: SITUATION: DOCTOR NEEDED</i> .....	167
<i>FIGURE 6.94: SEQUENCE DIAGRAM SHOWING MEASUREMENT OF CES DETECTION AND START OF NEW SITUATIONS</i> ..	167
<i>FIGURE 6.95: SCD POLICY LIFECYCLE</i> .....	171
<i>FIGURE 7.96: LINKING OUR CONTRIBUTION CHAPTERS WITH OUR SECURITY MANAGEMENT FRAMEWORK</i> .....	172
<i>FIGURE 7.97: OUR CONCEPTION FOR A MAPE AGENT</i> .....	172

# Table of Tables

---

TABLE 2.1: <i>XACML PSEUDO AUTHORIZATION POLICY FOR THE ACCESS CONTROL OF STATIC WEBPAGE.</i> .....	20
TABLE 2.2: <i>DISCLOSURE POLICY EXAMPLE.</i> .....	22
TABLE 2.3: <i>EXAMPLE ON WS-POLICY REPRESENTATION FOR SECURITY POLICY.</i> .....	23
TABLE 2.4: <i>AN EXAMPLE ON CIM-SPL POLICY.</i> .....	29
TABLE 2.5: <i>SAMPLE PDL POLICIES.</i> .....	30
TABLE 2.6: <i>EXAMPLE 1, CONNECTION FAILURE POLICY</i> .....	32
TABLE 2.7: <i>EXAMPLE 2, BANDWIDTH PERFORMANCE MANAGEMENT.</i> .....	32
TABLE 2.8: <i>EXAMPLE OF USING THE LOCK OPERATION IN NETCONF.</i> .....	34
TABLE 2.9: <i>COMPARISON BETWEEN SNMP AND NETCONF.</i> .....	35
TABLE 4.10: <i>SPECIFIED SECURITY REQUIREMENTS</i> .....	81
TABLE 6.11: <i>SUPPORTED HEALTHCARE EVENT TYPES THAT THE CONTEXT MANAGER CAN IDENTIFIES.</i> .....	142
TABLE 6.12: <i>SUPPORTED VO EVENT TYPES THAT THE CONTEXT MANAGER CAN IDENTIFIES.</i> .....	150
TABLE 6.13: <i>SUPPORTED GRID-COMPUTING EVENT TYPES THAT THE CONTEXT MANAGER CAN IDENTIFIES.</i> .....	157
TABLE 6.14: <i>PERFORMANCE TESTING OF THE AUTHORIZATIONS QUERIES (200 SEQUENCE)</i> .....	162
TABLE 6.16: <i>PERFORMANCE TESTING OF THE SITUATION IDENTIFICATION (50 SEQUENCE)</i> .....	166
TABLE 6.17: <i>CLARIFYING THE STANDARD DEVIATION AND ERROR OF THE MEASUREMENTS</i> .....	169



# Chapter 1

## General Introduction

*“A problem well stated is a problem half-solved.”*

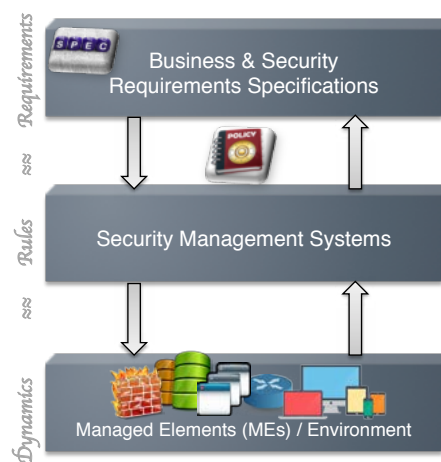
*Charles F. Kettering*

*Quote number 40299*

*Retrieved December 16, 2014, from Quotes.net*

Security of complex and highly dynamic information systems is a key property. Security management aims at ensuring and maintaining this property. Effective and efficient security management becomes a challenging task to achieve with business requirements (e.g., mobility, portability, and ubiquity).

Management of security to protect organizations' assets ensures that business requirements are met inside each element of the managed environment. To always keep all managed elements on the track of specified requirements, security management systems entail the creation of two sides: downward (security governance task) and upward (security supervision task) (Figure 1.1). The downward stream involves electing and refining security specified requirements into tangible enforcement rules. The upward stream involves monitoring the organization assets in order to measure the impact of the applied rules; defined during the downward stream.

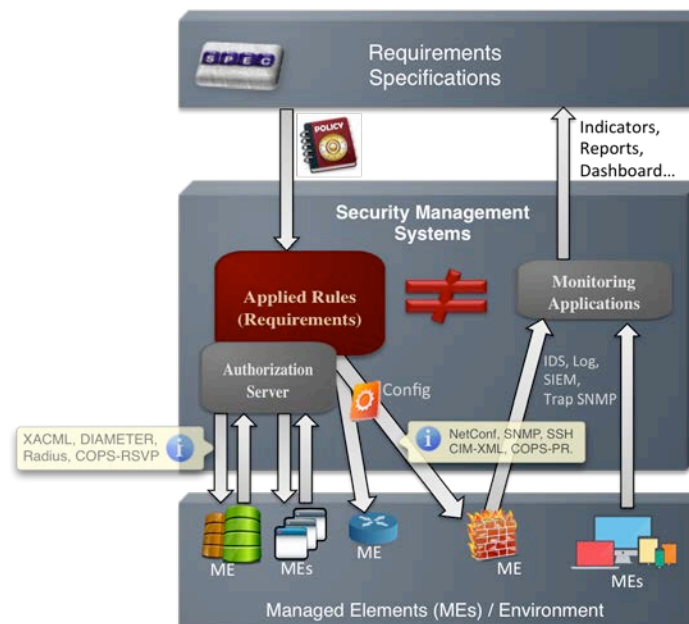


**Figure 1.1: Today's generic architecture of Security Management System**

The scope of this work concerns ensuring that applied rules are always meeting the specified requirements despite the dynamic behavioral changes of the managed elements. In Figure 1.2, the downward governance task aims at expressing the specified requirements into security rules applicable to the heterogeneous managed elements. To apply these rules, there are two communication possibilities between the security management system and the managed elements. One possibility is that the communications are predefined. This is the case of interactions between the managed elements and *authorization systems* (e.g., RADIUS, Cisco TACACS). Accepting this possibility requires the managed elements to contact the authorization system to inform about permissions and prohibitions. Rules at this possibility are centrally applied at the authorization system level. On a second possibility, the rules must be transformed into a configuration that is at once complied with requirements and understandable by the managed element. Configurations can be either independent (abstract) or dependent (technical) of the technology used at the managed

elements. Technical configurations are directly applied to the managed elements (e.g., Cisco, Linux, Unix, Windows). Abstract configurations are represented in form of abstract classes (e.g., using Managed Object Format (MOF) syntax from the Common Information Model (CIM)) or in form of high-level settings using configuration tools (e.g., Management Information Base). Abstract configurations can then be transformed into a more concrete technical form (e.g., SNMP configurations). The upward supervision task aims at monitoring the actual behavioral states of all managed elements. The monitoring process works on collecting interesting relevant information and reporting it to reflect the environment status. Interesting information could be related to different elements (e.g., servers, routers and different machines log files, intrusion detection systems (IDS) to inform about attacks, analyzed and structured information from SIEM (security information and event management), and other centralized management information). Interesting information is either used directly in their produced form as technical events or is treated (grouped, analyzed, filtered, etc.) to create readable indicators for humans. These report indicators participate in identifying whether the requirements are still met or not.

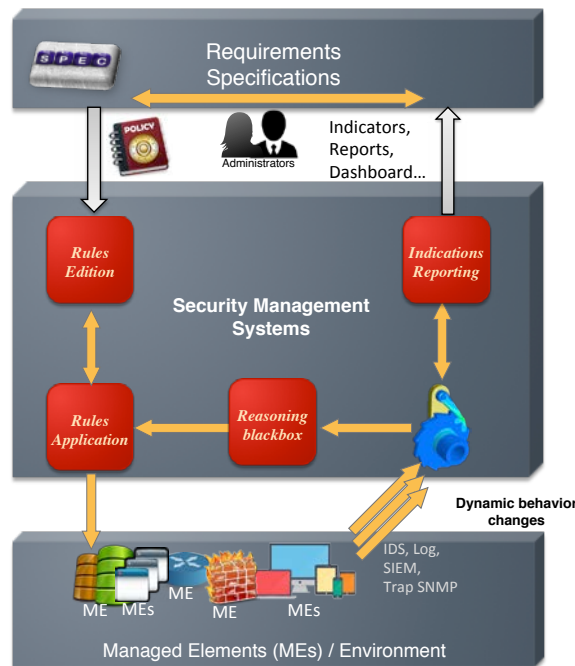
Today's presented vision of managing security is not effective and efficient enough (Figure 1.2). The disconnection between the reported information during the supervision task and the applied rules during the governance task prevents security management systems from guarantying that requirements are *always* respected by the managed elements (i.e., continuously at any given time). Although the administrator(s) can identify the affected requirements, the question raised is "why the disconnection is still a problem"? Highly dynamic environments are full of modifications and changes in terms of the managed elements' behavior. The problem source returns to the difficulty and the complexity produced from having the disconnection between the triples: rules, requirements and behavioral changes. With such disconnection, it is hard to recognize due to certain behavior changes, which rule(s) is/are still valid and still meet(s) the associated requirement(s).



**Figure 1.2: Today's detailed vision of Security Management**

We were part of the European project "PRE DYKOT". The project aims at ensuring that the Policy *always* REfined DYnamically and Kept On the Track; all along the duration of its application. Theoretically speaking, the project proposed to connect the downward and upward streams to overcome the illustrated disconnection between governance and supervision. Thus, it creates conceptually two security loops: a large and a small one. The large loop did exist before. It represents a manual loop involving the human to participate in refining the rules based on the provided reports concerning the behavior. The innovative small loop is an automatic loop which application creates an adaptive solution. The small loop helps detecting the behavior and (re) acting with security configuration to adapt the behavior back to meet the requirements. Thanks to the reasoning box

(Figure 1.3), the small loop is created. So, the innovation the project brought is to propose capturing the supervised feedback coming from monitoring the environment behavior changes. The feedback refers to semantically representing technical raw events or even analyzed SIEM events (e.g., alert of an attack).



**Figure 1.3: PREDYKOT innovation as generic architecture**

The thesis contribution is related to the small-automated security loop of PREDYKOT. To realize our contribution, we have chosen to research in the trend of policy-based management (PBM). PBM is a management approach effective to use for complex systems with highly dynamic environments. PBM aims at self-adapting the policy due to dynamic changes within the managed elements without interrupting the management system functionalities.

Nevertheless, the management community initially founded the policy-based management. Hence, the approach aims at performing management tasks (e.g., quality of service, service level agreement) by ensuring requirements such as adaptability, reliability and scalability. Afterwards, the security community integrated the necessary requirements for ensuring the security property (i.e., the confidentiality, integrity and availability (CIA)). In consequence, studies were conducted based on two objectives: management using configurations (with obligation-oriented policies) and security (with authorization-oriented policies). Performing automatized security management requires both objectives, and this therefore requires a unification solution.

## I. Research Questions

From observing the state of the art, studies have shown that both worlds of security and management consider two axes in their solutions: a language for the policy expression, and architecture for the policy enforcement. Therefore, the challenge that the PREDYKOT generic architecture requires is finding a generic implementation. From comparing both figures 1.2 and 1.3, one can distinguish three interesting points to question about the feedback.

- RQ 1: First point to question comes at the connection between the behavior feedback and the specified requirements. Studies hold variety of policy expression languages and models. The question is therefore how to unify all existing policy expressions in one solution that represents the behavior feedback and the heterogeneous requirements?



- RQ 2: The reasoning box is the innovation point that represents the observing of behavior changes and sends them as a meaningful feedback. Therefore, the question is how to express such a meaningful feedback and how to manage the lifecycle of the policy? Does the feedback representation help bringing closer the expression of the high-level requirements with the low-level policy rules? Why reacting on raw or simple events is not enough anymore as a feedback?
- RQ 3: Current infrastructures are heterogeneous and incompatible with each other. Therefore, the enforcement of policy decisions once taken should be adaptably enforced to overcome such heterogeneity. So, the third point is what enforcement architecture could apply both authorization and configuration decisions into any technical environment?

## II. Thesis Contribution

We imagine that any security management system could be visualized as a middle-box between the requirements and the managed environment (cf. Figure 1.3). Considering PREDYKOT results, the small loop renders the security management system automatic and adaptive to behavior changes. This means, the security management system is aware of the behavior changes in form of a meaningful feedback and is able to adapt to this feedback by enforcing the correct policy (i.e., managing the policy lifecycle). Our contribution takes place in finding an implementation to the small loop.

With many heterogeneous expression solutions for both security and management objectives, it is very difficult to make all the solutions converge into one (e.g., a meta-model solution (Barker et al., 2009)). However, it is possible to use an abstract and generic approach that can fulfill both security and management expression objectives. Attribute-based is a generic expression approach first advertised as an access control model called ABAC (attribute-access control model) (McCollum, et al., 1990). Any object is characterized with attributes and this is the feature of the approach. Every expression is based on the term “attribute”. For instance, the role of a subject is doctor. The role is an attribute. Technically speaking, XACML proposes both a standard architecture and a language featured with its support for the attribute-based access control approach. Therefore, we used XACML version 3 in our security management solution to express the policy. (Kabbani et al., 2013)

Nevertheless, adequately using the attribute-based approach alone is too abstract. Such an abstraction produces difficulties when technically controlling and analyzing the low-level behavior. For instance, describing all the environmental events with different types of attributes and then considering them into the policy is a hard process. Therefore, the behavior changes should be represented using a specific-attribute that could semantically express what is happening inside the environment. Moreover, this meaningful representation as the feedback needs to be expressed inside the security management policy. Our contribution then is introducing the definition of such specific-attribute to be the term “situation”. Situations represent conditions and circumstances in which one object of the managed system (e.g., firewall) finds itself. Situations start and end, and therefore have a lifetime. Expressively, situations are feedbacks on what is happening inside the managed environment (e.g., during the night, any access is prohibited). They are also possible to characterize objects as attributes inside the policy (e.g., the firewall is under attack). Technically speaking, situations need to be identified. Situation identification is the reasoning on what starts, ends and keeps a certain situation ongoing. To monitor behavior changes, we use the Complex Event Processing technology to identify meaningful situations. We build the “Situation Manager”, with reference to (Adi et al., 2004), as the agent responsible for identifying the relevant and accurate situations and providing them to the policy decision process. It manages the entire identification process and keeps informing about whether a situation is and is not valid for authorization decisions. (Kabbani et al., 2014)

As an outcome of using the *situation* term, we overcome the limitation of current security policies by avoiding frequent rules updates. Moreover, the expression of situations provides the policy with a more abstract view of the behavior by analyzing the monitored events instead of basing the policy on events. However, we still need to enforce security and management decisions in order to adapt the managed environment upon the situations identification. We are proposing an adaptable enforcement architecture that could apply both authorization and configuration decisions without restrictions on the used technology inside the managed environment. Our contribution is based on an event-based architecture that follows the Bus technology (Publisher/Subscriber messages exchange) (Laborde et al., 2014).

### III. Dissertation Outline

The dissertation is divided into four parts. The first part contains Chapter 1 considering the general introduction. Part II discusses our research background and presents the state of the art. It contains two chapters dealing with policy expression and deployment architecture. The third part presents our contribution within the PREDYKOT project. Our vision is detailed in depth through three chapters dedicated to expression, architecture and concrete scenarios as a proof of concept. The fourth part concludes the thesis work with the final chapter and proposes future directions.

In Chapter 2, we present the current approaches to express the policies in both management and security communities. We found that the security community uses access control models and languages to express the security policy. As for the management community, the management policy is expressed using obligation languages.

In Chapter 3, we present the management and security architectures state of the art. We insist on their characterization by providing sound examples.

In Chapter 4, we present the contribution to the policy expression. Integrating the notion of situations into the security management policies as an attribute. We define the situation notion and explain the identification phase in profound. Finally, we present the functionalities of our situation manager.

In Chapter 5, we present the expected architecture in order to comply with the presented policy expression. We present the “adaptability” as a key concept to this chapter and explain why and how an architecture is considered adaptive. Afterwards, we present our adaptive unified architecture that combines XACML, Event-Based and Situation Manager architectures.

In Chapter 6, the objective of this chapter is a proof of concept. It promotes the generic and unity of our contribution through presenting three different scenarios. The first scenario is broadened inside the healthcare environments. A second scenario discusses the importance of having security management of configuration that is flexible and dynamic. Previous scenarios have security-oriented objectives. Therefore, and towards proving the generic concept, we investigate our security management contribution on a third scenario with management-oriented objectives. It is about ensuring dynamic authorization inside energy management systems. With these three different scenarios, we expect no limitation for applying our solution to any environment. We finalize this chapter by evaluating our framework through testing the SCD architecture performance.

In Chapter 7, we conclude the presented work highlighting our answers to the listed research issues. Moreover, as a result of observing our contribution, we propose an outlined methodology with structural procedures that we followed in each presented scenario. Then, we mention future directions for which this thesis could be considered as the foundation stone on unified and dynamic authorization based environments.

## IV. Dissertation Structure

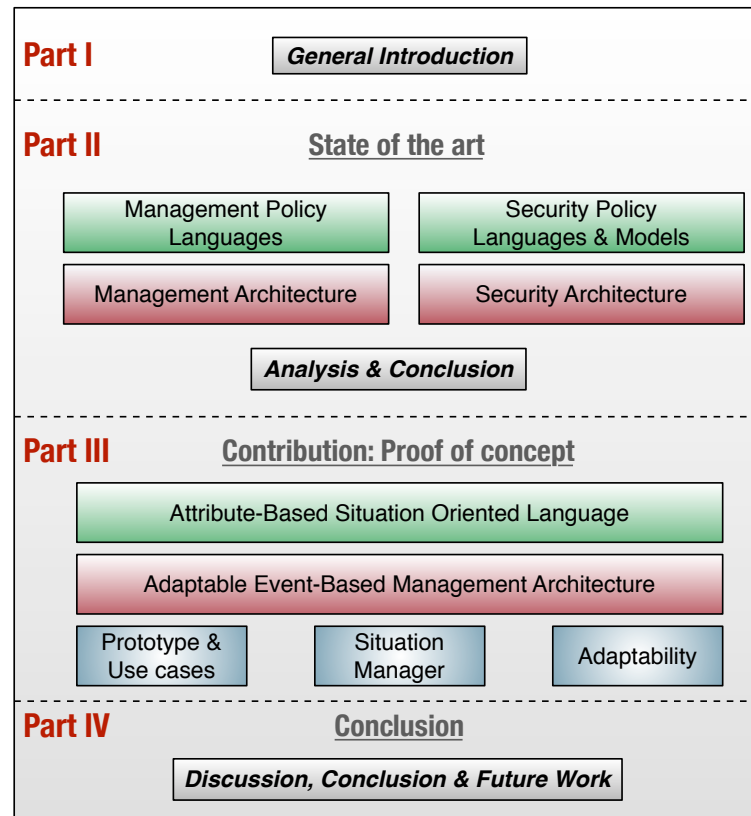


Figure 1.4: Dissertation Plan

## V. Publications arising from thesis work

### International conferences and workshops articles

- [Romain Laborde](#), [Bashar Kabbani](#), [François Barrère](#), [Abdelmalek Benzekri](#). *An adaptive XACMLv3 policy enforcement point (regular paper)*. Dans : *IEEE International Workshop on Adaptive Systems for Communication Networks (WASCOM 2014)*, Västerås, Sweden, 21/07/2014-25/07/2014, (Eds.), [IEEE Computer Society](#), 2014.
- [Bashar Kabbani](#), [Romain Laborde](#), [François Barrère](#), [Abdelmalek Benzekri](#). *Specification and Enforcement of Dynamic Authorization Policies oriented by Situations (regular paper)*. Dans : *IFIP International Conference on New Technologies, Mobility and Security (NTMS 2014)*, Dubai, UAE, 30/03/2014-02/04/2014, (Eds.), [IEEE Communications Society](#), p. 1-6, avril 2014.
- [Abdelmalek Benzekri](#), [François Barrère](#), [Romain Laborde](#), [Bashar Kabbani](#). *Managing Dynamic Authorization (regular paper)*. Dans : *Workshop on Code, Cryptography and Communication Systems, Meknes, Morocco, 07/11/2013-08/11/2013* (conférencier invité), [Ecole Supérieure de Technologie](#), (en ligne), novembre 2013 (keynote speaker).

### National conferences and workshops articles

- [Bashar Kabbani](#), [Romain Laborde](#), [François Barrère](#), [Abdelmalek Benzekri](#). *Managing Break-The-Glass using Situation-oriented authorizations (regular paper)*. Dans : *Conférence sur la Sécurité des Architectures Réseaux et Systèmes d'Information (SAR-SSI 2014)*, Saint-Germain-Au-Mont-d'Or (Lyon), France, 13/05/2014-16/05/2014, (Eds.), [HAL-INRIA](#), (en ligne), mai 2014.

# PART II



## STATE OF THE ART

*The second part presents the state of the art. We present in two chapters the studies related to the thesis work in term of policy expression and implementation. Our target studies are ones related to both security and management trends. Chapter 1 lists the background on different policy languages and security models. Chapter 2 lists the background on different implementation architectures for the policy application and enforcement.*

---

# Chapter II

## Security Policy & Management Policy Expressions

---

*The following chapter deals with the different methodologies to express a security management policy. These methodologies have always been a matter of heterogeneity. The chapter is divided into two sections. The first one focuses on the expressions of the security policy in term of access control models and authorization languages. The second section focuses on the expressions of management policy through two types of languages: configuration and obligation.*

---

### Summary

#### **I. Security Expressions**

- *Expressing Policy using Models*
  - 1) Historical view on models
  - 2) RBAC - Role Based Access Control and its family
  - 3) Models influenced by RBAC
  - 4) UCON - Usage Control model
  - 5) ABAC - Attribute-Based Access Control
- *Expressing Policy using Security Languages*
  - 1) XACML
  - 2) PERMIS
  - 3) WS-POLICY

#### **II. Configuration Management**

- *Expressing Management Policy through Languages*
  - 1) CIM/PCIM-SPL (DMTF management solutions: Common Information Model/Policy Core Information Model - Simplified Policy Language)
  - 2) IETF PIB - Internet Engineering Task Force, Policy Information Base
  - 3) PDL - Policy Description Language
  - 4) The Ponder Policy Specification Language
  - 5) NETCONF: network configuration language

# ***Chapter 2: Security Policy & Management Policy Expressions***

---

*“To those who have exhausted politics,  
nothing remains but abstract thought.”  
Honoré De Balzac*

*Columbia World of Quotations  
Retrieved October 08, 2014, from Dictionary.com*

---

## **Introduction:**

Security management has to be considered as a binominal concept that intersects with the world of management and the world of security. Therefore, the challenge is finding the best policy expression for the management of security. Policy expressions are subject to different objectives. Studies are carried on expressing policies with orientation towards either the management or the security objectives. On the one hand, the observation that one can retain from the security literature is their dependence on models in order to express the policies. The policy languages are either representing specific-language or model-based authorization solutions. On the other hand, the management community presented studies on different languages that match their objectives. Modeling in management was not interested in expressing the policy, but in representing the managed information for configurations.

In the 60s, studies on policy emphasized security objectives first with constraints related to single time-sharing of mainframe computers. Solution expanded later to contain individual enterprise constraints. Afterwards, policy evolved to target environments with multi-layers networks. In the late 60s, the United States department of defense was not satisfied with the management of rights limitations (read, write, etc.). Bell labs were recruited to present the first security policy that is considered the earliest access control model.

In Figure 2.5, an illustration of the access control models' history evolution is presented. The draw starts the story from 1973 to 2005. By 1972, the security consideration was no longer a monopoly by the US military. Studies were published to secure the files management into the existing operating systems at that era. At this point, the security objectives were derived to fulfill both military and organizations requirements. The big achievement in the period of 1977 to 1989 was basically on three research axes. The first research axis was interested in defining models for the security of operating systems as a reasoning logic through preserving the ownership of resources exclusively to the users (discretionary) or to decouple the relationship (non-discretionary). Military requirements have encouraged the second axis to focus on improving the security of transmitted and processed data (i.e., namely in term of integrity and confidentiality). The scope of the third axis was even broader as it required studies to focus on both security and business objectives. Therefore, this research axis was pushed toward finding security commercial solutions adapted to business organizations. With the expanding of Internet, security objectives to manage users accounts and roles were crucial. Therefore, RBAC<sup>1</sup> was a revolutionary solution that presented a template to express security policies with consideration to a hierarchy of their roles and work domains. At the same time, the management

---

<sup>1</sup> Role-Based Access Control

community was interested in presenting languages that help expressing solutions of controlling and administrating the networks complexity (e.g., Clark's policy for network administration, Border Gateway Protocol and Inter-Domain Routing Protocol).

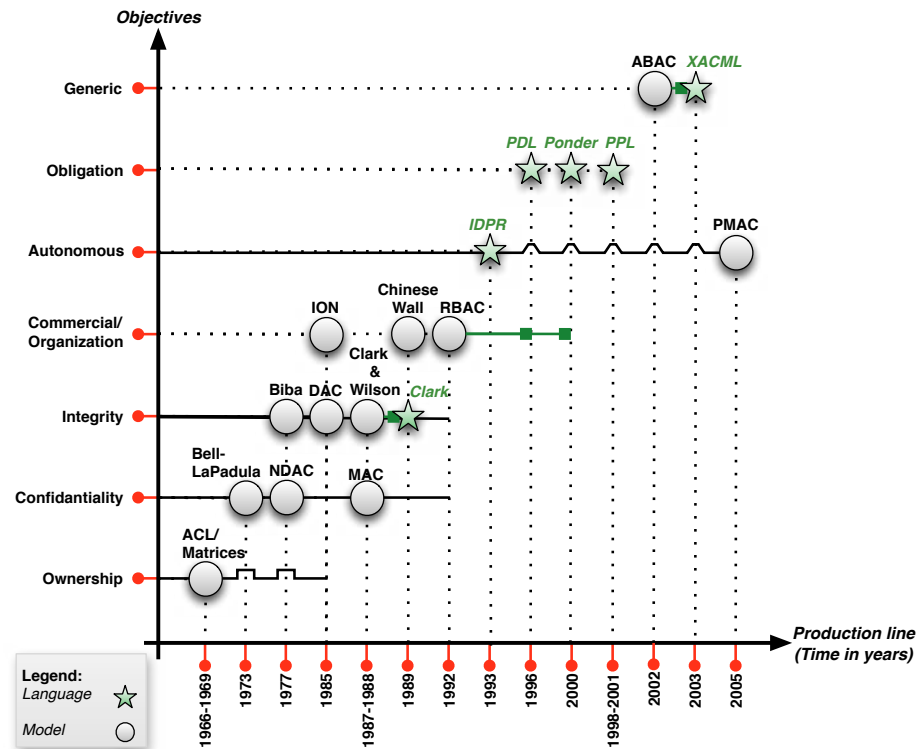


Figure 2.5: History of security models with security and management languages

Significantly, the production line of the *security models* and the implementation of *languages* for both worlds have rapidly increased since 1993 (i.e., as will be explained at the following state of the art). The more security objectives evolve, the more security models and templates production increase (privacy, availability, etc.). Likewise, the more management objectives evolve, the more the language production increases (autonomy, scalability, etc.).

The 1999's innovation was the presentation of the policy framework to the management community that abstractly considers some security objectives. As a consequence, Ponder born as a management language, applies RBAC security templates. It could be then considered as the first trial towards a security management solution.

Chapter 2 aims at more detailing the historical progress illustrated in Figure 2.5. It goes even beyond this history panorama by stressing the heterogeneous security and management expressing solutions in the literature. Chapter 2 is structured to present first security expression solutions, represented by the different existing models and languages. Second, the Chapter presents management expression solutions represented by the different existing languages.



# I. Security Expressions

*“A model is a graphical, mathematical, or verbal representation of a concept, phenomenon, relationship, structure, system, etc.”*

*Business Dictionary*

## I.A. Expressing Policy using Models

As security policy is a structured statement that represents the agreed security requirements. It is based on the security policy that secured systems must enforce. Therefore, software engineering best practices recommend models to represent the conceptual understanding of the security requirements. The advantages of having a model are to facilitate the understanding by highlighting necessary features and eliminating the unnecessary ones. This helps the decision maker in designing and simulating scenarios.

In 1970, Hoffman first presented formally the access control model in the domain of computer systems privacy. Until today, models have expressed the security policy in order to be easily represented in a policy language that can be enforced and deployed in the managed environment. These models have become a sort of templates for security policy writers. It is enough to determine what security features and quality should be respected to meet the requirements in order to choose a security policy model that matches this determination. Actually, choosing models becomes almost axiomatic in the business process plan. The requirements should just describe on what the system will be essentially based (e.g., roles, tasks, teams, organizations, etc.) in order to choose the right model.

The drawback appears when the requirements, the environment, or the business plan changes. In using traditional access control models, which do not consider dynamicity and flexibility, the solution was to simply change the model and pick one that should meet the new changes. In 1993, the first appearance of models that could adapt to environment changes was introduced (Abrams et al., 1993; Thomas & Sandhu, 1993). Since then, access control model makers keep presenting new models to follow the technological advances in the environment and business. The secret behind adapting to the environment changes is keeping the conceptual templates heterogeneous as much as possible to cover all kinds of security requirements—that is, by representing flexible notions (views, contexts, domains, organizations, etc.) and pushing the dynamic to be at the level of the policy rules. Changing the rules to pick the one that adapts to the environment changes and managing them by the flexible notions gives the dynamic to access control models. Moreover, the linking between the conceptual level (e.g., roles) and the low-level rules (e.g., firewall rules) imposes more engineering efforts, such as role engineering.

### I.A.1. Historical view on models

Two of the most famous traditional access control models are discretionary access control (DAC) and mandatory access control (MAC).

In 1985, the Department of Defense (DoD) presented the DAC model at the National Computer Security Center. The DAC model is implemented essentially in the Unix/Linux operating systems to control access to files. The DAC model recognizes that the subject, object, and action define the state of the system. The access control matrix lets subjects define permissions for actions on their own objects (Harrison, Ruzzo, & Ullman, 1976). The operations on a matrix include entering or deleting an action in the matrix and creating or deleting an object or a subject. The DAC model supports the permission delegation for information's owners in order to decide who can do what on



their information (Bell, 2005; Bus & Harrison, 2006). Such flexibility may allow wrong decisions regarding access control restrictions or maliciously set insecure or inappropriate permissions. The DAC model does not enforce information flow control at the level of process in which the model allows information leakage for users that are not allowed to read it (i.e., harm the integrity of the information). In terms of dynamicity, for example, the operation *chown* permits granting and revoking permissions, bypassing the system administrator control (Hu, Ferraiolo, & Kuhn, 2006; Park & Sandhu, 2002). To the contrary, non-discretionary access control (NDAC) identifies the contexts in which authority is vested in some users, but there are explicit controls on delegation and propagation of authority (Damianou, Bandara, & Sloman, 2002). NDAC policies have rules that are not created by the owners and can be changed only through administrative actions.

Most organizations own the property of information rather than the end user. The MAC model follows the non-discretionary concept by means of administrative predefinition of control information to overcome the confidentiality of information (i.e., system entities have no control) (Cullough, 1990). The MAC model introduces the security clearance concept that is calculated based on the sensitivity of information contained in each resource. The subjects are first controlled by this manner to satisfy the military requirements of the United States DoD. The sensitivity level of objects and subjects determines their classification or category. For instance, lattice-based access control is a MAC model that defines the following security levels: Top-Secret, Secret, Confidential, and Unclassified. MAC-based systems are difficult to use and administer because of the restrictions and limitations imposed by the operating system.

Sophisticated security models that formalize security policies for commercial purposes include the Clark–Wilson model (Clark & Wilson, 1987), the Bell–LaPadula model (Bell, 2005), and the Chinese-wall model (Brewer & Nash, 1989). The Clark–Wilson model aims at ensuring the integrity of the accounting system and improving its robustness. The Clark–Wilson model recommends well-formed transactions and enforces the separation of duty. Well-formed transactions preserve and ensure the integrity of data. Separation of duty partitions the tasks and associated privileges by means of creating cooperation between multi-users to complete sensitive tasks. This reduces the possibility of fraud or damaging errors. The Chinese-wall policy is applied essentially to financial information systems. It helps individual consultants by preventing conflicting information flows (known with *conflict of interests*). *“The Chinese Wall policy combines commercial discretion with legally enforceable mandatory controls. It is required in the operation of many financial services organizations and is, therefore, perhaps as significant to the financial world as Bell-LaPadula's policies are to the military.”* (Brewer & Nash, 1989).

In September 2006, the National Institute of Standards and Technology (NIST) presented a list of concepts that every access control would need (Hu, 2006, NISTIR7316, p. 3). Objects are defined as entities that store and receive information and imply access to this information (e.g., records, fields in a database record, blocks, pages, files, directories, network nodes, electrical switches, relays, etc.). Subjects are seen as active entities that represent a person, process, or device. They cause information to flow among objects or change the system state (Gasser, 1988; Gallagher, 1988). Operations are active processes launched by a subject. Permission (privilege) is the right granted to a subject to perform some authorized actions on the ecosystem (Ferraiolo, Kuhn, & Chandramouli, 2003).

An access control list (ACL) is a list planted inside an object that contains identification of all the subjects that are allowed to access this object. Entries of the list are formed in pairs (subject, set of rights). Each ACL represents a column in the access control matrix, a table with rows representing subjects' identification and columns representing objects. The cells of this table are access rights that link a subject to an object. Separation of duty (SOD) is a principle that implies that under no

circumstances should a user have enough permission to abuse the system. Safety is a concept that measures that the leakage of permissions to an unauthorized principal will not be produced by the access control configuration. A safe configuration exists when no permission leaks to an unauthorized or unintended principal (Hu, 2006, NISTIR7316, p. 4).

The limitation that can be considered from analyzing the above-mentioned traditional models is their *rigidity*—strong attachment to specific properties, categories, or metrics. For instance, the Bell-LaPadula model is concerned about confidentiality and presents a confidential policy to prevent unauthorized access. The MAC model is based on the regulations mandated by a central authority (Boutaba & Aib, 2007). These models were only sufficient to classical computing systems such as databases and file systems.

## I.A.2. RBAC and its family

A new era in access control modeling started when security researches began to drive an access control based on classifications, or categories. Role-based access control (RBAC) has been defined to classify the access rights based on roles inside organizations. For instance, a doctor can access her patient information. A trainee doctor, on the other hand, could not modify the patient information. This methodology of identifying a subject by its role has reduced the management headache, as it groups the characterizations of individuals. But what if enterprises needed to make the authorizations based on individuals themselves or their tasks, or another property?

RBAC imposes a difficult process of understanding the organization's structure from the role point of view only. The output of this process links access rights with the defined roles. Therefore, RBAC imposes role engineering to facilitate this process, which is the process of developing an RBAC structure for an organization. Large firms often discover the need for hundreds of roles, which they must structure into an efficient hierarchy to manage permissions for the various roles of the organization's many IT systems. Some organizations also include non-IT permissions in roles. (Coyne, Weil, & Kuhn, 2011)

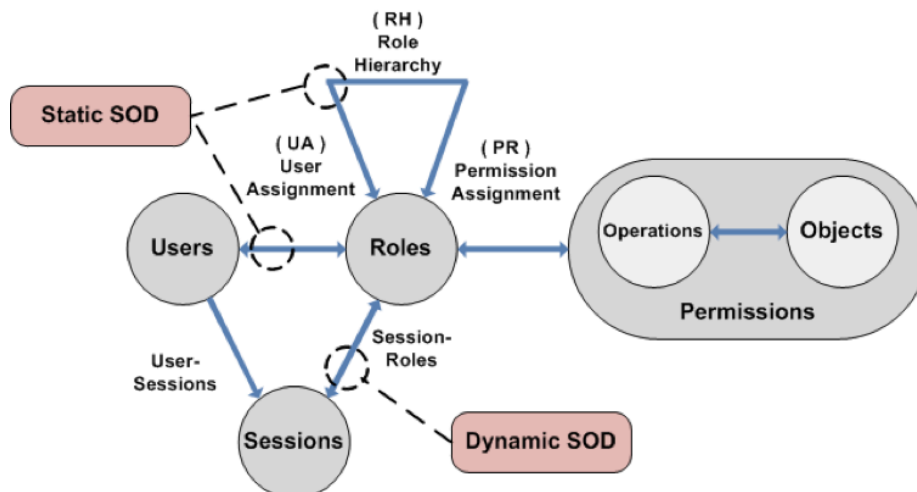


Figure 2.6: RBAC Model (ANSI INCITS 359, 2004)

Defining roles by an engineered approach is called a top-down method, as the bottom-up approach is rather interested in data mining techniques by extracting patterns from the databases and users' permission lists (i.e., role mining). RBAC is based on the principle that the most crucial information to access a resource is the role that a user plays in the system (Chadwick, Otenko, & Ball, 2003). RBAC defines a role as “a job function within the context of an organization with some associated semantics regarding the authority and the responsibility conferred on the user assigned to

the role” (Gavrila & Barkley, 1998; Gouglidis & Mavridis, 2010). That means that a user is associated to a role and that permissions are associated to that role. Currently, Gartner’s recent market analysis considers RBAC as one of the most commercially adopted means of access control (O’Connor & Loomis, 2011).

After introducing RBAC to organizations, researchers were going through creating a family for this model and making this family grow bigger with the requirements’ evolution. The traditional access control models no longer satisfied organizations. They claimed the need to align requirements with technological advances, but keeping the role structuring valid. Therefore, researchers worked to produce models that are dynamic in the way that roles are assigned differently when the context changes, but the role assignment strategy and the context definition depend on the model’s creation objectives. Here are some examples of the RBAC family to explain this type of dynamicity:

- ❧ **Privacy-Aware.** P-RBAC extends RBAC to support privacy issues. Its objective is to keep limited to the public privacy policies, privacy statements, and privacy acts like HIPPA in healthcare applications (Ni, Bertino, & Lob, 2008).
- ❧ **Dynamic Coalition in Distributed Environments.** dRBAC is applied in cloud computing to handle policy administration, role mapping and role delegation, such as in multi-tenant policy management issues (Freudenthal, Pesin, Port, Keenan, & Karamcheti, 2002).
- ❧ **BTG-RBAC.** RBAC is a rigid model with only two access control decisions: Allow or Deny. In healthcare environments, Break The Glass (BTG) scenarios require flexible policies that allow users to bypass the access control by breaking or overriding the policy in a controlled and reasonable manner. The BTG-RBAC model aims at providing an BTG option, which allows users (originally unauthorized) to break the glass and have access (Ferreira et al., 2009).
- ❧ **Temporal-RBAC (TRBAC).** This RBAC extension tackles a dynamic aspect with regard to the role availability as it differs between users at certain time-slots, known as temporal dependencies among roles. TBAC supports enabling and disabling roles periodically and temporarily (Joshi et al., 2001).
- ❧ There are many other models in this family. For instance, here are some extensions of the RBAC model: **I-RBAC**: Isolation Enabled Role-Based Access Control (Gunti, Sun, & Niamat, 2011); **RBAC<sup>+</sup>**: Dynamic Access Control for RBAC-administered web-based Databases (Bouchahda, Thanh, Bouhoula, & Labbene, 2010); **PuRBAC**: Purpose-Aware Role-Based Access Control (Masoumzadeh & Joshi, 2008); **CRiBAC**: Community-based Role interaction-based Access Control model (Jung & Joshi, 2012); **ROBAC**: Role And Organization Based Access Control Model (Zhang, Zhang, & Sandhu, 2006).

### I.A.3. Models influenced by RBAC

AC models, other than RBAC and its family, drive the authorization system to deliver authorization decisions on giving or taking rights based on a class or a category, other than roles. Access control models based on the concept of team, task, and token have been introduced. Task-based access control (TBAC) is a task-oriented and traditional subject-object access control that is proposed in boundaries of agent-based distributed computing and workflow management. During the completion of tasks, the model involves authorizations in accordance with some application logic (Thomas & Sandhu, 1997). However, work on managing the workflow has been treated in the research without the need to create yet another model (Laborde, Kamel, Barrere, & Benzekri, 2007). Team-based access control (TMAC) is applied in collaborative environments where activities are accomplished through organized teams. The abstract notion of a team encapsulates a collection of users in special roles, assigning them specific tasks or goals. This model follows the concept of activating a context to assign permissions. Finally, token-based access control is implemented in the cloud-computing environment. With interest in efficiently managing jobs, the model defines six types

of access levels and an enforcement strategy that reduces the number of jobs required (Khaled, Husain, Khan, Hamlen, & Thuraisingham, 2010).

AC models are based on context, organizations, states, etc. In smart spaces, context-based access control (CBAC) is a model aiming at managing authorizations based on the trust level of the participants, calculated based on their context: location, current date, device type, etc. (Smirnov, Kashevnik, Shilov, & Teslya, 2013). A competing model to CBAC is the situation-based access control (SitBAC) model. It is a conceptual model that represents mainly healthcare organization scenarios by defining phenomena describing the access to patient data (e.g., patient's data-access scenarios). The SitBAC formally represents access situations as a concept ontology that includes patient, data requestor, medical record (MD), task, and response with their attributes and relationships (Beimel & Peleg, 2010, 2011). SitBAC is introduced to cover essentially one-security management requirements: confidentiality in healthcare. To control authorizations on mobile devices, the stateful access control model (SACM) aims at creating new rules or deleting old rules in order to dynamically distribute them to mobile devices. The authorization decisions then are delivered based on the stored state in the model (dos Santos et al., 2013). Another model is Nephele, which constructs hyper-groups, a special form of network domains, and keeps the authorization and authentication of users controlled locally at each of these domains (Margaritis, Hatzieleftheriou, & Anastasiadis, 2013). The risk-adaptable access control (RAdAC) model, or contextual risk-based access control, presupposes that there is a risk factor and a necessity associated with each access. Therefore, for each access or operation, the system based on this model should measure and assess the risk and the necessity. The calculation is made of the risk of accessing a certain resource against the denied access cost (Diep et al., 2007; McGraw, 2009).

The final dynamic access control we present here is the organization-based access control (ORBAC or OrBAC). OrBAC considers the *organization* as the most important term to base the authorization decisions on. *Organization* is considered as an abstract layer that lets role, activity, and view concepts abstracting the subject, action, and object concepts. It aims at defining a security policy independently from the implementation. Based on the behavior of the organizations, the authorization decisions may change. The behavior is represented by contexts in which they are defined as logical rules, and reasoning on them activates these contexts. Like all access control models, OrBAC faces a matter of conflicts and defines *strategies* to overcome the rule incoherence (Kalam et al., 2003).

It is essential for this work to mention that every dynamic access control model mentioned above is dynamic because it works on activating, changing, and modifying the rules of the security policy. Therefore, dynamic access control models are subject to conflicts, as they involve activating contexts, tasks, teams, etc., thus violating old rules. Hence, the refinement must be performed frequently and repeatedly during the runtime.

#### **I.A.4. Usage Control model**

A new model recently introduced aims at separating the usage control from being managed by the access control, which will be a reason to create more models. UCON was officially introduced to define the usage of rights upon digital objects (Park & Sandhu, 2002). UCON includes trust management, to assign authorization to unidentified subjects in an open environment like Internet, and digital rights management (DRM), assures the control at the client-side (Feltus, 2008). UCON supports the usual access control functions: authorization, obligation, and conditions. Authorization evaluates subjects before the usage to decide if they deserve the access or not. Obligation verifies, both before the usage and during it, that the subjects respect predefined conditions. Conditions are based on the environment or the system. The new proposition by this model is the ongoing decision process that

goes before and during the usage and the manipulation of objects or subjects by updating them during or after the usage. However, UCON shows weaknesses in detecting and handling policy conflicts, hierarchies, and temporal constraints along with revocation of user rights. The model also adds administration complexity in term of users' identities, especially in highly heterogeneous and dynamic environments (Danwei, Xiuli, & Xunyi, 2009; Lazouski, Martinelli, & Mori, 2010; Zhang, Parisi-Presicce, Sandhu, & Park, 2005).

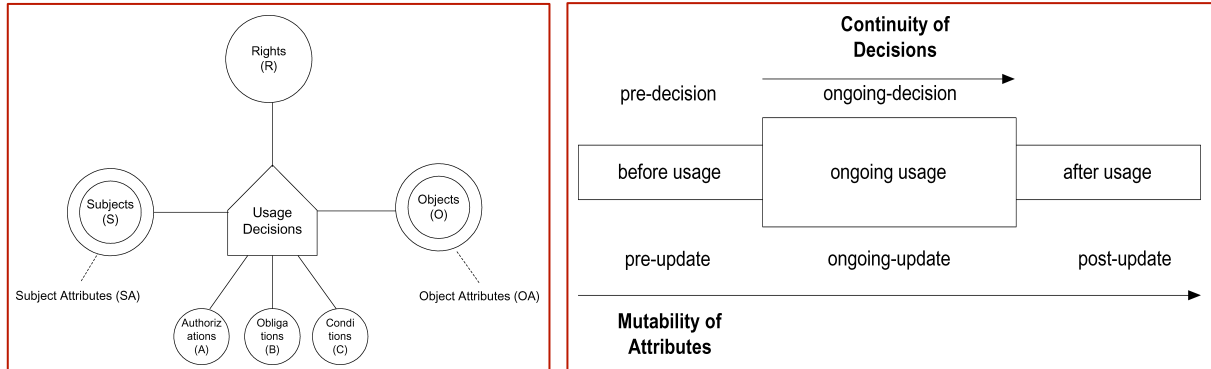


Figure 2.7: UCON model and its properties (Zhang, Parisi-Presicce, Sandhu, & Park, 2005).

## I.A.5. Attribute-Based Access Control

Knowledge is a very important aspect during the security policy expression, especially for a brand new system or network. This knowledge is only gathered by well defining and analyzing the future designed system or network in order to provide well-defined security requirements. Then, the effort on expressing the security policy comes based on these requirements. However, the more the knowledge about the system or the network in terms of expected behaviors (e.g., context of the system or the network) and the environment capabilities, the more accurate and comprehensive the expression with regard to the security and expression requirements.

From analyzing the access control models methodology, the need for a generic approach to express the security policy is obvious in order to fulfill all the expression requirements and to provide a uniform approach for an organization instead of 700 models (Barker, 2009).

Attribute-based access control (ABAC) is not an access control model, as many researchers describe it (Kuhn et al., 2010). Rather, ABAC is an approach to expressing the security policy based on *attributes*, a generic term that can handle and meet any expression requirements.

The presentation of the ABAC model is a first step toward a generic solution for access control systems (Yang & Jia, 2014; Hur & Noh, 2011; Lang, Foster & Siebenlist, 2009; Li, Gao & Wang, 2013). The model's generality comes from the notion of attribute, which aggregates all properties and categories specified in older models. This abstract view makes the expression of policies easier, as the most important step become the definition of the attributes that an AC system should handle. ABAC represents a set of attributes that characterizes a subject; based on an evaluation process, the subject will be authorized or not to access controlled resources. For instance, the roles are attributes characterizing subjects and are considered in the evaluation process. Therefore, RBAC is a special case of ABAC, where the RBAC model considers all evaluation processes to be based on roles.

Based on the assessment of attribute information (i.e., time of the day, persons logged on to the network) and information about a user (e.g., role and location), the authorization system can give almost instantaneous decisions about whether a subject is appropriately authorized to perform an action on an object. The ABAC advantage is the leveraging of attributes about subjects, objects, and their environment conditions. In conclusion, ABAC avoids role engineering. However, in case



attributes were not defined consistently, the security policy becomes more dynamic than would be preferable for audit and attestation. In consequence, ABAC requires specifying a larger number of rules, making the refinement more difficult (Colantonio, 2012). Nevertheless, in case the attributes were defined consistently and comprehensively, the security policy should be stable enough with minimum possible dynamicity and therefore with conflict-free rules.

### **I.A.5.I Attribute Terminology**

The power of attributes is that they do not need to be defined or categorized. However, it helps the usage and the linking in the security policy and for this manuscript to be consistent. Therefore, the terminologies to be defined are mainly, with no limitation, subjects, resources, actions, environment, organization, domain, etc.

The NIST institute adopted the concise definitions of the multiple terms (attributes) that are frequently used throughout the Federal Government and the Identity and Access Management (IAM). An object, or a resource, represents the protected entity against unauthorized access. The subject is any entity of the system or the network that requests to perform an action, an operation, or a process that needs access authorization for the targeted object. Most of the subjects are considered human. However, if a computer or another non-human subject performs the operation, it must be performed on behalf of an authorized person or organization.

Attributes are characteristics of a subject such as name, birthday, address, medical record, and role in the organization. These attributes individually or combined comprise the subjects' unique identity. The environment attributes are conditions that represent dynamic factors, such as context: time, location, threat level, and temperature.

Policy rules govern the allowed behavior within the environment of an organization. The rules are written based on attributes like subjects, objects, actions, and the environment's conditions. Managed objects' attributes help in describing their identities, also known as resource attributes.

We<sup>2</sup> believe that ABAC is not a model. ABAC is an expression method that approaches to facilitate and generalize the expression of security and management policies based on an abstract terminology, which is "*attribute*".

## **I.B. Expressing Policy using Security Languages**

In access control, heterogeneity introduces more heterogeneity. By analyzing the literature around AC models diversity, the policy is usually expressed considering three main aspects: access control models, security requirements, and the domain of application according to its technical environment (e.g., cloud computing, healthcare, smart grids, etc.). In the literature, these three elements permit the representation of the expressed policy to be oriented either by authorizations with interests prioritizing security objectives – namely CIA (Confidentiality, Integrity, and Availability) based on access control models.

Authorization languages are introduced mainly for security management purposes to tackle security issues. These languages prioritize CIA requirements. W3C P3P (Platform for Privacy Preferences Project) and W3C APPEL (An Adaptable and Programmable Policy Environment and Language) use a P3P Preference Exchange Language. APPEL language is call control language based

---

<sup>2</sup> The Institute of Research in Informatics at Toulouse (IRIT), the team Service IntEgration and netwoRk Administration (SIERA)

on XML grammar for interchanging of structured data in distributed systems through three levels: the core, the policy and the domain (Turner, Reiff-marganiec, Blair, Campbell, & Wang, 2007). APPEL is dedicated to sensor networks and homecare applications. Languages represent security and privacy in pervasive computing environments; for example, Rei is a policy language based upon the deontic logic (Tonti et al., 2003).

There are policy language representations associated only with specific models. For instance, the IBM Enterprise Privacy Authorization Language (EPAL) is a specific policy language for purpose-based access control, an extension of RBAC. ASL (Authorization Specification Language) represents an RBAC model using first-order logic (Jajodia, Samarati, & Subrahmanian, 1997). Another RBAC-specific policy language is Tower (Hitchens & Varadharajan, 2001).

Nevertheless, there are three languages among the most common and important ones to study when handling issues related to access control. First, the standardized XACML<sup>3</sup> by OASIS is an attribute-based language. At the second place, PERMIS<sup>4</sup> a language, an architecture and special infrastructure oriented to manage the privileges and the RBAC model polishes its syntax through the usage of *Roles*. Its strength is coming from the ability to be integrated virtually into any application and any authentication scheme (e.g., Shibboleth (Internet2), Kerberos, username/passwords, Grid proxy certificates and Public Key Infrastructure (PKI)). At the third place, the WS-POLICY is a dedicated language for web-services that is also polished by the RBAC model. (Han & Lei, 2012)

### **I.B.1. XACML**

The OASISXACML<sup>5</sup> standard describes both an XML policy language and an access control decision based on an XML request/response language. To meet the general access control requirements, the policy language has standard extension points for defining new functions, data types, combining logic, etc. Concerning the request/response language, it customizes a query to inquire whether a given access request should be allowed or not, and then to interpret the result. However, the response is an answer about whether the request should be allowed choosing one of four values: Permit, Deny, Indeterminate or Not Applicable. The Indeterminate value occurs because of an error or some missing required value, so a decision cannot be made. The Not Applicable is also a special case when the request can't be answered by this service.

The XACML policy language is used to describe general access control requirements in terms of constraints on attributes. Specifically, attributes could be any characteristics of any category such as the subject, the resource, the action, or the environment in which the access request is made. Attributes have an identifier, which is a Uniform Resource Name (URN), and a data type also identified by a URN. Considering attributes makes the language very flexible. Moreover, XACML language is natively extensible.

In Figure 2.8, the XACML security policy is composed of several XML tags. The top XML tag in the hierarchy is the policy set. XACML security policy could contain several policy sets. Policies are grouped in a policy set. However, any policy includes the following:

- A **target** element which is a first filter for searching the applicable policy
- A set of **obligation expressions** that are instantiated when a matching request is processed. PEPs must enforce obligations.

---

<sup>3</sup> XACML eXtensible Access Control Markup Language

<sup>4</sup> PERMIS Privilege and Role Management Infrastructure Standards

- A set of *advice expressions* that are instantiated when a matching request is processed. An advice expression is similar in its form to an obligation. However, PEPs may or may not enforce an advice expression. Advice expressions are new features introduced by version 3 of XACML.
- A set of *rules*, which are expressions to determine if a request is denied or permitted. A *rule* contains a *target* and may include *obligation* and *advice expressions* specific to this rule. Obligation and advice inside rules are new to version 3 of XACML.

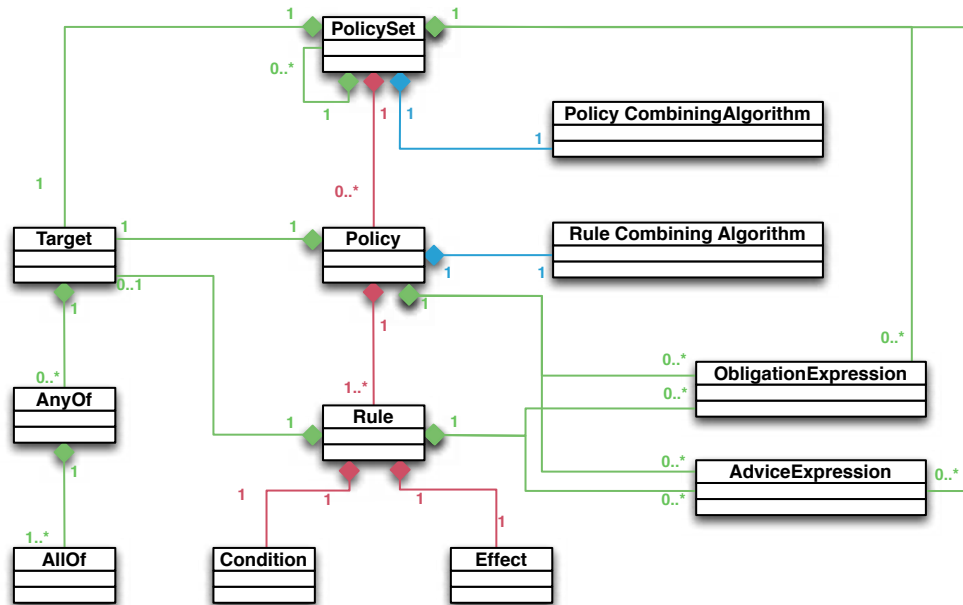


Figure 2.8: XACML policy version 3.0, OASIS standard, 2013.

An example that can demonstrate the expressiveness power and features of XACML is through version 3. The example takes the filtering of web page resources by controlling the access to them according to several levels of the request. A static web page contains immobile contents (e.g., static images and texts). There should be a mechanism to control the appearance of images for only authorized users and not all of them. Therefore, there exist a specific permission for each image. As a result, the web page that is viewed by a user (i.e., at a specific time) permits dynamically only the authorized images. Rules are based on assigned users' attributes such as their roles. Based on this sample scenario extracted from WSO2 samples, we can create the following high-level policy:

#### Web Page Authorization Policy

The main page web *index* is allowed for access (namely *view*) by three groups of users: “*publicUsers*“, “*internalUsers*“, and “*adminUsers*“. The public users can *view* two types of images (*view.gif* and *help.gif*). The internal users are allowed to view in addition to the two previous types the *copy.gif*, *move.gif* and *cancel.gif*. The admin can see all previous images and the *add.gif*, *edit.gif*, and *delete.gif*. Roman, François, and Abdelmalek are users of the mentioned groups respectively. Outside this policy, default decision is deny.

XACML is an attribute-based language that manipulates attributes coming from different sources (e.g., the request itself) in order to deliver a response that contains decision(s). The XACML representation for the described authorization policy is listed in the Table 2.1.

Based on this policy, the request will consider the *view* requests coming from everywhere inside an organization and deliver responses to these requests. Assuming Bob as nonuser (or from



unmentioned group), the response of the policy would be ‘deny’. However, any of Romain, François, or Abdelmalek would be allowed to view the web page with the images *view.gif* and *help.gif*.

The XACML authorization language is a flexible, powerful and stable standard language. OASIS standardized XACML to express policies based on attributes. This feature bestowed the language greater flexibility to express any security objectives, including the CIA requirements. At many state-of-the-art studies that we came by, there is regrettably an absence in presenting the XACML strengths. The trustworthiness in the sources of attributes that a party provides is an important issue to consider as the decision itself is made based on the evaluation of these attributes. The issuer of the attributes information is well covered by the XACML syntax and gives the decision trust a level in terms of the quality of security.

The management requirements and objectives do not elect the XACML language for the task of configuration management. The XACML was not designed initially as an obligation oriented language. However, the flexibility of the language does not stop at the point of expression. The extensibility, which the language and its framework offer, permits the supportability of new features and requirements to be included. Therefore, many studies ignore this feature and evaluate the language according to its current and standard drawn visage.

**Table 2.1: XACML pseudo authorization policy for the access control of static webpage<sup>6</sup>.**

---

<Policy>
<Target><AnyOf><AllOf>             <AttributeValue>index.jsp</AttributeValue>             <AttributeValue>view</AttributeValue>           </AllOf></AnyOf></Target>           <Rule Effect="Permit" RuleId="Rule_for_all_groups">             <Target><AnyOf><AllOf>               <AttributeValue>publicUsers</AttributeValue>               <AttributeValue>internalUsers</AttributeValue>               <AttributeValue>adminUsers</AttributeValue>             </AllOf></AnyOf></Target>             <Condition><Apply FunctionId="string-at-least-one-member-of">               <AttributeValue>view.gif</AttributeValue>               <AttributeValue>help.gif</AttributeValue>             </Apply></Condition></Rule>           <Rule Effect="Permit" RuleId="Rule_for_all_internal_user_group">             <Target><AnyOf><AllOf>               <AttributeValue>internalUsers</AttributeValue>               <AttributeValue>adminUsers</AttributeValue>             </AllOf></AnyOf></Target>             <Condition><Apply FunctionId="string-at-least-one-member-of">               <AttributeValue>copy.gif</AttributeValue>               <AttributeValue>move.gif</AttributeValue>               <AttributeValue>cancel.gif</AttributeValue>             </Apply></Condition></Rule>           <Rule Effect="Permit" RuleId="Rule_for_all_admin_user_group">             <Target><AnyOf><AllOf>               <AttributeValue>adminUsers</AttributeValue>             </AllOf></AnyOf></Target>             <Condition><Apply FunctionId="string-at-least-one-member-of">               <AttributeValue>add.gif</AttributeValue>               <AttributeValue>edit.gif</AttributeValue>               <AttributeValue>delete.gif</AttributeValue>             </Apply></Condition></Rule>           <Rule Effect="Deny" RuleId="Rule_deny_all"/>
</Policy>

---

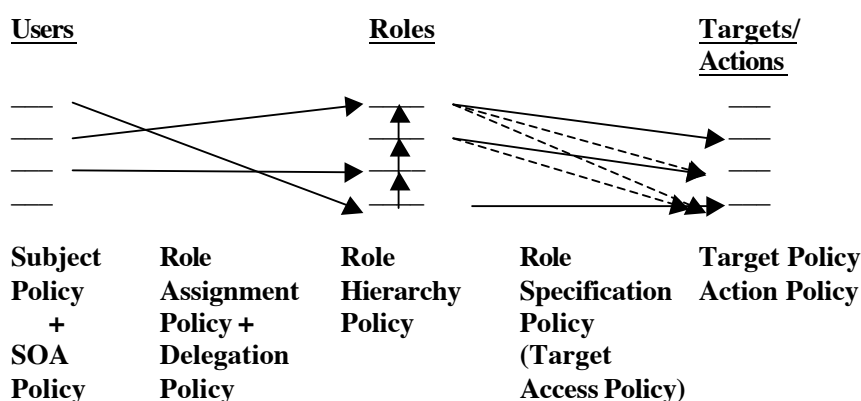
<sup>6</sup> This is not an executable policy; it is an extraction from the original written in a simpler manner as XACML example <https://raw.githubusercontent.com/wso2/commons/master/balana/modules/balana-samples/web-page-image-filtering/resources/web-image-filter-policy.xml>.

## I.B.2. PERMIS

One of the soundest authorization solutions that is harmonized with the attribute certificates X.509 and standardized by NIST, PrivilEdge and Role Management Infrastructure Standards (PERMIS) allows the management of privileges based on roles by providing a complete framework (system and infrastructure). PERMIS ensures the implementation of the access control based on the RBAC model.

The framework provides a privilege management infrastructure (PMI) with a secure cryptography approach through public key encryption technologies (PKI).

PERMIS X.509 PMI RBAC Policy (Chadwick & Otenko, 2002) is comprised of several sub-policies (see Figure 2.9). The PMI Policy domain unifies all the sub-policies domains. A unique object identifier (OID) is assigned to each policy to globally identify it. A policy language needs to specify multiple heterogeneous policies combinations and their matching enforcing mechanisms (Zhou & Meinel, 2007). However, one drawback of PERMIS is that it does not offer such a possibility.



**Figure 2.9: The X.509 PMI RBAC Policy and its Sub-Policies (Chadwick & Otenko, 2002, p. 5, figure 1)**

Obviously, PERMIS is a good utility to manage the authorization of organizations using the notion of domain. An example of a PERMIS policy can be made to better understand the expression using its RBAC language. The Disclosure Policy example, extracted from (Lopez, Canovas, Gomez-Skarmeta, Otenko, & Chadwick, 2005), is applied “*when two or more domains are involved in a trust relationship, where users from one domain can request access to resources in the other domains, it is necessary to define which user’s attributes could be revealed to those domains. If the domain requesting those attributes is a highly trusted domain, due to a previously established very restrictive security agreement, the home domain could reveal all the user’s attributes. Otherwise, if the relationship established between the domains is not so trusted, the home domain could decide to conceal some of them*”.

The Disclosure Policy aims at controlling access (namely the disclosure) to all available attributes that exist in the managed domain (called CCS<sup>7</sup>). Therefore, the policy defines the following elements: Subjects, Roles, SOA, Targets, Actions, and Target Access. Subjects are the only allowed set of external domains that can request access to the internal domain user’s attributes. Roles describing what role each external domain has to play. SOA is the Authorization Authority (AA) that manages the Authorization Certificates issued by the home domain (the PERMIS authorization infrastructure through subject-attribute pairs) for each external domain (CCS). Targets describe what

<sup>7</sup> Credential Conversion Service integrates external authorization schemes (non SAML-based) into authorization scenarios which make use of SAML as the main language for assertions. (Cánovas et al., 2004)

set of users will have their attributes disclosed at the access request moment. There is only one action defined here (i.e., disclose) and it gets one parameter (i.e., the attribute to disclose). The Target Access describes the link (between a particular set of users and the assigned attributes) that can be disclosed to which domains, and under which conditions (see Table 2.2).

In conclusion, the simple Disclosure Policy expresses that only CCSs (Subjects) with LongTerm-CC attribute value, will obtain access to the attribute *studentRole* type, with the value ERASMUS, assigned to the targeted users (Students). Upon the receiving of the attribute query, a request is generated in form of a message for each user attribute. The message contains the specified target domain (i.e., including subject and role), the requested attribute (i.e., including type and value) and the required action.

PERMIS implies that any organization needs (mostly obligated) to create and design roles (possibly need to provide a hierarchy) then assigns them to domains and subjects – it is known as the role engineering phase – in order to be able to use the language.

**Table 2.2: Disclosure Policy example (Lopez, Canovas, Gomez-Skarmeta, Otenko, & Chadwick, 2005).**

---

```

<X.509_PMI_RBAC_Policy OID="2.6.2004.24.1.2005">
  <SubjectPolicy><SubjectDomainSpec ID="SD_International_CCSs">
    <Include LDAPDN="cn=CCS, o=SAMLDomain, c=C "/>
  </SubjectDomainSpec></SubjectPolicy>
  <RoleHierarchyPolicy>
    <RoleSpec Type="permisRole" OID="1.2.826.0.1.3344810.">
      <SupRole Value="ShortTerm-CCS"></SupRole>
      <SupRole Value="LongTerm-CCS">
        <SubRole Value="ShortTerm-CCS">
          </SubRole>
        </SupRole>
      </RoleSpec></RoleHierarchyPolicy>
    <SOAPolicy><SOASpec ID="PERMISDomain_UAM"
      LDAPDN="cn=UAM, o=PERMISDomain, c=C "/></SOAPolicy>
    <RoleAssignmentPolicy><RoleAssignment>
      <SubjectDomain ID="SD_International_CCSs"/>
      <RoleList>
        <Role Type="permisRole" Value="LongTerm-CCS"/>
      </RoleList> <Delegate Depth="0"/>
      <SOA ID="PERMISDomain_UAM"/>
      <Validity/>
    </RoleAssignment></RoleAssignmentPolicy>
    <TargetPolicy><TargetDomainSpec ID="All-Users">
      <Include LDAPDN="o=PERMISDomain, c=C"/>
    </TargetDomainSpec><TargetDomainSpec ID="Students">
      <Include LDAPDN="ou=Students, o=PERMISDomain, c=C"/>
    </TargetDomainSpec><TargetDomainSpec ID="Professors">
      <Include LDAPDN="ou=Professors, o=PERMISDomain, c=C"/>
    </TargetDomainSpec>
  </TargetPolicy>
  <ActionPolicy><Action Name="disclose" Args="role value"/></ActionPolicy>
  <TargetAccessPolicy><TargetAccess>
    <RoleList><Role Type="permisRole" Value="LongTerm-CCS"/></RoleList>
    <TargetList><Target Actions="disclose">
      <TargetDomain ID="Students"/>
    </TargetList>
    <IF><AND><Substrings><Arg Name="role" Type="String"/>
      <Constant Type="String" Value="studentRole"/></Substrings>
      <Substrings><Arg Name="value" Type="String"/>
      <Constant Type="String" Value="ERASMUS"/></Substrings>
    </AND></IF></TargetAccess></TargetAccessPolicy>
</X.509_PMI_RBAC_Policy>

```

---

### I.B.3. WS-POLICY

As of September 2007, WS-Policy is a W3C recommendation that represents a set of security policy specifications dedicated for web services only. These specifications describe the business (i.e., including security) policies capabilities and constraints on agents and end-points (e.g., required security tokens, supported encryption algorithms, and privacy rules). The specifications also describe

the association of policies with services and end-points. The WS-Policy specifications allow representing web services policies using XML (for security, quality of service, etc.) and allow the consumers of web services to specify their policy requirements. The WS-Policy framework aims at allowing web services to express their constraints and requirements using *policy assertions* (i.e., an assertion is an individual requirement, capability, other property, or a behavior). WS-Policy is featured with its flexibility concerning token types, cryptographic algorithms and mechanisms used. The intention behind this flexibility is providing enough compatibility and interoperability information determined by web service participants (i.e., along with any other necessary information) in order to actually allow a participant to enroll in a *secure messages exchange*. (WS-Policy Specification, 2006)

The policy assertions for a web service are in forms of requirements or advertisements. The operators tags for policy combinations are XML tags mainly. The *wsp:ExactlyOne* asserts the satisfaction of only one child node. The *wsp:All* asserts the satisfaction of all child nodes.

An example on WS-Policy is brought from the WS-Security Policy specification document (Della-Libera, 2005). Table 2.3 shows an example on a security "*Effective Policy*". The policy includes following assertions (i.e., requirements or other properties) concerning: bindings, associated property, token, and together integrity and confidentiality. By explaining each line:

- Line 1: The main *wsp:Policy* indicates the beginning of a policy statement for what all beneath elements are required to be satisfied.
- Line 2: In this tag, the kind of security binding is defined as *symmetric*. This means, for instance, that the encryption token used from the message initiator to the message recipient is also used from the recipient to the initiator.
- Line 3: Contains an example of a nested policy *wsp:Policy*. Its elements are assertions that qualify the behavior of the *SymmetricBinding* assertion.
- Line 4: Indicates a *ProtectionToken* assertion. It involves the population of the *Encryption Token* and the *Signature Token properties*. It is used mainly for the message protection using signature and encryption.
- Line 5: Another nested policy element (*wsp:Policy*) that indicates the type of token to be used for the *ProtectionToken*.
- Line 6: Holds that a Kerberos V5 APREQ token is to be used by both parties in a message exchange for protection.
- Line 9: Specify that signatures must be generated over plaintext and before encrypting it into cipher text.
- Line 10: The assertion implies the requirement of the encryption of the signature (i.e., made over the signed messages parts).
- Line 13-16: These lines indicate what the primary signature must cover of the message parts. In this case, the signature is covering the *soap:Body* element (line 14) and any SOAP headers in the WS-Addressing namespace (in line 15).
- Line 17-19: These lines indicate what message parts must be encrypted. In this case, only the *soap:Body* element will be encrypted (line 18).

**Table 2.3: Example on WS-Policy representation for security policy (Della-Libera, 2005).**

---

```

<wsp:Policy>
  <sp:SymmetricBinding>
    <wsp:Policy>
      <sp:ProtectionToken>
        <wsp:Policy>
          <sp:KerberosV5APREQToken sp:IncludeToken=".../IncludeToken/Once" />
        </wsp:Policy>
      </sp:ProtectionToken>
    <sp:SignBeforeEncrypting />
  </wsp:Policy>
</sp:SymmetricBinding>
</wsp:Policy>

```

---

---

```

        <sp:EncryptSignature />
      </wsp:Policy>
    </sp:SymmetricBinding>
    <sp:SignedParts>
      <sp:Body/>
      <sp:Header Namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
    </sp:SignedParts>
    <sp:EncryptedParts>
      <sp:Body/>
    </sp:EncryptedParts>
  </wsp:Policy>

```

---

One of WS-Policy standard family major drawbacks is letting the policy assertions of the WS-\* standards to have very big XML structures. This makes these assertions incredibly difficult to be expressed by humans (Simon et al., 2011). Another drawback of WS-PolicyConstraints is its dependent on the message structure and syntax. Thus, it is difficult to map between different vocabularies. Moreover, the matching for two policy assertions requires checking the XPath queries content. (Speiser, 2010)

## II. Configuration Management

Configuration management aims at applying changes to the managed system (or to its entities) in order to make sure it meets the security requirements. For instance, the governance of systems implies providing a new configuration for firewalls and routers when new hires arrive concerning their allowed devices. Therefore, it is crucial for any security management system to have the ability to provision configurations. Configurability is the ability to initialize and dynamically modify all managed targets without interruption (i.e. subjects, objects, actions and environment by modifying, for example, parameters, start tracking activities, or change database paths) and the values of the attributes governing the behavior of an enforcement process (e.g., location of a software component or adding auditing mechanisms) (Moui, 2013, p. 59). A configuration of a system consists of collecting specific functional or non-functional parameters or attributes (also known as configuration data). The parameters' values govern the expected functionalities and behaviors that the managed system should convey. Continually, the reconfiguration process is the act of modifying or adjusting the existing configurations (Akue, Lavinal, Desprats, & Sibilla, 2013).

The *configuration*, or called also *setting*, is the simplest, maybe the oldest, adaptation methodology for an application. The software application holds a predetermined variety of parameters that guides its execution. For example, the router's routing table, the number of simultaneous connections for an Apache web server, or simply the IP address of a machine. The variable can be configured or customized at the time of instantiation (Static adaptation) or during the runtime (dynamic adaptation). The configuration mechanism is provided either as parameters during the application launching, based on a configuration file, or by using an external configuration manager. The latter solution is the one that offers the most flexibility, but it is also the most difficult to implement (Desertot, 2007). Upon the configuration (or reconfiguration) of software to bestow a dynamic adaptation, the management of the behavior in terms of the software and its components life cycle becomes more crucial. The decision of applying a configuration requires the consideration of the transition between the old and the new one. For example, the routing software module may probably need to complete performing the routing of waiting IP datagrams inside a queue, before taking into account the new routing table (Cheaito, 2012, p. 51).

The management world is concerned about the importance of the configuration process to the level where studies are interested about the information of the managed systems. Standards from the Distributed Management Task Force (DMTF) have been proposed to provide mechanisms for the

definition and use of management information for systems, networks, applications and services. The striving objectives of provided standards are to represent all IT components through management models, store management information in data structures and to allow data repositories access and updates. Currently, there are several initiatives dedicated to the representation of these standards. Previous efforts to meet these management standards were presented through several models, such as those used by SNMP (simple network management protocol), DMI (distributed management interface), and CMIP (common management information protocol). Such models are actually unified using the Common Information Model.

The Common Information Model is the most common model used for the system and network management. However, there are other information models about managed resources like: the SNMP architecture using the MIB/SMI (Rose & McCloghrie, 1990; McCloghrie, Case, Rose, Waldbusser, 1999; Schoenwaelder, Jason, Strauss, Weiss, 2001). The information model MIB/GDMO (ISO 10165-4) is used for both architectures OSI and TMN, and the policy based management uses the PIB/SPPI (RFC 3159). However, these information models do not represent the advanced and concrete technologies and do not consider the abstraction offered by CIM. Moreover, the frameworks using these architectures (e.g., SNMP, CMSP) are incompatible with each other. Therefore, each framework ought to get its own application domain – despite the common interest of these frameworks and their managed environments.

Nevertheless, there exist another recent promising model taking on its own the support of a unique network configuration protocol (NETCONF). The data model Yang, until 2013, is not yet available as a complete standard approach for NETCONF users. However, the Yang data modeling language is available and supports the expression and description of complicated and complex configurations. It is more powerful than CIM, SMI/SNMP and SPPI. The Yang language is able to describe complex routing tables (supporting list of routes) and critical configuration operations (locking of data sources during multiple accesses). Yang does not currently support Software Defined Networking (SDN) specifications. However, the architecture proposed by Yang for NetConf supports SDN and is standardized by IETF (Shafer, 2011). As a conclusion, one can say that Yang needs a little more time to get the light from CIM and to be considered as fully matured, especially that it is not totally supported by the technical leaders like CISCO. (Schonwalder, Bjorklund, & Shafer, 2010; Knertser & Tsarinenko, 2013)

## II.A. Expressing Management Policies through Languages

In the literature, studies have identified the management policies in a similar way like the security policies based on two structures CA and ECA. Obligations are “*the actions that must be performed by manager objects within the system when specific events occur and a set of conditions are met*” (Boutaba & Aib, 2007). Management policies are languages eventually oriented by obligations to ensure the management task. Obligation policies are based on either ECA structured languages and therefore obligations are invoked upon specific event(s) occurring. As another choice, policy languages are structured as CA rules to represent a special form of the obligation policies. In such cases, obligations are stored as configurations to be used, fired or installed when necessary (e.g., in form of parameters). Both ECA and CA policies seek meeting the management requirements through providing the *provisioning* feature in order to adapt the behavior (i.e., provision management configurations).



Studies on management policies identify languages' abilities to represent the expressed policy with an obligations orientation. That is, policies are interested in prioritizing management objectives – namely configuration ones based on management information models. The following sections are examples on the most common management policy languages.

## II.A.1. CIM/PCIM-SPL DMTF management solution

CIM is an object-oriented, vendor- and system-independent information model defined by the DMTF used to describe various components of managed systems, including software and hardware elements and their relationships. Based on an object-oriented modeling approach, CIM describes the elements of IT systems using object classification, and specifies relationships among objects using object-oriented concepts, such as inheritance and dependencies. The goal of CIM is to model all aspects of the managed environment, including systems, devices, networks, operating systems, applications, and even users.

CIM is not simply a model, but contains a metamodel as well (DSP0004, V 3.0). It allows the integration of other different information models, especially MIB/SMI and MIB/GDMO (Nataf, Festor, & Doyen, 2003). Based on the CIM metamodel (see Figure 2.10) (Vergara, Villagrà, and Berrocal, 2005), the CIM language expresses all management information using these UML basic elements.

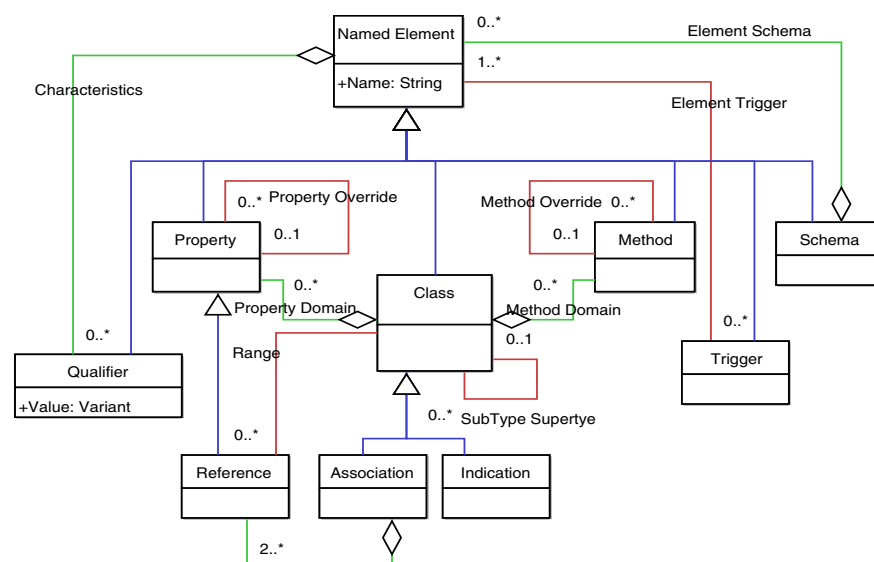


Figure 2.10: CIM metamodel schema

The CIM language aims at describing all managed systems and networks usually used components. The metamodel helps the definition of the CIM language through three levels of abstractions:

- The basic model that defines the classes and the generic associations toward all management domains;
- The common model that describes the classes and associations specific to each domain of management (e.g., applications, services, users, networks), but however independent from the implementation;
- Finally, the extensions that are specified for each implementation.

Nevertheless, the world of management consider the configuration policies in terms of representations as an externalized logic that determines the behavior of managed systems, networks,

applications, and services (Agrawal, Lee, & Lobo, 2005). Following this reasoning, each policy language depends on two information models, one for the resources being managed, and another for the policies themselves (Agrawal, Calo, Lee, & Lobo, 2007).

Therefore, the IETF Policy working group, a leader in management standards and solutions, mostly carries out the ongoing studies on policy specification and expression concerning the configuration management. IETF does not specify policies using a language, but through an object-oriented information model (Elleson, Strassner, Moore, & Westerinen, 2001). The DMTF developed the Policy Core Information Model (PCIM) model as an extension of the Common Information Model (CIM) to express management policies – strictly expressions without real implementations. In Figure 2.11, a policy rule aggregates policy conditions and policy actions. Regarding this representation, the policy expression follows the statement of the CA structure: *if (set of conditions) then execute a set of actions*. (Lymberopoulos, 2004)

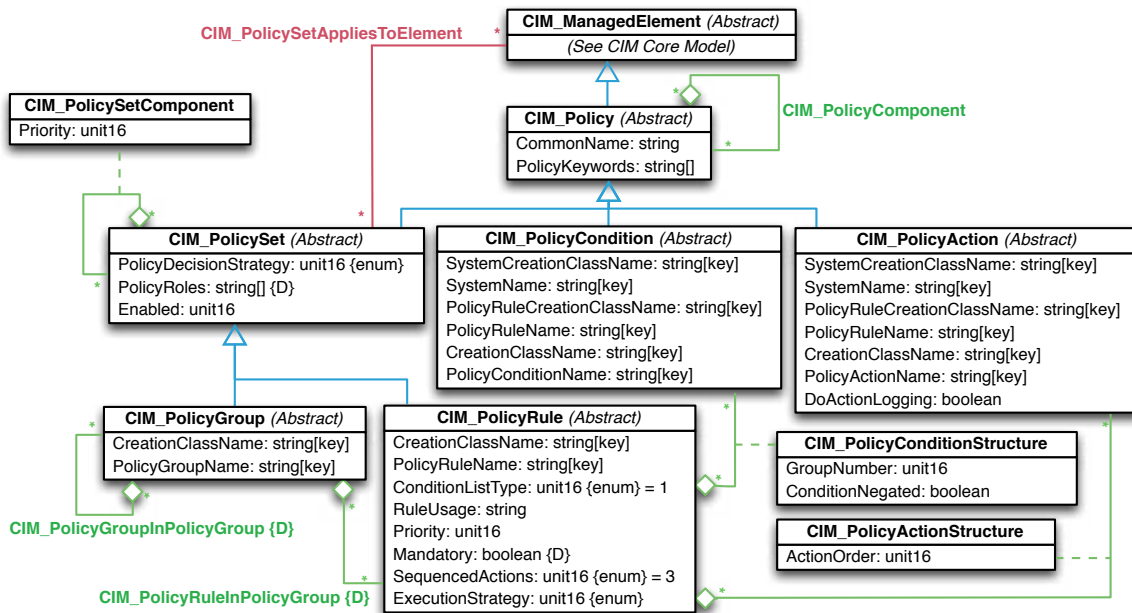


Figure 2.11: Detailed subset of the CIM policy schema v2.38.0

For a better understanding of CIM schema, let's take a simple example on the management information representation for a network printer. Supposing that a printer is connected to an intranet, it contains several parameters such as an IP address, description, ink percentage, serial number, etc. In order to configure the printer (or reconfigure it), one needs to consult the existing information and the possibility of modifying the settings.

With CIM, all classes inherit from the class *CIM\_ManagedElement* in order to specify and express the management information of the managed resources. However, the most generic, appropriate and simple class to represent the management information of network resources like the printer is *CIM\_ComputerSystem* – the switch, router, firewall, etc. can be seen as computer systems as well.

Therefore, the previous explanations show only the modeling on where to find the management information of the printer in order to configure its parameters. However, the CIM solution needs to work on top of a framework in order to apply the standards. Therefore, DMTF presented the WBEM framework. WBEM proposes a DMTF standard framework and architecture for CIM and implement tools like Open Pegasus to provision configurations.



Nevertheless, the configuration management handles models in such a similar way as in access control which aims at describing and providing guidelines for network and security managers to write their policies. Even though the comparison is stated, it is not very balanced. The configuration management addresses and models much more complicated management information than the common access control models do concerning the information about the protected resources – in terms of variety of information and management metrics. However, the truth about both models is that they need languages, namely policy languages, in order to use the information and apply, specify and implement the management and control decisions and actions. Therefore, the configurations need to be refined to low-level languages in order to be technically implemented. In other words, using CIM alone through WBEM will only ensure the supervision task of management. The governance task needs to be ensured using the notion of *policy* (i.e., including policy languages, architectures, frameworks, etc.).

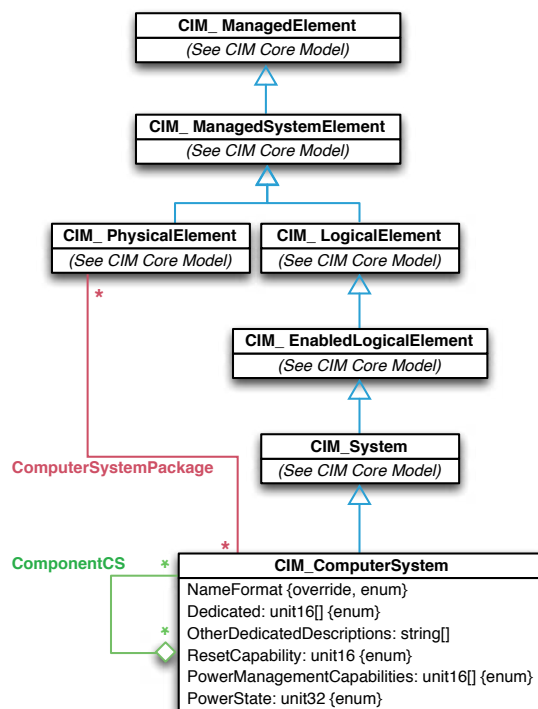


Figure 2.12: Extract from the CIM system schema v2.42.0

### II.A.1.1 CIM-SPL

In the configuration management section, the presented study on CIM illustrated its remarkable importance for the modeling of the management information. However, we presented also the lack of DMTF CIM model for having a policy language to render and complement its policy model – proposed by the DMTF Policy Working Group. Therefore, CIM-SPL (Simplified Policy Language for CIM) has been introduced to comprise CA structured rules (Agrawal, Calo, Lee, & Lobo, 2007). This version of CIM-SPL did not support the reaction to events and therefore the DMTF community proposed the use of an external Event Model (DMTF DSP107, 2003). In this case, an externalized party provides the event processing and handling for the CIM-SPL language. The event model defines event-related abstractions and describes the CIM indications hierarchy. The modeling aims at designing the information about all the events that can be detected by the supported technologies in CIM. Upon the receiving of a CIM InstIndication instance (i.e., a subclass of CIM Indication), this implies the occurrence of an event and the need to make actions. Therefore, the corresponding policy is called entirely for evaluation. This evaluation is known as an unsolicited policy evaluation. The other supported type is the solicited evaluation. This type is proper to the policy

enforcement point to call all policies for the evaluation that are relevant to a specific resource. There are only two strategies to apply the evaluation: evaluate all applicable policies or evaluate the first applicable policy (DMTF DSP0231, 2009).

**Table 2.4: An example on CIM-SPL policy (Boutaba & Aib, 2007).**

---

```

Import CIM_X_XX_XXXX::CIM_LocalFileSystem;
Strategy Execute_All_Applicable;

Policy {
    Declaration {
        computer_system = collect(Self, CIM_HostedFileSystem, PartComponent,
            GroupComponent, null, true)[0];
        storage_config_service = collect(computer_system, CIM_HostedService, Antecedent,
            Dependent, CIM_StorageConfigurationService, true)[0];
        logical_disk = collect(Self, CIM_ResidesOnExtent, Dependent, Antecedent, null, true)[0];
        storage_pool = collect(logical_disk, CIM_AllocatedFromStoragePool, Dependent,
            Antecedent, null, true)[0];
        fs_goal = collect(Self, CIM_ElementSettingData, ManagedElement, SettingData,
            CIM_FileSystemSetting true)[0];
    }
    Condition { (AvailableSpace / FileSystemSize) < 0.25 }
    Decision {
        storage_conf_service.CreateOrModifyElementFromStoragePool("LogicalDisk",
            /* ElementType Volume */
            8, job, fs_goal, 1.25 * FileSystemSize, storage_pool, logical_disk) |
        DoSomethingOnFailure()
    }
} :1 /* policy priority*/;

```

---

These evaluation processes at each event (or combination of events) were considered as an exhaustive process each time to evaluate all the rules in the corresponding policy (or even all the policies). Therefore, the proposition for an extension to CIM-SPL was made (Pan, Lobo, & Calo, 2009). Based on the WBEM management framework, the implementation of new providers to detect and react on events (i.e., software modules that are responsible of specific tasks not supported initially by WBEM). However, the processing of events was still not supported. The CIM-SPL extension served on supporting the management requirements, and additionally supports the requirements of access control through the representation of the RBAC model. The only implementation of this language was proposed through a policy engine during the Open Group's Open- Pegasus project (Han & Lei, 2012).

IETF PIB, PDL, and DMTF CIM-SPL languages have been introduced mainly for the objective of managing networks and systems. However, PIB expressed the need for a policy model and an externalized party to process events. PDL drawback is the conflict resolution and the complexity of its event model. Finally, CIM-SPL needs an externalized party to manage and process its desired events.

## II.A.2. IETF PIB

In distributed heterogeneous systems, access control policies that follow RBAC (i.e., proposed by the NIST) are distributed based on the *provisioning* strategy defined by IETF. The strategy is about considering the management information to be distributed to the targeted enforcement elements using the COPS-PR protocol. PIB (Policy Information Base) is an information representation for the policies modeled by RBAC.

The PIB using the provisioning approach follows three main points (Westerinen et al., 2001). The first point is about representing the policy information using a model (e.g., PCIM or PCIMe). The second point is to represent the policies using PIB for each independent specific device (i.e., including translating the policy to be understood into the device capabilities in order for the device to be able to enforce this policy). The third point is related to the communication phase of policies. COPS-PR

(Chan et al., 2001) is a protocol designed for negotiating capabilities, transporting and installing the PIB structured policies to be configured into the targeted device.

The IETF provisioning approach or strategy is a generic solution for the management domains and not dedicated to solutions through PIB only. In addition to PIB, the distribution of configurations has been proposed for DiffServ and IPsec recently. As presented previously, the PCIM and its extensions (PCIMe) (Moore, Elleson, Strasser, Weterinen, 2001) represent policies information for any management domain using a generic set of classes and associations. Therefore, IETF PCIM represents a good, yet generic, candidate for modeling the information for a policy expressed and specified following the PIB approach.

To conclude the study on PIB, the PIB solution is not enough standalone to fulfill the management needs in terms of representing the policy and its information without the usage of a modeling solution (e.g., PCIM) nor for sure a communication protocol (i.e., COPS-PR).

### II.A.3. PDL

Another example of the management policies language provided by IETF is the Policy Description Language (PDL) from Bell Labs. PDL is the first language introduced for network administration with rules structured on ECA (Lobo, Bhatia, & Naqvi, 1999). PDL aims at managing the network based on a declarative language in order to define policies. Policies rules take the form of “event(s) *cause(s)* action(s) *if* condition(s)”. PDL introduces another language to declare and manage the events separately using Policy Defined Event (PDE) language. PDE follows the form “event(s) *trigger(s)* policy-defined-event *if* condition(s)”. This language processes events in several forms and accepts the expression of complex operations on them based on the concept of time window (e.g., combination and coordination) (Strassner, 2003, “How to Express Information in a Common Way”, p. 126-127). Examples on PDL proved the expressiveness of the language (Kohli & Lobo, 1999). However, the language lacks the ability to handle aggregate or composite policies and requires an additional programming language support. The PDL language causes the requirement for excessively complex PDE hierarchies (i.e., because one PDE keeps triggering one (or more) PDEs, until an action or sequence of actions is performed). Importantly, PDL has a limited support for conflict resolution, and with complex PDE hierarchies it becomes more limited.

**Table 2.5: Sample PDL policies (Chomicki, Lobo, & Naqvi, 2000).**

---

```
// policy for product order processing
defectiveProduct causes stop
orderReceived causes mailProduct
orderReceived causes chargeCreditCard

/* policy for Soft switch overload control in the presence of an excessive number of signalling network
time outs */
(normalmode, group(callmade|timeout)) triggers overloadmode if
(Count(timeout)/Count(callmade)>ThOvld)
```

---

One of the CA structured languages to represent the network administration rules is PFDL (Policy Framework Definition Language) (Nicklisch, 1999). PFDL is a definition language necessary for the expression of the PCIM model. However, the work on PFDL did not continue by the IETF policy group. Therefore, the language stayed as a general-purpose language proposed for the PCIM model to be used to specify rules through a management console (PAP) to provide a view for the administrator to express QoS, access control, and customized web applications management policies.

## II.A.4. The Ponder Policy Specification Language

Appeared in 2000, Ponder is a language, framework, and architecture, which are mainly oriented for network management. The language supports authorization, obligation, and configuration. Ponder is an event-oriented language that keeps an eye on the QoS through performing monitoring (surveillance) on network behaviors and activities based on predefined metrics (accuracy, performance, etc.) (Damianou, Dulay, Lupu, & Sloman, 2001). However, the security side is assured by a special representation of the authorization policy that adapts to Ponder and RBAC requirements only. Ponder gets a higher score when it comes to management through the support for obligation than its score for security management. The lack of support limits its capabilities to compete with other access control implementations. Four groups represent the different types of Ponder policies: positive and negative authorization (i.e., including policies for information filtering), obligations, restrictions (i.e., using the refrain syntax) and delegation.

Ponder is defined as a declarative language, object-oriented, flexible, expressive and extensible, allowing security and management policies creation for distributed systems. As a key concept, roles provide Ponder with the semantic of grouping policies based on subjects (e.g., DiffServ edge router). Ponder is an RBAC-Based framework. However, the only contradiction between Ponder and the RBAC model is the support missing for the inheritance of privileges within a Role Instance Hierarchy. For the composition of policies, Ponder uses the concept of *domains* by constructing sub trees identified by abstract paths. The *domains* represented in the policies work as a repository to group and prioritize all Ponder elements (i.e., subjects, targets, policies). To formalize the concept, Ponder defines Meta policies to define permitted policy types and authorized concurrent action sequencing (Polyrakis & Boutaba, 2002; Sloman, 1994).

Ponder major drawback is its expressiveness. Moreover, it leaks the temporal structure that provides, in particular, history-based SOD (Separation of Duty, that is, having more than one subject required to complete an action (e.g., a task)). Ponder policies do not support workflow management. They are a matter of containing conflicting rules, especially positive authorization and negative authorization rules (Jin, Krishnan, & Sandhu, 2012).

Based on the ECA scheme, Ponder obligation policies aim at describing the actions that must be performed when a particular event occurs. Ponder obligation policies can help the expression of three kinds of policies:

1. Security policies: by allowing a security officer to specify the actions to be taken when a security policy is violated (exceptions);
2. QoS management policies: by allowing the QoS manager to manage the quality of service offered to users (e.g., scheduling systems backups or performing software configurations),
3. Auditing policies: by allowing the specification of what information is to be logged, where and by whom.

## II.A.4.1 Examples

The first example on Ponder policy is for managing the QoS of the network by monitoring the connections made by network users. Once a connection failure event occurs (i.e., is detected through the ConnectionFailure obligation), the Ponder framework starts counting the failures. Upon the third time event occurrence on a particular identified user (i.e., the line “3 \* FailedConnection(userId)”), the user identification will be disabled in the managed organization. This deactivation action is logged “t.deactivate () -> s.connection (userId)”.

Table 2.6: Example 1, connection failure policy

```
inst oblig ConnectionFailure {  
    on 3 * FailedConnection (userId);  
    subject s=/region/admin;  
    target <User> t=/region/users^{userId};  
    do t.deactivate () -> s.connection(idutilisateur);  
}
```

Another trivial example to present Ponder for a QoS management policy is when the performance within a managed environment goes down – the performance is measured regarding the bandwidth. In this example, the obligation policy type perfIncreaseT specifies what to do as reactions on events. A performance degradation event perfDegradation(bw, source) triggers the obligation policy. The perfDegradation event parameters (bw, source) are reused for the specification of the actions. Sequentially, the policy subject invokes the action bwReserve(bw) and log(bw, source) on the target object. The subjects interface is responsible for the implementation of the log action. Respectively, the name assignments s and t for the subject and target in policy perfIncrease used in the actions specification. The target type is restricted only to those whose their IDL is Router – see the angle brackets next to the target attribute specification of the perfIncreaseT policy type.

Table 2.7: Example 2, bandwidth performance management (Damianou, Dulay, Lupu, & Sloman, 2000).

```
type oblig perfIncreaseT (subjects,target<Router>t) {  
    on perfDegradation(bw, source);  
    do t.bwReserve(bw) -> s.log(bw, source);  
} // perfIncreaseT  
  
inst  
    oblig pl = perfIncreaseT(brEngineer, coreRouter/+edgeRouter/);  
    oblig perfIncrease {  
        subject s = brEngineer;  
        target t = coreRouter/ + edgeRouter/;  
        on perfDegradation(bw, source);  
        do t.bwReserve(bw) -> s.log(bw, source);  
    } // perfIncrease
```

In conclusion to the study on Ponder, the solution presents a pure management solution oriented by obligations using rules structured as events, conditions, actions. Therefore, research studies presented that the most complete solution to manage, represent and express management policies orientated by obligations and configuration is the Ponder solution. Ponder is able to process and manage the events it needs to enforce its policy ECA rules. In addition, the obligation policy languages claimed to support security policies, namely authorizations, and also QoS policies. Obligations are ensuring configuration provisioning in the form of restrictions, delegations, or direct actions (e.g., logging, downgrading the bandwidth to recover the performance).

## II.A.5. NETCONF configuration language

We highlighted that there are many languages in the management world that do not provide an access control support, but manage and administrate the configuration of networks and systems devices (objects) (i.e., configurations include the distribution of management policies). Configuration languages express rules in both structures CA and ECA in order to perform management functions on the behavior of systems and networks. The management functions could be limited to perform configuration actions like (set, update, etc.) or it could be more complicated configuration task like in the new NetConf (i.e., responding to detected events and perform the right relevant actions).

With the intention of visiting some configuration languages, NETCONF is mainly used for configuration management. It uses operations to configure network devices based on RPC layers (Wang, Zhang, Li, Li, & Gao, 2010). In 2000, IETF SNMPconf working group started a provisioning solution designed for configuring network elements (Chadha & Kant, 2008). Its motivation refers to the absence of a standard protocol suitable for network elements configuration. Therefore, the SNMP community launched SNMPconf well suited for configuration and monitoring of network elements. The workgroup developed the MIB to represent the desired network-wide DiffServ-based QoS behavior (McFaden, Partain, Saperia, & Tackabury, 2003, IETF RFC 3512). SNMPconf through MIBs specifies the configuration data of the managed elements. SNMPconf is a standard configuration policy language using the SNMP communication protocol and Net-SNMP tools to configure and provision network policies. For instance, the command *snmpset -options hostname-address community MIB-objectID a new-ip-address* allows the MIB object IP address to be set (Mauro & Schmidt, 2001, Appendix C, p. 367).

Moreover, there are languages for the management of quality of service (QoS) for the two approaches: the RSVP signaling (IntServ/RSVP) (Braden, Zhang, Berson, Herzog, & Jamin, 1997) and the differentiated services (DiffServ) (Blake et. al., 1998). Per-Hop Behaviour (PHB) uses EF Configuration Policy (Ferrari & Chimento, 2000; Jacobson, Nichols, & Poduri, 1999). The configuration management of network policy information (CA structured) is managed and represented by a PCIM (policy core information model), defined by RFC 3060. Policies are storable in a repository and are downloadable by management information base (MIB) scripts directly to the managed nodes. There is a special version of PCIM for the QoS (QPIM) (Snir et al., 2003). IETF PCIM with DMTF/QPIM provides a configuration management solution – namely traffic flow management network device configurations – that defines policy actions for Differentiated Services (DiffServ) and Integrated Services (IntServ) networks. A QPIM policy rule example for establishing an EF Per Hop behavior (EF-PHB) on a DiffServ node is “*If (traffic belongs to EF aggregate) then do EF-actions*” (Lymberopoulos, 2004).

Nevertheless, NETCONF seems to be the best nomination to give an example on configuration languages. NETCONF is one of the most powerful solutions, because of recently proposed with revolutionary features that are specified to meet what management expects inside a management policy. However, there is not yet a complete implementation to all features proposed in NETCONF. Moreover, security is not a priority for this solution (Seitz & Rissanen, 2008; Wang, Zhang, Li, Li, & Gao, 2010)!

NETCONF respects the ECA rules structure and represents a sort of reaction rules (*On events if condition(s) then action(s)*). However, the policy language expression in which the structured rules are written is still shy from getting out to lights. The YANG representation language aims at “*modeling data used to model configuration and state data manipulated by the Network Configuration*



*Protocol (NETCONF), NETCONF remote procedure calls (RPC), and NETCONF notifications*". (Shafer, 2011, RFC 6020)

Therefore, performing configuration management using for this IETF solution based on policies needs both NETCONF and YANG together. NETCONF offers a conceptual model of high-level layers to use while managing configuration (see Figure 2.13).

Layer	Example
Content	Configuration Data
Operations	<get-config>, <edit-config>,
RPC	<rpc>, <rpc-reply>,
Transport Protocol	BEEP, SSH, SSH, console

**Figure 2.13: NETCONF architecture (Chadha & Kant, 2008, p. 168)**

Many NETCONF configuration operations help the administration and management of networks and systems. The language introduced the concept of “data store”, *a network device uses to store information about the configuration of the device* (Chadha & Kant, 2008, p. 171). The *get-config* gets all or part of the configuration from the data store and the *get* operation gets the same plus the state data as well. The *edit-config* edits the configuration in the data store by allowing functions like merge, replace, create, or delete on the configuration elements. There are other operations as well like *copy-config* or *delete-config* (to copy or remove the totality of an existing configuration). Useful operations like validate (takes the candidate configuration and verifies its syntax and structure in order to report the existing errors), commit (commit a configuration candidate (e.g., modified) toward the target running element based on rules like *all* or *nothing*), and restore (recover back the candidate configuration in the data store from the current configuration that is running).

The lock / unlock operations are the most interesting operations that allow the admin to place / remove the lock of the data store where the configuration is stored. Like in database management systems, the idea is to prevent other clients from updating the configurations while it holds the lock. The main parameters that this operation takes are the *target* that defines the name of the configuration data store to lock / unlock and the *response* of the policy decision of allowing or not allowing the locking / unlocking and communicate it through the usage of IETF RPC (Remote Procedure Call). (Enns, Bjorklund, Schönwälder, & Bierman, 2011, RFC 6241)

**Table 2.8: Example of using the lock operation in NETCONF.**

---

```

<rpc message-id="101"! xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">!
  <lock> <target> <running/> </target> </lock>
</rpc>
<rpc-reply message-id="101"! xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <rpc-error> <!-- lock failed -->
    <error-type>protocol</error-type>
    <error-tag>lock-denied</error-tag>
    <error-severity>error</error-severity>
    <error-message> Lock failed, lock is already held </error-message>
    <error-info> <session-id>454</session-id> </error-info>
  </rpc-error>
</rpc-reply>

```

---

Based on the configuration requirements defined originally during an IAB workshop that paved the way for NETCONF and YANG (Schoenwaelder<sup>8</sup>, 2003, RFC3535), the following table shows the difference between SNMP and NETCONF in terms of configuration management.

**Table 2.9: Comparison between SNMP and NETCONF (Schönwälder, 2012).**

No	Description	SNMP	NETCONF
R1	Configuration vs. Operation state	-	+
R2	Concurrency support	O	+
R3	Configuration transactions	-	[+]
R4	Multiple Configurations	-	[+]
R5	Distribution vs. activation	-	[+]
R6	Persistence of configuration state	O	+
R7	Configuration change notifications	-	+
R8	Configuration dump and restore	-	+
R9	Support of standard tools	-	+

---

<sup>8</sup> The name in references is written in both ways “Schoenwaelder” and “Schönwälder”.



## Chapter's Conclusion

---

*The Chapter 2 presented the expression of policies in both worlds the management and security. The security community first concentrates on models to provide expression templates.. Security solutions either apply templates to policy languages (e.g., PDL and Ponder applying RBAC model), or are totally specific solutions without any attachment to access control models (e.g., APPEL). The management community represents policies directly through languages. Two names can be found in the literature to describe the languages for management policies: configuration languages (e.g., NETCONF) and obligation languages (e.g., CIM-SPL).*

---

# Chapter III:

## Security & Management Architectures

---

*The Chapter focuses on the State-of-the-art architectures to implement the policy, within the security and the management communities. Hence, we divided the chapter into three sections to illustrate different studies on architectures. The first section qualifies in defining a general architecture for any security management system. The second section presents the different existing management architectures and the third one presents the different existing security architectures.*

---

### Summary

#### ***I. General Security Management Architecture***

- *Architectures with management objectives*
- *SNMP-Based Architecture*
- *Policy-Based Management*
- *PBM Communication Paradigms*
- *DMTF WBEM Architecture*
- *Ponder Architecture*
- *Concluding management architectures*

#### ***II. Architectures with security objectives***

- *AAA Architecture*
- *OASIS XACML Architecture*
- *Concluding security architectures*

# Chapter 3: Security & Management Architectures

---

*"I tell you, sir, the only safeguard of order and discipline in the modern world is a standardized worker with interchangeable parts. That would solve the entire problem of management."*

*Jean Giraudoux*

*Columbia World of Quotations*

*Retrieved October 31, 2014, from Dictionary.com*

---

## Introduction:

Due to the rate at which new technologies are introduced for systems and networks, security management of targeted environments faces significant challenges. The business requirements for ubiquity and mobility have created more heterogeneity in these environments, making the management task even harder. Therefore, there are always efforts toward and calls for the unification of approaches and architectures. As standards try to create and provide the most suitable solutions, standardization leaders aim to improve the management of security in systems and networks. For example, the Internet Engineering Task Force (IETF) and the Organization for the Advancement of Structured Information Standards (OASIS) specialize in standardizing access control architecture. In terms of system and network management, the Distributed Management Task Force (DMTF) with IETF, the International Telegraph Union-Telecommunication Standardization Sector (ITU-T) and the Telecommunications Management Network (TMN) are leaders in providing standardized architectures.

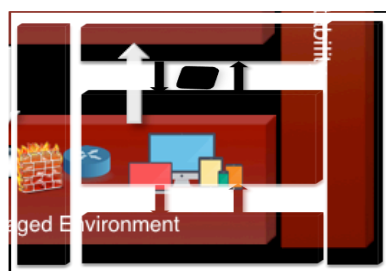
One should point out that the security policy expression and architecture are strongly connected. Any expressed security policy using a language needs to be applied and used by an architecture in order to be effective. Therefore, the problem presented in this chapter is strongly related to the expression issues discussed in Chapter 2.

In security management, one can recall that the security policy has two tasks to fulfill. The first is security, which is usually undertaken using authorization. The second task is management, which is expressed as obligations. Therefore, after the expression of the security policy, the architecture that enforces this policy should accomplish these two tasks. The access control architecture ensures that the implementation and the application of the security policy at the environment level meet the system and network requirements (that is, the decision-making and the enforcement process). The application of the security policy is required for management decisions. The implementation of the security policy is required to deploy all necessary mechanisms (i.e., hardware or software) in order to enforce the management decisions.

# I. General Security Management Architecture

Studies on the architecture of security management are influenced by the heterogeneity previously presented in the security policy expression. As networks and systems management and access control are studied separately (Clemm, 2006, pp. 157–159), it is natural too that heterogeneous architectures will be found among the existing security management solutions. For instance, network management proposes different management solutions in terms of frameworks, architectures, and languages—for example, the Open Systems Interconnection (OSI) framework and model (ISO<sup>9</sup> 10040, 1998), the ITU-T TMN framework (ITU-T Recommendation M.3010), and the IETF SNMP-based architecture (Harrington, Wijnen, & Presuhn, 2002). Some architectures are expression-related solutions like SNMP, and others are domain-specific language (DSL) expressions. In access control, most studies focused on models have been conducted since the 1990s. Therefore, security policy models that are “*applied when needed*” (e.g., RBAC, OrBAC, UCON, etc.) apply specific architectures that follow the models’ expressions and requirements with respect to the environments’ capabilities.

Despite the existing architectures’ heterogeneity, the mentioned expression requirements can help in proposing a general view on what security management should respect in terms of architecture (see Figure 3.14). The systems that ensure the management of the security, should respect the requirements of the managed systems and networks (that is, expressed in the security policy) in terms of the CIA requirements.



**Figure 3.14: General architecture of the security management.**

Any security management system (SMS) architecture should understand the expressed security policy by being able to (a) implement it by making decisions based on it, (b) communicate these decisions, and (c) enforce the decisions inside the environment. Therefore, the architecture needs to be flexible in order to include new technologies at technical perceptions and new requirements at

---

<sup>9</sup> International Organization for Standardization

organizational perceptions. Moreover, the SMS architecture needs to be adaptive enough to consider the changes in the environment's behavior.

Unfortunately, the general view is far from being completely respected because of the divergent solutions that provide the SMS architecture and the distance between management and security. Heterogeneity in system and network management exists through different administration solutions toward ensuring the supervision and the governance. Heterogeneity in access control exists through the different authorization solutions to ensure CIA constraints. The distance between security and management refers to the understanding of requirements. Those concerned with systems and networks management consider security as a partial service that should be assured to meet a required level of quality. However, those responsible for access control consider security to be a priority and ignore many management requirements.

By analyzing the different architectures in both domains, it was possible to identify two logics or visions in systems and networks management in terms of security. In systems and networks management, the architectures for management of security can be classified as passive and active. The passive architectures accept or allow what happens in the environment's behavior, without active response or resistance. However, the strength of passive architectures is their posterior reasoning about what situations happened, which occurs prior to giving them a response (i.e., usually expressed in literature as *reactions*). For instance, networks management architectures supervise the networks to watch the behavior, usually using the *polling* strategy. Once the number of notifications has increased, the reasoning of the governance side decides to get over the overload situation. Hence, the governance system adapts the network bandwidth by reducing the notifications number by configuring the *polling* settings. In access control, there are different architectures but still subscribed to the passive class. For example, in smart buildings equipped with alarm systems like intrusion-detection systems, access control systems based on passive architectures would deny the access of an unauthorized subject after detecting an intrusion alert (that is, a situation when an intruder enters a prohibited zone).

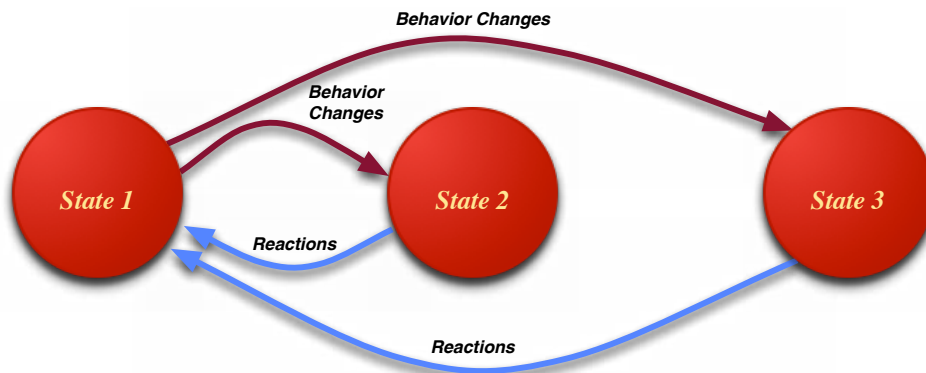


Figure 3.15: State diagram for passive architectures.

The active architectures are farther from the behavior than the passive architectures are. Active architectures aim at blocking the changes to be made to the environment's behavior unless it is authorized instead of adjusting the behavior to become authorized or acceptable. In other words, active architectures are more interested in allowing (or denying) subjects the access to resources (objects) for usage reasons, which will certainly affect the environment's behavior. A suitable example for this architecture would be one involving a *Firewall*. A firewall is responsible for protecting resources from

getting into the *hacked*<sup>10</sup> state, where unauthorized subjects should gain access to make changes in the environment's behavior. However, access requests could be a part of the behavior changes as well; that is when this part needs a permission to continue making behavior changes inside the environment (e.g., segments in the dataflow asking to allocate memory space somewhere in the data store). From a management viewpoint, active architectures are being used to impose obligations in the environment behavior. For instance, in a smart grid, the system may encounter a huge number of access requests where subjects need to use energy sources at the same time—a *gigantic access* situation. In this situation, the access control system (e.g., based on RBAC) would deny all roles equal to users and allow only administrators. With this decision, the access control system may use the *bandwidth cap* as an obligation in order to limit the connection and data usage for requesters during specific periods. By this, the obligation of the access control systems itself imposes changes in the environment's behavior.

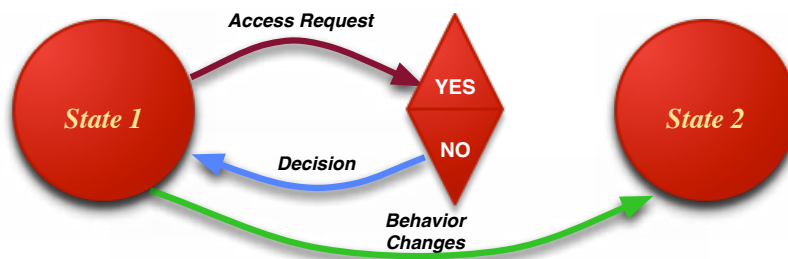


Figure 3.16: State diagram for active architectures.

The two active and passive architectures in literature are totally disconnected, which means that the implementation or deployment is independent and exclusive to the limit of our knowledge. One can apply the active architecture and not the passive and vice versa. This heterogeneity in available solutions for security management defines a problem, and it is what invites research for finding a unified approach (that is, security policy expression and architecture).

## I.A. A look at existing architectures

The ultimate goal of any systems and networks management platform and services stays the facilitation of the administrator and security officer daily tasks, especially facing evolving approaches over time solving the new posed problems. The following three sections try to provide a look on the existing possible architectures that one should consider whenever choosing the systems architecture to deliver a security management solution.

The first section presents one of the famous approaches to provide systems and networks management; policy-based management is a complete research trend that aims at proposing several solutions to adapt the heterogeneous requirements inside the systems and networks environments. Each solution proposes a specific architecture, however, the IETF was able to define a standard initial architecture to propose new ones based on this initiation. Therefore, the literature contains now heterogeneous architectures that may or may not respect the standard architecture.

Nevertheless, and based on the essential objective of each architecture, one can divide the systems and networks management architectures that studies consider for security management solutions into two architectures: management architectures and security architectures.

<sup>10</sup> Gain unauthorized access to data in a system or computer, Oxford Dictionary.

The management premises address the heterogeneity problem into networks and systems following the approach of OSI, IETF and ITU-T. They set the foundations for these disciplines: modeling management architectures, representation of the environment or the integration of management in networks and systems (Bennani et al. 2001). Propagating specific management solutions to adapt specific areas is what introduces the problem of heterogeneity. Therefore, studies are ongoing on the unification of these management architectures for global control of networks and systems. The leader on capturing this problem is the DMTF WBEM research initiative.

The security premises address the heterogeneity problem into networks and systems following the approach from OASIS, AAA and IETF. Other architecture solutions are associated with the chosen security model (e.g., PCIM or context-aware RBAC). Despite the efforts, heterogeneity exist in the security architecture solutions due to either the specificity of the architecture because of its coupling with models or to the architecture's abstraction that need to be adapted before it can be applied to technical constrains of the managed environments. Nevertheless, all the security architectures are poorly supportive to the management objectives (and the opposite is true as well). For instance, OASIS presented the XACML standard with an associated architecture but it does not conform to all the management's world objectives.

The difference between both worlds (management and security) does not stop at the design level of architectures, but also the implementation of these architectures. There are several ways of implementation that assure the communication between the different architecture's components. Therefore, a fourth section is interested in presenting several heterogeneous communication paradigms.

It is important to point out that the objective of the following study is to show that there is no proposition for a unified solution that can reply to all the security management solution in conformity to both objectives: management and security. Therefore, the chapter is structured to present first the renowned management architectures. Afterwards, the following section presents the standardized security architectures. Next, a section introduces the eminent architecture of Policy-Based management. This approach first presented by the management community and then widely adopted in many security, management and even telecommunication researches (Yavatkar, Pendarakis, & Guerin, 2000). Finally, all architectures use communication protocols to exchange information in order to perform management or security. The last section goes through the logic behind each followed communication paradigm.

## II. Architectures with management objectives

Today's user of networks and systems is able to access anytime and anywhere services (email, data, etc.). Networks management must be able to react immediately to any event (for example, failure of equipment or mobility requested by users) to ensure the availability and reliability of services.

The existing approaches in distributed management, for instance, are considered in five models defined by the OSI in [ISO 10040]: the *organizational model* that concerns the identification and definition of entities participating in the activity of management, the *architectural model* that structures the defined entities in the organizational model, the *informational model* for the specification and representation of the management information, the *functional model* that represents the set of management operations to be implemented, and the *communication model* that defines protocols allowing the information exchange between the defined entities in the organizational model.

The management of networks and systems faces the heterogeneity problem in two different areas: computer networks and telecommunication networks. The environments of networks are becoming more sophisticated. Incorporating facilities includes integrating different technologies from different vendors. Therefore, management of networks and systems has become critical with different standards hatched to facilitate this task.

In the context of the OSI model [ISO 7498-4], the ISO standard proposes to treat conceptually many problems in all areas of management. However, with the complexity of its implementation, especially within enterprise networks, the management community preferred the IETF proposition with less powerful – but extremely simple – solution based on TCP/ IP-based networks.

In parallel, integrated management of network and systems has emerged in the world of telecommunications. The ITU-T organization has developed a model called TMN [ITU-T M3010] that is strongly linked to ISO. It responds to some problems related to business telecommunications operators, for example, by integrating the different entities involved in the business of telecommunications (customers, service providers and network providers).

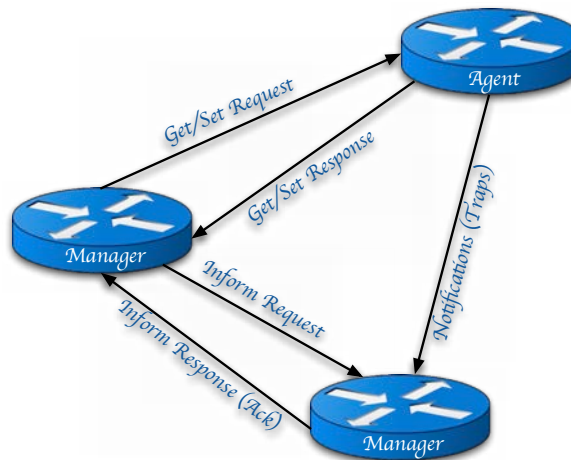
From the available management architectures, we found that SNMP-based and DMTF WBEM are most frequently used in the literature. Therefore, the study will be limited to these common management architectures.

### II.A. SNMP-Based Architecture

The architecture describes the relationship between managed objects (agents) and management machines (managers)—that is, in the management of networks and systems using the simple network management protocol (SNMP) and a management-information base (MIB). On the one hand, an agent-to-manager SNMP-based architecture is based on reporting events coming from the agents about the managed objects (see Figure 3.17). On the other hand, the manager-to-agent architecture aims at governing the agents by provisioning network configurations. Finally, the manager-to-manager is partially an administration purpose architecture; it intends to forward management decisions from one manager to another.

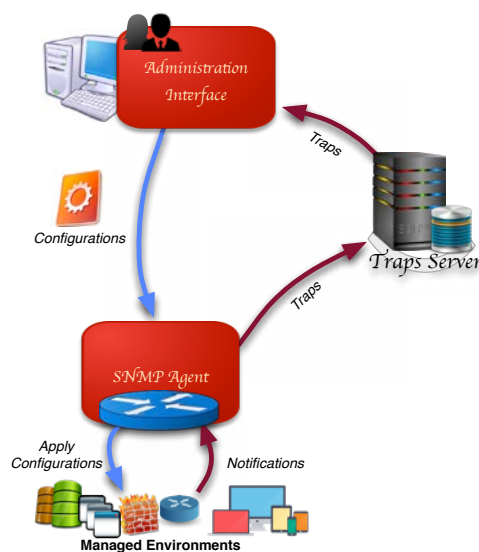


The SNMP-based architecture aims at overcoming the heterogeneity and emphasizing the role of the systems and networks administration. Moreover, its additional advantage is the support of access control requirements.



**Figure 3.17: SNMPv2 agent-manager relationship.**

The technical management architecture using SNMP is presented in the Figure 3.18. The interoperation between agents is made based on notifications that salute the need for configurations to be sent in order to adapt the managed objects to the specified requirements. The SNMP technical architecture is based on the UDP communication protocol. Its main objective is to encapsulate the heterogeneous network devices (e.g., Cisco routers) with software agents. The agents are clients to a Traps server (i.e., responsible of capturing notifications from different agents). Traps servers communicate through a UDP connection the traps to the administrators' interfaces. The latter will propose the suitable configurations through the Net-SNMP configuration language to be applied by the agents. The communication between the administration interface (manager) and the agents is also through a UDP connection (i.e., asynchronized connections).



**Figure 3.18: SNMP technical architecture**

## II.B. Policy-Based Management

Policy-based management (PBM), one of the management approaches, is proposed to overcome the technical heterogeneity of systems and networks. PBM is an administrative approach that is used to simplify the security management of a given endeavor by establishing security policies to drive situations that are likely to occur. Its main objective separates the rules governing the behavior of a system from its functionality. It promises to reduce maintenance costs of information and communication systems while improving flexibility and runtime adaptability. PBM is today present at the heart of a multitude of management architectures and paradigms including SLA-driven, business-driven, autonomous, adaptive, and self-\* management (Boutaba & Aib, 2007).

PBM has been defined from different aspects and perspectives. Multiple definitions of this management approach introduce its adaptability to all domains related to system and network management. Despite the variety in definitions, PBM has a core objective of achieving business requirements while ensuring the management of networks and systems.

In the late 1990s, the Quality of Service (QoS) concern has influenced the PBM. Afterward, the notion of policy-based network management (PBNM) was introduced. PBNM promises organizations the ability of controlling the QoS through watching and governing networked applications and users. Moreover, PBNM promises the organizations' administration the ability to control policies through instructions translated into specific management configurations, to bypass the traditional network operations (Jude, 2001).

Nowadays, the PBM approach arrives at a very generous level of *extensions*. There is not only PBNM, but the policy-based approach is applied to many domains, such as policy-based energy management (PBEM), policy-based security management (PBSM), policy-based QoS management, etc. However, the core feature that all policy-based solutions have kept is the policy-based architecture that proposes two main points: the decision-making and the enforcement of this decision. Moreover, to adapt to the heterogeneity of the environments, especially the distributed one, PBM proposes communications between one-to-one points and many-to-many points.

### PBM Architecture:

Most security or management approaches use a particular framework (model, language, and architecture) perfectly adapted to them and generally inspired by the PBM architecture. The Network Working Group of the IETF proposed this widespread, standard and well-known architecture (Yavatkar, 2000). PBM architecture consists of several elements: a policy repository (PR) that stores policies, a policy decision point (PDP) that allows evaluation of the relevant policies (set of rules) and shares a decision based on their evaluation, and a policy enforcement point (PEP) responsible for enforcing the decisions taken by the PDP (Figure 3.19). The architecture assumes a *one-to-one* relationship where there is a unique point for subjects to ask for access requests and a unique point to make decisions based on a unique security policy. However, this simple, centralized architecture does not fit the complexity of the new advances at the environment level (e.g., in large-scale systems where distribution is a priority: virtualization, mobility, etc.).

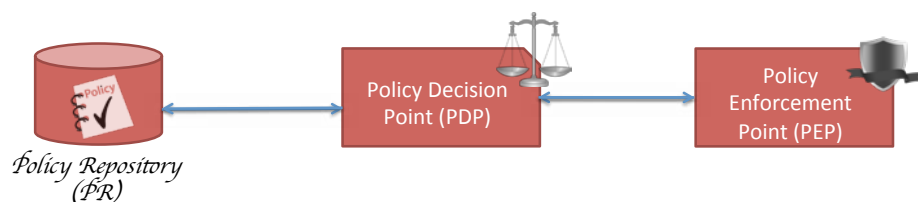


Figure 3.19: Policy-based management architecture (RFC 2753).

The *one-to-one* architecture starts by the policy officer writing the security policy based on requirements. The security policy will be stored in the PR in order to be restored whenever the PDP wants to use it. The PDP makes management decisions. The PDP agent is entirely independent from the environment it manages. The PEP enforces the PDP's decisions. The main role of this agent is to be the interface between the managed objects (resources) and the PDP. The PEP could be seen as a *safeguard* for resources, as it protects them by enforcing only authorized subjects or ensures their availability by enforcing suitable configurations. The communication between the PDP and the PEP is defined based on the usage of the PBM architecture. If the purpose of using the architecture is oriented for authorizations in management or access control, then the communication takes the handshake form in a synchronized request/response manner, known as the *outsourcing* communication. If the purpose of using the architecture is oriented for obligations in management or access control, then the communication usually takes the provisioning form in an asynchronized manner, where the PDP pushes configurations in one direction to PEPs, known as the *provisioning* communication.

In distributed networks and systems, PBM has emerged as a centralized management solution (Mohaban et al., 2004). PBM aims at providing systems and networks with a centralized architecture that facilitates the administration task. The policy within the PBM architecture is either translated into decisions (i.e., in the case of access control) or pushed as configurations (i.e., in the case of management, such as other policies, rules, commands, etc.), but not both. These communication ways are implemented and enforced separately across the networks or the systems management.

In literature, by following one, and only one, communication way per usage for either the management or the access control purpose, the PBM needs to define more an architecture that adapts to distributed-environments. Therefore, in terms of security management, there are three ways to establish the relationship between the PEP and the PDP based on the written security policy and the chosen communication way (i.e., outsourcing or provisioning): one-to-many (PDP-PEPs), many-to-one (PDPs-PEP), and many-to-many (PDPs-PEPs).

Nevertheless, it is important before going ahead with the architectures to explain the deployment of PDPs and PEPs. PDPs and PEPs can be separated as shown in Figure 3.20, where each point is completely independent with no overlapping functionalities. They can also be partially separated to share some of each other's features, where the PDP may contain a local PEP (LPEP) in order to communicate with other PDPs, or the PEP could contain a local PDP (LPDP) in order to obtain decisions and configurations (e.g., it could be a policy information base (PIB) when using the SNMP protocol). Finally, they could be co-located on the same machine and totally integrated in terms of functionalities and communications (Follows & Straeten, 1999).

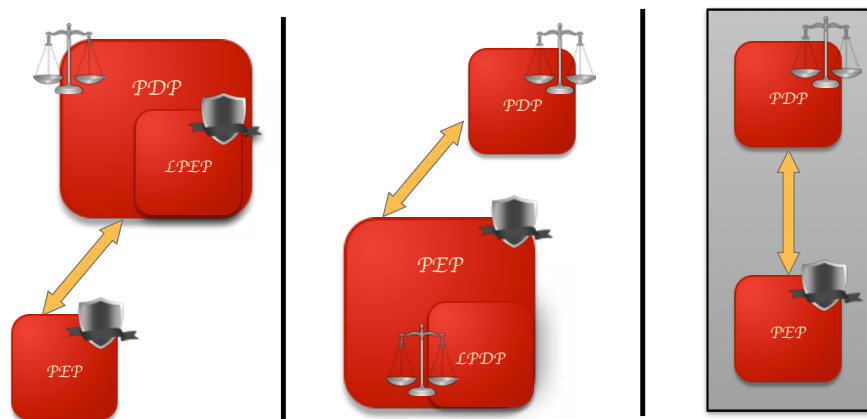
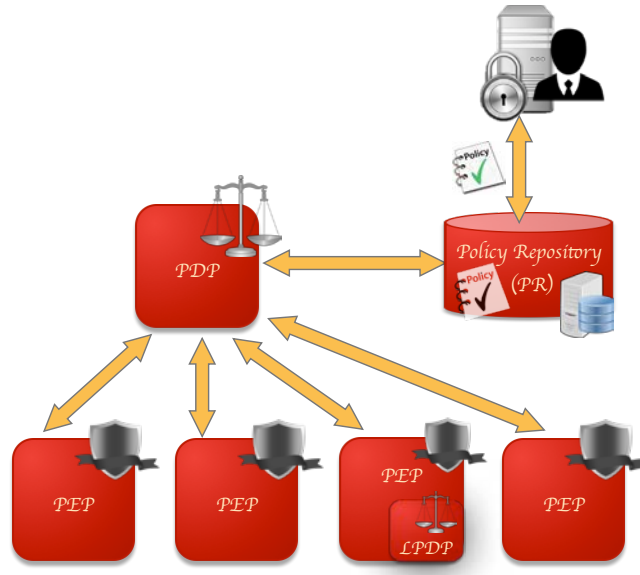


Figure 3.20: Relationships of PEP-PDP/LPEP, PDP-PEP/LPDP and PDP/PEP.

### One-to-Many:

The *one-to-many* architecture allows the usage of a unique and centralized PDP to make management decisions for many distributed PEPs. This kind of paradigm is useful whenever the administration that provides the management decisions is centralized, but the environment is heterogeneous and distributed.



**Figure 3.21: One-to-many relationship using the PBM architecture.**

In the outsourcing communication mode, the PDP agent is responsible for responding to the access requests, parallel and synchronized, coming from the different PEPs. The PEPs then enforce the decisions and the related obligations, if any. In the provisioning mode, the PDP responds to the incoming observed events and sends configurations to the PEPs based on the written security policy. This architecture has a widespread usage in terms of distributed network management using a centralized approach, such as in a hospital or clinic, but not in widely distributed systems like smart grids and healthcare organizations.

## **II.C. PBM Communication Paradigms**

Communications between components of the different presented security and management architectures are ensured based on the chosen paradigm: outsourcing or provisioning. The implementation of these communications with respect to the paradigm is ensured through communication protocols (e.g., COPS, RADIUS, SAML, AAA Diameter, XACML, SNMP, CIM-XML, SSH, etc.). In security management, the communications are what inform the policy about what is happening inside the managed environments.

Eventually, PBM supports behavior changes through providing adaptability. The adaptability is ensured by changing the security policy without redeveloping or interrupting the system (Sloman & Lupu, 2002). To achieve this aim, PBM offers the above architecture that adapts behavior changes to meet the requirements. Originally, the PBM seeks adaptability to resolve the heterogeneity of the networks and systems environments. However, the PBM solutions and choices are becoming more diverse, and with the heterogeneity in expression, the architectures are even more heterogeneous.

PBM communication solutions, in providing adaptability for the environment changes, are categorized as *outsourcing* and *provisioning* (see Figure 3.22). These two communication paradigms give the choice to administrate (supervise and govern) the environments by either the *management of configuration* or by the *controlling the access and usage*. This is another sort of heterogeneity in PBM by providing two different solutions, and this provides different architectures for each relationship explained. Moreover, this creates more distance between the management and the security of systems and networks.

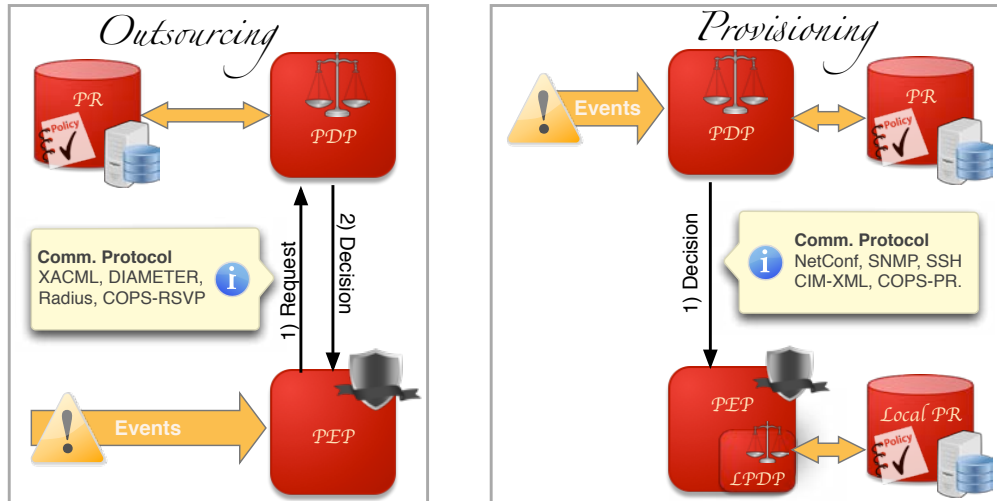


Figure 3.22: The PBM communication paradigms of outsourcing and provisioning.

Both outsourcing and provisioning paradigms were defined and standardized by the Network Working Group through the Common Open Policy Service (COPS) (Durham et al., 2000). The COPS stack is designed to meet all of the PBM architecture requirements. The COPS protocol cannot be directly used to distribute policies. Therefore, it represents the foundation for extending protocols that are specializations in performing a specific task (e.g., COPS-PR uses provisioning for equipment configuration, COPS-RSVP uses outsourcing for access control requests/responses, etc.). (Barrère, Benzekri, Grasset, Laborde, & Raynaud, 2001; Nataf, Festor, & Doyen, 2003)

Other adoption of the IETF *outsourcing* paradigm is proposed through the protocols RADIUS (RFC 2865), TACACS+ (RFC 1492) and Diameter (RFC 3588). RADIUS uses UDP (port 1842) since it is a simple “Request-Reply” protocol (Accept/Request). TACACS+ was proposed by Cisco as a terminal access controller for access control systems. Finally, Diameter is a successor to RADIUS that should fix some of its shortcomings. Diameter uses reliable transport connections (i.e., TCP or SCTP (Stream Control Transmission Protocol)). Both RADIUS and TACACS+ are access control protocols, but evidentially do not define the policy. These protocols merely provide a means to communicate such information between a client and an authentication server. The policy is implemented externally using an authorization server (e.g., AAA<sup>11</sup> servers in the upcoming section).

The *outsourcing* addresses the kind of events at the PEP that requires an instantaneous policy decision. In the outsourcing scenario, the PEP delegates the responsibility to an external policy server (PDP) to make decisions on its behalf. For example, in COPS Usage for RSVP (RFC 2749, 2000) when an RSVP reservation message arrives, the PEP must decide whether to admit or reject the request. It can outsource this decision by sending a specific query to its PDP, waiting for its decision before admitting the outstanding reservation (RFC 3084, 2001).

<sup>11</sup> Authentication, Authorization, and Accounting (AAA)

As explained within the relationships, access control uses both paradigms. In the outsourcing, the PDP receives access requests from subjects through the PEP passing by standardized protocols such as COPS-RSVP, SAML, Radius, Diameter, or XACML (Rensing, Karsten, & Stiller, 2001). The access requests are analyzed according to the written security policy. The PDP returns its decision using standard protocols as well. The decision holds the possibility of three types of values: “allow, permit, etc.”, “deny, prohibit, etc.”, or “indeterminate, unknown, etc.” To complete the PDP’s decision, an obligation section could be sent with the decision to be enforced by the PEP (e.g., sending an email to the administrator after having enforced the authorization decision).

On the other hand, the IETF Architecture at the PDP level supports the “provisioning” paradigm (i.e., using the COPS protocol for support of policy provisioning (COPS-PR)). An extraction from the COPS Usage for Policy Provisioning document could best describe the adaption of IETF architecture to the Provisioning mode:

“The COPS Configuration model (herein described as the Provisioning model) makes no assumptions of such a direct one-to-one correlation between PEP events and PDP decisions. The PDP may proactively provision the PEP reacting to external events (such as user inputs), PEP events, and any combination thereof (N-to-M correlation). Provisioning may be performed in bulk (e.g., entire router QoS configuration) or in portions (e.g., updating a DiffServ marking filter).” (RFC 3084, 2001)

When using the provisioning paradigm, the PDP receives events and starts reacting to them based on the written security policy. The reaction represents decisions provisioned to the PEP in order to be enforced (i.e., passing by standard protocols such as COPS-PR, SNMP, NetConf, CIM-XML, SSH, etc.). The decision could be a direct order to apply a configuration on resources (managed objects) or it could be itself a configuration for the PEP, a set of rules (policy). The PEP either applies the configurations to the environment or stores the policy to be used later on by the LPDP. The LPDP also receives events about the environment and thanks to the provisioned policy, the PEP may not need to ask the PDP for decisions, as the decisions could be already handled and therefore be locally provisioned by the LPDP to the PEP.

From the literature, security management obviously needs both paradigms to really meet the CIA criteria. Moreover, the management of security is a two-tasks mission: the management and the security. Therefore, unifying both paradigms provides a robust, flexible, and powerful solution that could adapt the system and network entities to the environment changes and subsequently meet the management and the security requirements.

## **Technical architectures of Outsourcing & Provisioning**

In Figure 3.23, the technical specifications of the COPS-RSVP (i.e., outsourcing) and the COPS-PR (Provisioning) are with decision-making conceptualization. Upon the attempt of using the network from the user, the outsourcing paradigm is a synchronized decision making. The PEP maintains a TCP connection (i.e., open and close) and works as a client for the PDP. The PDP receives the requests from the senders and respond with direct synchronized decisions. The objective of the outsourcing is controlling access to resources. Upon network notifications (i.e., including the attempt of using resources), the provisioning paradigm is not necessarily synchronized. However, the PEP maintains a TCP connection and works as a TCP client for the server installed inside the PDP. In this case, the TCP connection objective is to receive policy configuration from the PDP in order to be stored into the PIB (policy information base).



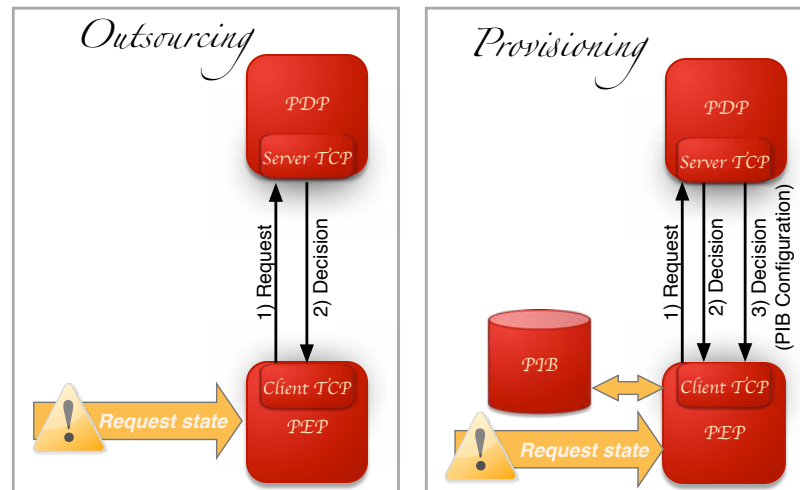


Figure 3.23: Outsourcing and provisioning technical architectures

As a technical comparison, both paradigms use TCP session not like the SNMP using UDP sessions. The difference between outsourcing and provisioning is the logic. The outsourcing is based on externalizing the decision-making to the PDP. The provisioning is based on the provision of what is needed for the PEP to apply the decisions. Technically, the outsourcing considers the PEP is the sender (i.e., informs the PDP about the access requests), but the provisioning considers the PEP as the receiver (i.e., takes policy configurations from the PDP). (RFC 2753; RFC 2749)

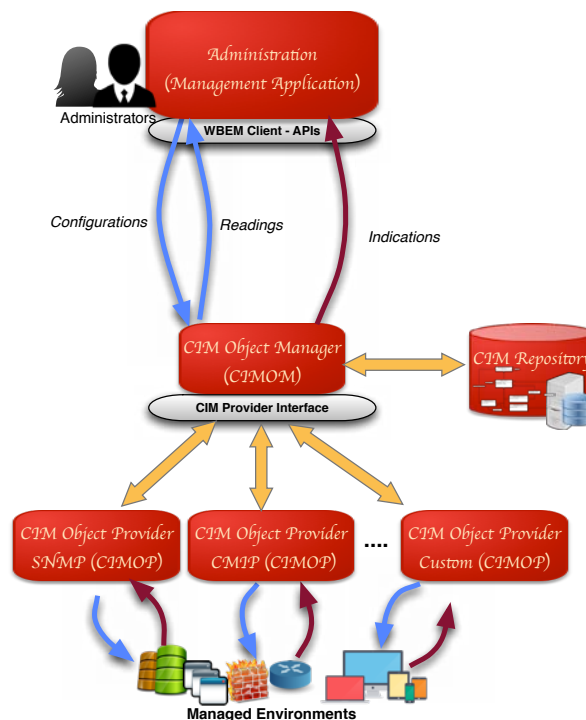
## II.D. DMTF WBEM Architecture

Communication systems including networks, middleware, services and complex systems represent a major terrain for the supervision in terms of configuration and protection. Integrated management aims at providing all activities that help optimizing the interoperability between heterogeneous management environments, namely using configuration and reconfiguration management. As the leader contributor to this domain offering standards and solutions, DMTF initiated the WBEM (Web-Based Enterprise Management) to unify the management of IT environments (see Figure 3.24). The aim of WBEM is to integrate existing management solutions through a uniform modeling of management information (CIM) to help performing advanced management (event correlation, proactive management, etc.). However, the standalone WBEM is not enough to provide governance and supervision. Therefore, the WBEM architecture in Figure 3.24 (the right side) supports multitude technologies for that complementing its environment. Integrating SNMP (Simple Network Management Protocol) standardized by IETF helps the supervision task.

WBEM architecture is mainly composed of two sides the WBEM server(s) and the WBEM client(s). At the server side, the architecture consists of two main components the CIMOM and the CIMOP. The CIMOM is the CIM *object manager* that controls the interactions between the CIM repository (i.e., where the management information are stored) and the client side or the provider side. The CIMOP is the CIM *object provider* that represents the interface between the CIMOM and the available providers that it can use. The *CIM providers* are software modules that are written using a high-level language (e.g., Java, C++, Python and C#). These modules are DSLs<sup>12</sup> written for specific objective that is to represent or encapsulate a specific technology (e.g., SNMP). The WBEM client(s) side is where the administrators supervise and govern the managed environment. There are three types

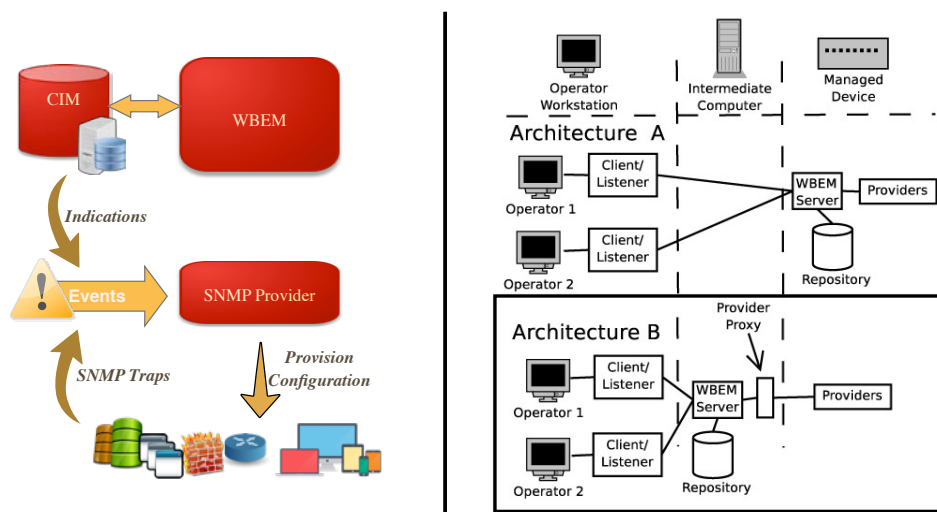
<sup>12</sup> Domain Specific Languages (DSLs)

of transactions that can be exchanged. The first two are setting and getting management information. The third is subscribing to notifications (i.e., CIM indications) that are triggered upon satisfying triggering conditions.



**Figure 3.24: WBEM architecture**

At the left side of the Figure 3.25, the integration of SNMP (or any other protocol) with WBEM is supported through software interfaces known as providers. The configuration management of communication systems is then possible to ensure using an SNMP provider. The SNMP provider listens to the environment notifications through so known SNMP traps. Then the mapping with the WBEM is through using a CIM class named *indications* in order to update the new monitored value of the environment (e.g., the connection number of machines to a networked switch). Moreover, the SNMP can get notifications (indications) to change values of parameters (i.e., provision configurations) into the environment (e.g., change the default configuration of a networked printer to print tasks on two sides).



**Figure 3.25: Integrated management using WBEM and SNMP for configurations provisioning**



## II.E. Ponder Architecture

Ponder (Damianou, Dulay, Lupu, & Sloman, 2001) is a security and management solution for distributed systems. Imperial College London implemented the Ponder architecture in a centralized manner. Its architecture is based on designing a policy repository (LDAP directory) that stores all management and security policies. Via a management console, the administrators create, store and modify their policies (in the form of Java RMI policy implementation). Ponder management strategy is to distribute multiple *Ponder Management Components* (PMCs) throughout the managed networks and systems. In the literature, these PMCs entities are no difference from PDPs/PEPs according to the IETF definition (i.e., PMCs are qualified to receive policies and to enforce them) (see Figure 3.26).

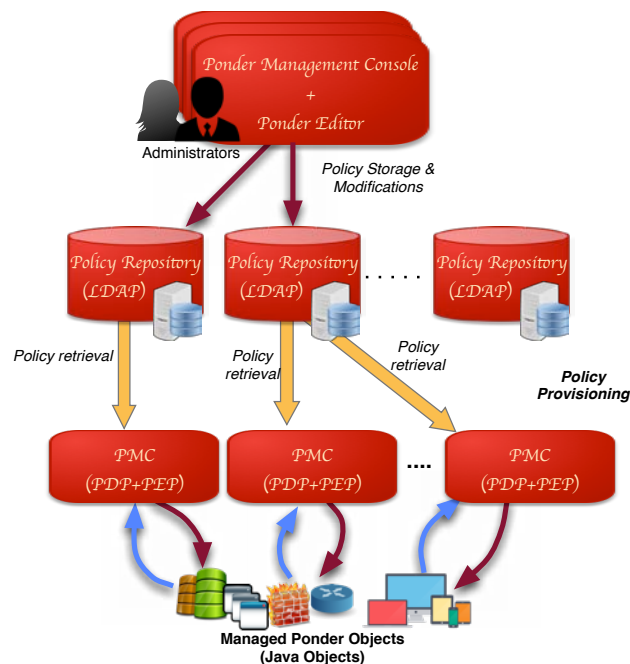


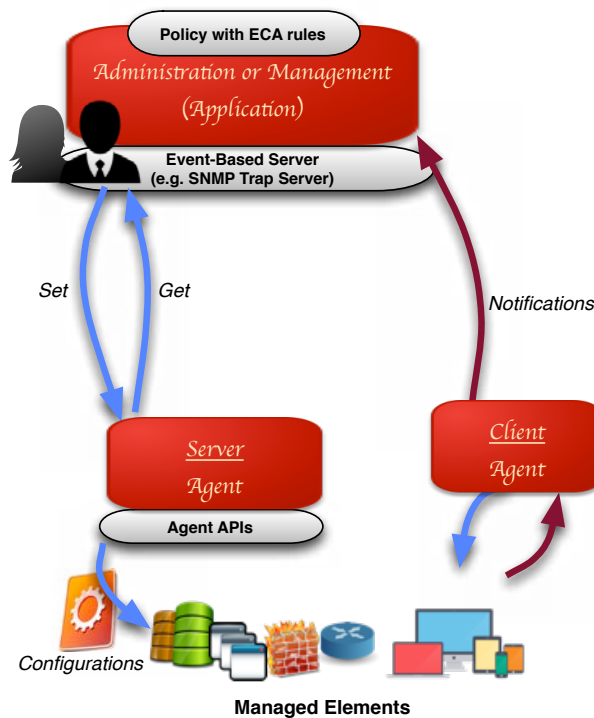
Figure 3.26: Ponder architecture

## II.F. Concluding management architectures

The reason of mentioning the SNMP/WBEM integrated management in this work refers to the focus of this research domain on improving the configuration and reconfiguration management aiming at improving the adaptation strategies. Therefore, the objective of this thesis to contribute with a security management solution is rather valorized when it respects the provisioning paradigm applied in this domain. As a result, it is important to define an architecture that combines the provisioning paradigm (for configuration management architectures), but stays correct in terms of offering the authorization service with respect to the security requirements (CIA).

In Figure 3.27, the provided illustration is a general view of all management architectures describing the common components and the different communication paradigms. Every management architecture has a manager or an administrator side that sends the management orders (i.e., in form of configurations). However, the supervision task, which involves knowing the reality inside the

environments in real-time, needs two operations<sup>13</sup>, to help monitoring the environment and to watch its behavior changes: set and get. Both operations participate in performing common management strategies. The get operation either called manually, timed with a programed interval system (e.g., polling), or automatically in form of triggers (e.g., SNMP traps, CIM indication, low-bandwidth notifications). The other side is responsible of governing the managed environment through applying the control orders requested by the administrators. The application is either by responding to the administrator subscription to notifications (e.g., by sending a notification of low bandwidth from the router agent) or by setting the configuration sent by the administrator (e.g., changing the assigned route for some IP addresses inside the routing-table at a specific router).



**Figure 3.27: General management architecture**

<sup>13</sup> The control operations did not change, however, we presented that NETCONF has some new technologies that help managing the overlapped access to the management information at the same time using the operations *lock/unlock*.

### III. Architectures with security objectives

The literature defines the security architectures based on the security objectives that an organization needs to obtain. For instance, organizations that are interested in implementing an encryption solution and maintain the confidentiality of their assets go for PKIs solutions (i.e., including its architectures). However, the security community worked on defining an access control architecture that can generally represent the triple security: Authentication, Authorization, and Accounting (AAA). This concept of AAA aims at providing services through an AAA server that separate the functionalities of authentication and authorization services at one side and ensure the management of applications at another.

The literature holds several adoptions in terms of the technical implementation to the AAA architecture. In 1991, the Livingston Enterprises presented RADIUS as the Remote Authentication Dial In User Service networking protocol. It aims at centralizing the management of Authentication, Authorization, and Accounting (AAA) for the users who connect and use a network service. It has been standardized later on by the IETF (Vollbrecht, 2006). RADIUS provides several implementations for the AAA architecture (Jacquenet, Bourdon, Boucadair, 2008, Chapter 11) and several trust relationships between services providers and consumers (Hassell, 2002, Chapter 1, Section 2). A second implementation for the AAA architecture is Privileges Management Infrastructure (PMIs). Considering their privileges, PMIs offer several ways to manage users' access to available resources (i.e., privileges are attributes or properties assigned by a credential or trusted authority). The privileges are retrieved whenever the needs are identified into the access control policy.

Finally, OASIS standardizes the XACML architecture with its complete solution (i.e., language, architecture and communication protocol). The XACML architecture meets the PBM architecture, respects the AAA standard architecture and can accept both implementations of a PMI and RADIUS as well. Moreover, the support of an issuer in XACML permits the integration of PKIs approach in its architecture to manage trustworthiness. XACML today is considered as one of the soundest security solution<sup>14</sup>.

#### III.A. AAA Architecture

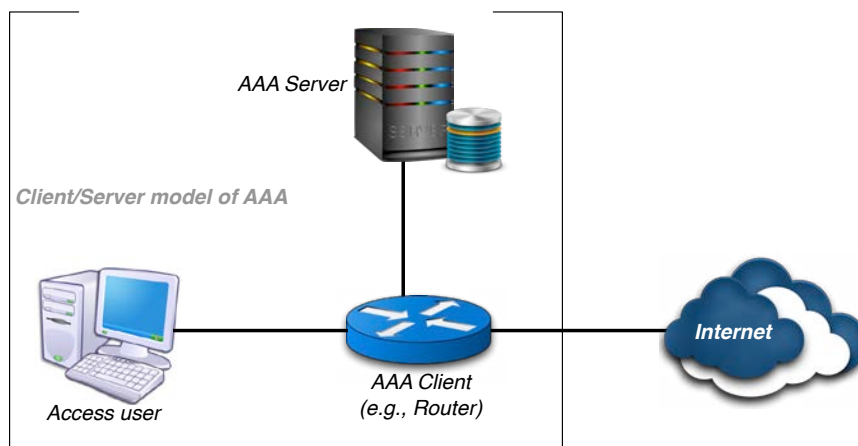
AAA proposes a standard generic architecture for management systems of services security. According to RFC 2904 of the IETF, the basic entities involved in authorization are (see Figure 3.28): a *user* requesting a service, the *main organization* that establishes contract with users (i.e., this involves the necessity of checking in an active or a passive form the user's authorization to trigger the execution of the service), the service provider's *AAA server* that allows access to the service based on the signed contract with the main organization of the user, and the *service equipment* dedicated to the services provisioning in response to service requests.

The AAA authorization approach imposes the necessity of undertaking two actions: 1) an authorization decision after evaluating the access control policy and 2) the provided decision should be enforced. Respectively, two separated entities perform these two functions the PDP (Policy Decision Point) and the PEP (Policy Enforcement Point). On the first hand, the PDP is a logical entity that takes authorization decisions by considering the following information (RFC 2906): *the requested resource*, *the required action* (view, modify, use, etc.), *the subject* requesting the resource, and *the policy* that

---

<sup>14</sup> <https://cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf>

manages access to the resource. On a second hand, the PEP is a logical entity that applies the authorization decision made by the PDP (i.e., this operation known as enforcement). This PEP is the guardian of the resource(s).

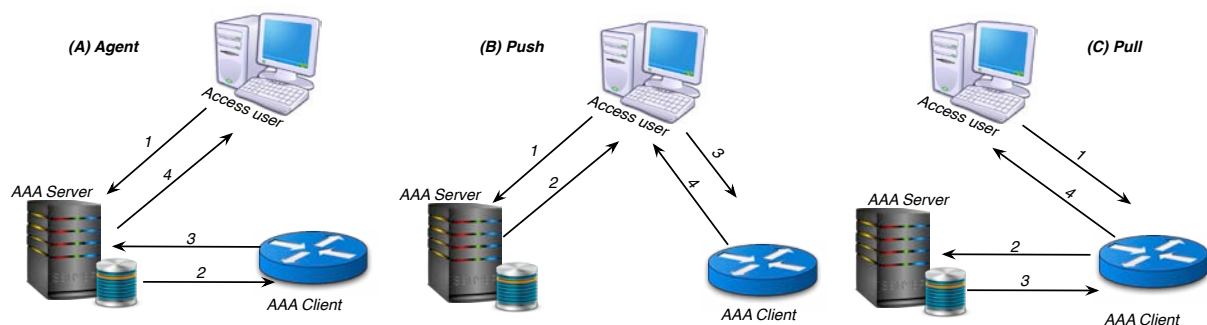


**Figure 3.28: AAA generic architecture (RFC 2903)**

The interactions between a PDP and a PEP can follow one of three models: Agent, Push or Pull. The agent model allows the user (requesting access) to submit her query to a third party (i.e., the AAA authorization server that is composed of the PBM elements: PDP / PEP). This AAA agent acts between the user and the equipment providing the service by applying the policy on the request. Upon the policy evaluation, the server allows or not the user to access the resource (i.e., the user receives notification of the denial as a message). In case of permission, the resource returns the query result to the PEP, which transfers it back to user (see Figure (A), 3.29).

In the push model, the user addresses her desire of using the service provided and managed directly by the service provider (i.e., the resource/PDP). The latter's role as an authorization server manages access to the resource by defining the authorization information (i.e., the right to use the service and start accessing the resource or not). In case of acceptance, the user receives a ticket to present it with each access request to the resource. The PEP by her turn is located at the resource level and gives the right of access to the user if the ticket is valid (see Figure (B) 3.29).

Finally, the pull model assigns the responsibility for managing the authorization information exclusively to the PDP. Upon the access request, the resource/PEP inquires the PDP for the authorization information in a solicited (active) way. The PDP grants (or not) the right of access to the user, based on the policy (see Figure (C) 3.29).



**Figure 3.29: Three AAA PDP/PEP interaction models: (A) Agent, (B) Push, (C) Pull (RFC 2904).**

## III.B. OASIS XACML Architecture

XACML (eXtensible Access Control Markup Language) is a security management solution composed of a standard language, architecture, and implementation. The XACML security management solution was created by OASIS in 2003 (Godik, 2003). XACML aims at standardizing the access control management through using the PBM architecture, XML representation and enhancing the interoperability between heterogeneous systems, as it presents a standard solution.

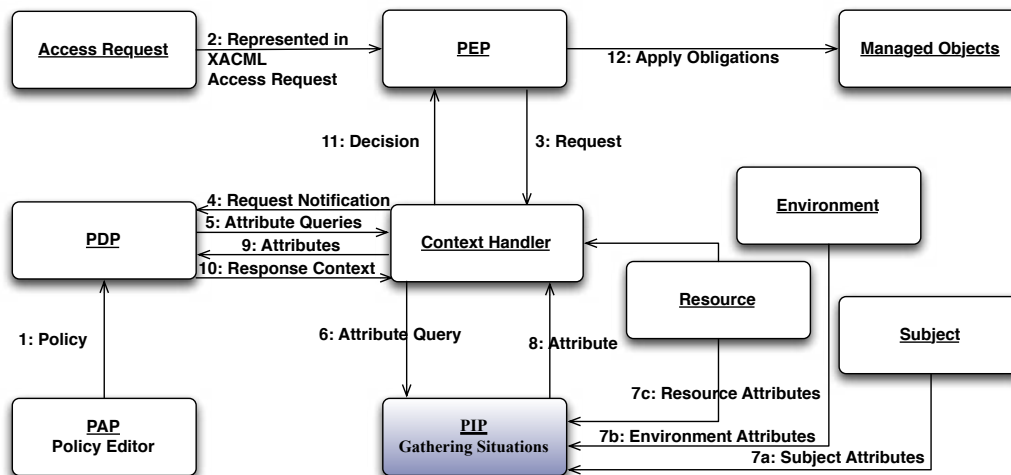
On the architectural side, XACML is based on the standard association between the PEP, the PDP, and the PR. In Figure 3.30, the drawn boxes are the main components that identify the XACML architecture:

- ❖ The ***Context Handler*** is responsible for ensuring the communication (and translation if necessary) between the PEP and other boxes or entities to retrieve or complete the required data for the evaluation process of the policy.
- ❖ The ***Policy Administration Point*** (PAP) creates and stores the security policies in the PR. Mostly, it is a user-friendly interface, or editor, which provides the policy writer (manager or administrator) with possible facilities to write his or her security policy. The policy is expressed in a semi-natural language close to the human English language.
- ❖ The ***Policy Information Point*** (PIP) is responsible for storing or retrieving complementary information about attributes' values. This information will be needed for the evaluation of the policy by the PDP,
- ❖ The ***Policy Decision Point*** (PDP) makes the management decisions. This management agent is independent from the application it manages. It receives requests using a standardized protocol such as COPS, SAML, or XACML. These requests are analyzed according to a policy; the PDP may ask for more information to make its decision. Finally, the PDP returns its decision, which can be “*accept*,” “*refuse*,” or “*indeterminate*.” An obligation section can complete this decision (such as “*send an email to the administrator after having enforced the authorization decision*”).
- ❖ The ***Policy Enforcement Point*** (PEP) enforces the PDP's decisions. The main role of this agent is to be the interface between the management application and the PDP. Hence, it translates the requests that are expressed in the application-specific language into the standardized protocol language understood by the PDP. The PEP can also get additional information that will help the PDP to make its decision. Finally, it translates the PDP's decisions into the application-specific language.

The workflow of the XACML architecture contains the following steps:

1. PAPs write policies and policy sets and make them available to the PDP. These policies or policy sets represent the complete policy for a specified target.
2. The access requester sends a request for access to the PEP.
3. The PEP sends the request for access to the context handler in its native request format, optionally including attributes of the subjects, resource, action, and environment.
4. The context handler constructs a standard XACML request context and sends it to the PDP.
5. The PDP can request additional subject, resource, action, and environment attributes from the context handler, if needed.
6. The context handler requests the attributes from a PIP.
7. The PIP obtains the requested attributes.

8. The PIP returns the requested attributes to the context handler.
9. The context handler sends the requested attributes. The PDP evaluates the policy.
10. The PDP returns the standard XACML response context (including the authorization decision) to the context handler.
11. The context handler translates the response context to the native response format of the PEP. The context handler returns the response to the PEP that enforces the authorization decision.
12. If the decision includes obligations, the PEP fulfills them.
13. Finally, the PEP enforces the authorization decision.



**Figure 3.30: XACML PBM architecture.**

The defined steps are the standard flow of actions that describe the XACML architecture. XACML is suitable for distributed environments where PEPs and PDPs are distributed in heterogeneous infrastructures. By separating the access policy of the protected applications and by providing standard authorizations expressions, XACML allows heterogeneous systems to share security policies. Thus, system administrators are no longer needed to write their policies using different languages. In addition, several research works presented that try to modularize the implementations of PDP and PEP in order to improve the reusability and the adding of new features on the fly (i.e., at the runtime) (Laborde, Cheaito, Barrère, & Benzekri, 2009; Cheaito, Laborde, Barrère, & Benzekri, 2011, Laborde, Kamel, Barrère, & Benzekri, 2008).

Moreover, one of the XACML strengths, in terms of both language and architecture, is the acceptability of extensions. The language is open to extensions and the architecture can accept new components, and while it supports the standards, XACML can accept many standard protocols. This makes the number one choice for the solution we seek the XACML architecture and language, as it supports ABAC. However, this choice is not sufficient alone because the standard XACML does not unify the approaches and architectures presented previously, and therefore it misses the unification of them and the adaptability, whenever the environment changes the behavior.

With comparison with the COPS management paradigms, the XACML is considerably similar to the outsourcing communication manner. However, the XACML does not obligate the PEP to maintain the TCP connection and therefore can manipulate several PEPs and PDPs. However, the XACML architecture does not support a similar technology to the COPS-PR. The provisioning paradigm helps the configuration of networks and this what is missing in XACML solutions.

### III.C. Concluding security architectures

The section mentioned the AAA and XACML solutions that focus on improving the access control management, namely authorization, by proposing abstract or implemented architectures that helps managing securely networks and systems. With the thesis objective of introducing a security management solution, it is remarkable that security architectures do not valorize the management objectives. As a result, it is important to stay correct in terms of offering authorization service with respect to the security requirements (CIA), but a security management architecture should combine the management features (for configuration management essentially) beside the security features.

In Figure 3.31, the provided illustration is a general view of all security architectures describing the common components and the different communication paradigms. For the security architectures, every one has an administrator (i.e., security officer) who responds to the security queries (i.e., in form of access requests). However, the authorization task, which involves managing who can access what, when and under what circumstances, needs two operations to help controlling the access in the managed environment: authorization decision making, and enforcing authorizations and obligations. Both operations participate in performing common access controls approaches.

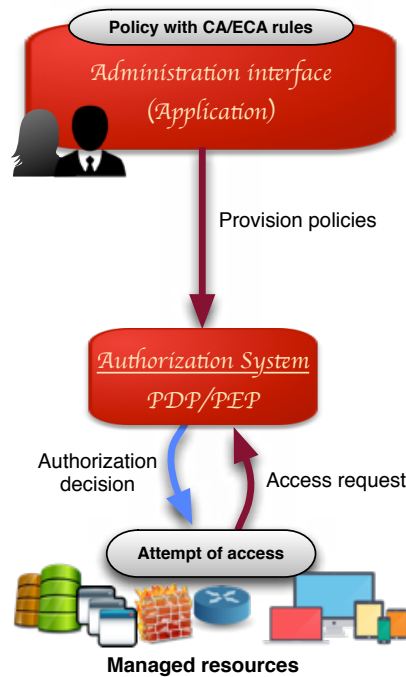


Figure 3.31: Literature general view of access control architectures

The authorization decision-making operation (i.e., the responsibility of PDPs) is about evaluating the security policy in order to check if the users who request access have the right to use the requested resource(s). Once the decision is made, the authorization is applied through the enforcement operation that should ensure the users getting or not the access to resources. The used paradigm for the exchange between the access requestor and the access provider is usually *outsourcing*. However, the authorization system logic of controlling the access is common, but the composition and the representation of the system are different from an architecture to another (e.g., Ponder calls PMCs instead of PDPs and PEPs).




## Chapter's Conclusion

---

*To wrap-up this Chapter, its main objective is to highlight the deep-seated differences between the current management architectures logic and the current security architectures logic. Essentially, the introduction of the PBM architecture opened thoughts to reach a common architecture that can fit both worlds' requirements. However, its abstraction and missing of implementation is not actually enough. Notably, the today's PBM architecture treats the communication paradigms separately, even though it supports both representations. The eager need for a unified architecture that could meet management and security requirements and participate in providing a security management solution is overwhelming.*

---

# PART III



# CONTRIBUTION

*The third part presents our contribution within the PREDYKOT project. Our vision is detailed in depth through three chapters dedicated to expression, architecture and concrete scenarios as a proof of concept.*

---

# ***Introduction to PART III***

---

We identified two crucial phases where studies have been heterogeneous: the expression and the architecture. Policy expression and applying it to architecture is the enforcement of this policy. Obviously, the policy seems to have a sort of life cycle. The management of the policy's life cycle was clearly stated through the European project called PREDYKOT.

On October 2011, the fifth call from the ITEA program kicked off PRDYKOT. ITEA is the EUREKA<sup>15</sup> cluster program supporting innovative, industry-driven, pre-competitive R&D projects in the area of software-intensive systems and services (SiSS). The R&D PREDYKOT project aimed at ensuring software modules that deliver policy refined dynamically and kept on the track. PREDYKOT addressed the *ecosystems* to shift their focus on the basic operational improvements of security policy management to the critical intelligence for business process improvement. The *intelligence* considered by this project is related to necessary mechanisms that ensure that the effectiveness of the security policy remains at any time. Moreover, the project proposes to undertake the contextual information, during the decision-making, to dynamically refine the security policy, (that is, by applying changes or modifications on the policy). The requirements that concerned this project were mainly the governance, the risk management, and the compliance.

## ***What is an Eco-System?***

THE NOTION OF ECOSYSTEMS ORIGINATES FROM ECOLOGY TO DESCRIBE A UNITY OF ACTORS LIVING INSIDE. THE ECOSYSTEM IS A SET OF SOLUTIONS THAT ENABLE, SUPPORT, AND AUTOMATE THE ACTIVITIES AND TRANSACTIONS BY AND BETWEEN THE ACTORS OF THE ORGANIZATIONS (BOSCH, 2009). SOFTWARE ECOSYSTEM IS "A SET OF ACTORS FUNCTIONING AS A UNIT AND INTERACTING WITH A SHARED MARKET FOR SOFTWARE AND SERVICES, TOGETHER WITH THE RELATIONSHIPS AMONG THEM. THESE RELATIONSHIPS ARE FREQUENTLY UNDERPINNED BY A COMMON TECHNOLOGICAL PLATFORM OR MARKET AND OPERATE THROUGH THE EXCHANGE OF INFORMATION, RESOURCES AND ARTIFACTS" (BERK, JANSEN, & LUINENBURG, 2010).

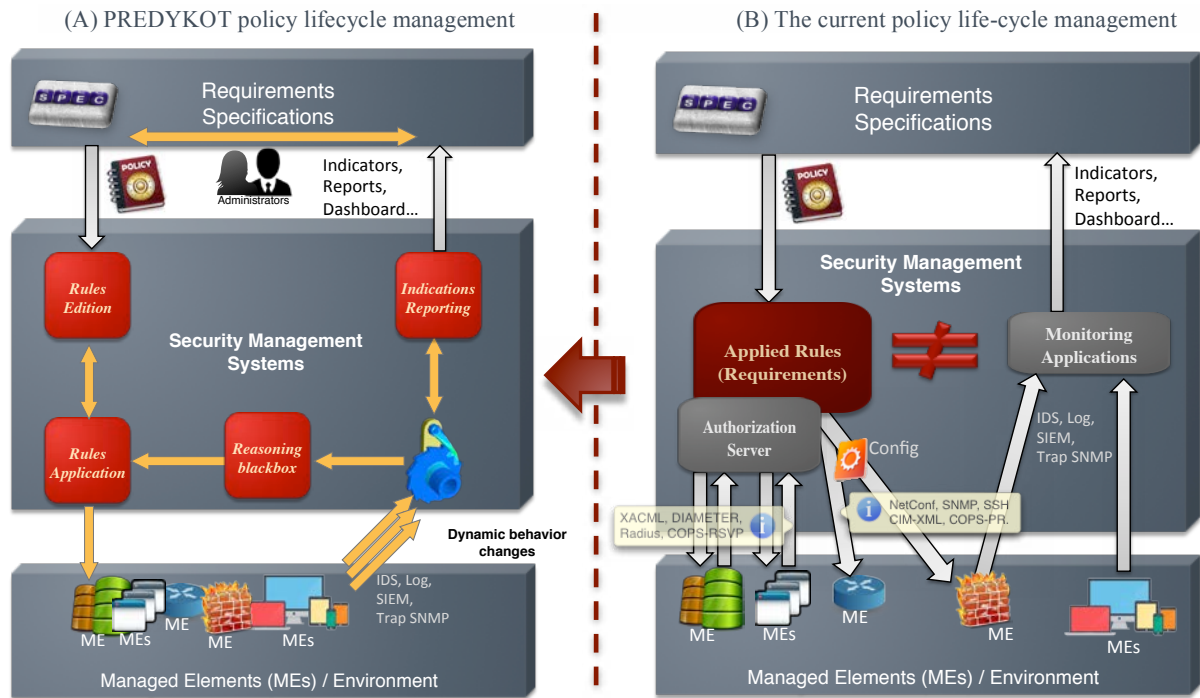
The PREDYKOT project developed a new method to present the PBM architecture. The project analyzed the market and the standards (from Gartner Reports: 2009, 2010, 2011 and 2012), and compared them with current industrial and research requirements. The outcome of this study was an architecture that creates a security loop by connecting the environment behavior changes automatically with the policy refinement process (that is, expressing the security policy and continually improving it to stay on track with the business requirements). This created a sort of feedback based on which the refinement process evaluates the current security policies. The feedback is an output of a reasoning phase that consists of a specific language and intelligence engine. The implementation of the PREDYKOT architecture and expression was through a methodology known as policy management or policy life-cycle management. This approach is based on dynamically refining, modifying, or completely changing the security policy (policies) whenever the requirements are not met. Therefore, this approach keeps verifying, or polling, the security policy to check its validity.

Today's policy lifecycle management forms a cascade workflow of sequential management tasks. However, current policy lifecycle management misses the link between the monitoring side and the security governance side (cf. Figure III.32, A). Without this link, there is no guarantee that changes

---

<sup>15</sup> The intergovernmental organization for pan-European research and development funding and coordination

at the environment are *always* meeting the defined security requirements. PREDYKOT proposes to automatically verify that whatever dynamicity produced at the environment, the policy stays effective and on the track (i.e., still meets the requirements). The automation link of guarantying policy effectiveness is possible at two points: the policy expression and the enforcement architecture. The reasoning box ensures the automation of the link. It keeps the expressed policy updated with managed elements behavioral changes.



**Figure III.32: Comparing PREDYKOT with the current policy life-cycle management**

According to PREDYKOT results, there are four principle elements to consider in order for an effective management of the policy lifecycle: the specified requirements, the security management policy, the reasoning feedback and the managed elements (cf. Figure III.33). Through the project, the consideration of the reasoning feedback inside the specified policy has created two implementation visions. This division into two views is a result of the question “what is the impact of the reasoning feedback on the policy lifecycle?”

The first view goes with associating the feedback with the policy rules. Therefore, whenever the feedback changes, the rules of the policy must be updated. In addition to managing the policy, modifying the policy (e.g., operations like adding, removing, replacing, altering) imposes the management of its rules. Rule-based management in highly dynamic environments could create a complex dilemma in term of the engine performance and the policy integrity. However, the use of a heavy engine (e.g., usage of Drools) that could manage reactive rules upon intelligent calculation of feedbacks was one of the PREDYKOT implementations presented by two partners: situation calculus using a reasoning engine (Samarji, Cuppens, Cuppens-Boulahia, Kanoun, & Dubus, 2013) and reasoning language (Hu, Patkos, Chibani, & Amirat, 2012). Such engines may reduce the performance weight on a heavy process like the management of rules.

Sure, we believe in implementing a security management solution that keeps the policy effective and on track whenever the feedback changes, but without changing the policy. This means, that the policy should apply authorizations and configurations suitable for every feedback. Dynamic authorization is giving permissions and prohibitions to managed elements based on the provided feedback (e.g., only analyzers are allowed to modify the documents during analysis task). Dynamic

configuration is provisioning the suitable obligations to the provided feedback (e.g., keep track of modifications when errors are produced). Our contribution is an approach to express authorizations flexibly so that it can provide dynamic permissions and prohibitions. Moreover, our contribution is an adaptive architecture to apply dynamic and changeable configurations according to the feedback so that there is no need to stop the security management system.

Our implementation vision preferred to keep the policy stable during the automatic loop, which means limiting policy modifications upon change requests on requirements. The next thesis chapters are following this PREDYKOT implementation vision.

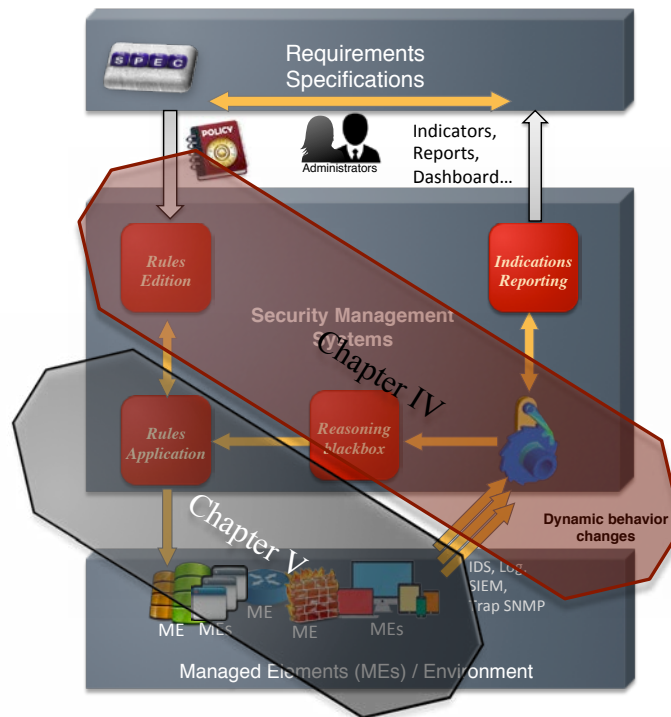
Figure III.33 gives a link to point our upcoming contribution within the PREDYKOT project. With the next two chapters, we are addressing our vision of an effective dynamic decision without modifying the policy. The first chapter addresses three points:

1. How to define the feedback in order to express dynamic security management policies?
2. How this feedback is defined, identified, and made available for consumption by the policy?
3. How to represent the dynamic security management policies using this feedback?

The second chapter is concerned with the adaptive enforcement of the decisions both authorizations and configurations. It presents the adaptive architecture through three points:

- How the feedback will impact dynamically the policy decisions?
- How the dynamic decisions will be implemented and enforced adaptably, without interrupting the security management system?
- What impact the enforced decisions would have back on the feedback?

At the end of this part, to proof our generic contribution, characteristics, we apply our PREDYKOT implementation to three different scenarios through expressing and implementing each of the scenarios policy.



**Figure III.33: Linking PREDYKOT architecture with next two chapters**

# Chapter IV

## Expression of A Dynamic Security Management Policy

---

*Chapter 4 aims at clarifying studies around expressing the dynamic behavior changes (the feedback) in term of two notions contexts and situations. Next, we present our choice of expressing the feedback based on the notion of situations. We engineer the policy to consider the situations feedback by proposing a methodology of expression, and going in depth on how to define and manage them. We present our choice of a flexible language that respects the attribute-based approach and expresses the dynamicity of the behavior. We present our choice of a flexible language that respects the attribute-based approach and expresses the dynamicity of the behavior. As a security management language, we need to orient XACML in order to be able to give adaptive decisions and configurations. Therefore, the last section presents the orientation mechanism and structure of XACML using situations.*

*In order to support such a methodology, we provide a tool for the situations calculus. A set of software modules implements the situations manager that can be considered in any deployment infrastructure.*

---

### Summary

#### **I. Expressing the Policy with Feedback**

- Terminologies: Situation, Context, Decision
- Methodology: SCD Policy Engineering
- Evaluating the Policy Expression using SCD

#### **II. Feedback Implementation**

- Situation Management
- Complex Event Processing
- Adaptable Situation Manager
  - 1) Situation Design & Identification
  - 2) Dynamic Situation Configuration
  - 3) Situation Preservation

#### **III. Representing the SCD Policy-Expression**

- Dynamic XACML Policy oriented by Situations

# Chapter 4: Expression of A Dynamic Security Management Policy

*“It is very unfair to judge any body’s conduct, without an intimate knowledge of their situation. Nobody, who has not been in the interior of a family, can say what difficulties of any individual of that family may be.”*

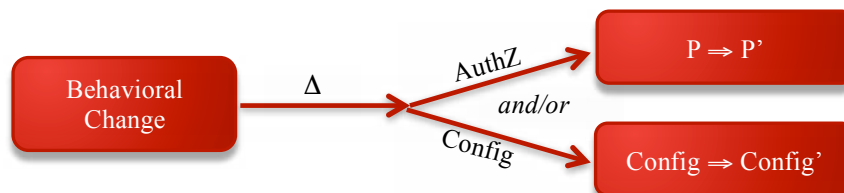
*Jane Austen*

*Columbia World of Quotations*

*Retrieved November 10, 2014, from Dictionary.com*

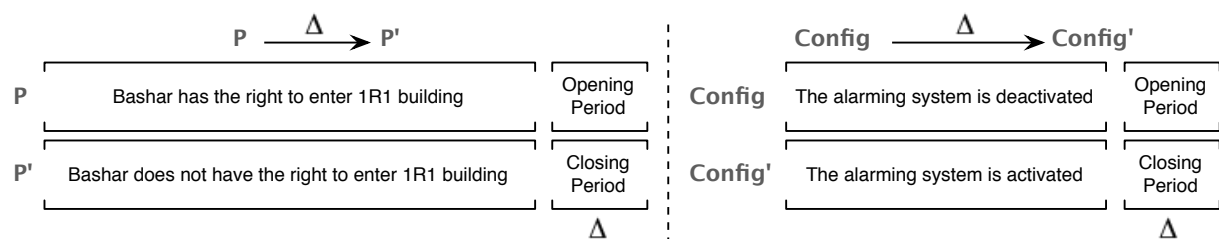
## Introduction:

Dynamic security management involves expressing the policy with understanding the dynamic behavior of managed elements and with accordance to the initially specified requirements. For instance, insecure states like systems under attacks become inevitable nowadays. Under attacks, the managed elements, and consequently the managed system itself, won't behave normally. Hence, each change at the environment causes a behavior change ( $\Delta$ ) of this environment's elements. In consequence, this change may impose an equivalent change on the security policy as well, concerning both authorizations and configurations. A behavioral change at the authorization (AuthZ) point imposes changing the current permission (P) with a new permission (P'). Likewise, a behavioral change at the configuration point imposes changing the current configuration (Config) with a new configuration (Config'). We could conceptualize this by the following:



To introduce this chapter, we demonstrate one simple example to use all along the underlying sections. Therefore, we specify its requirements. Then, we explain how to express these requirements with current policy expression approaches and what are their drawbacks. Finally, we conclude by presenting our motivation towards a contribution at the policy expression level.

Let's consider that at IRIT (Institute de Recherche en Informatique de Toulouse), we deployed a security alarm system. We want our dynamic security policy to manage Bashar's permissions and alarm systems' configurations with consideration to the closing/opening periods. Thus, we specify the following four simple security requirements:





To express the specified requirements in a security policy, current common approaches are either the CA (i.e., the common pseudo-code notation is “*If condition Then action*” (Strassner, P. 164, 2004)) or the ECA (i.e., the common pseudo-code notations are “*Upon/On event If condition Then/Then Do/Do action*”) (Hu1, Yeh, & Laun, 2009; Bonatti, 2010). As there are several ECA notations, throughout this thesis we will depict rules using the following pseudo-code notation: “On event **if condition then action**”.

On the first hand, the CA policies are dedicated for authorizations and are evaluated whenever an *access request* is received. On the other hand, the ECA policies are dedicated to ensuring configurations and are evaluated whenever a relevant event holds. However, authorizations have been managed using ECA by considering the event (E) as a constant type (i.e., E is always an *access request*).

We start by expressing the authorizations then we express the configurations. However, we need to define first what are the “*opening period*” and the “*closing period*”. At IRIT, 1R1 building is open to employees at the morning from 7:00 and until the 20:59 in the evening. Predictably, the building is closed outside this period (i.e., from 21:00 until 6:59).

We suppose that the access control event (E) is always: a person wants to enter. Hence, we express the authorization policies as follows:

- ☞ **Authorization Rule 1:**  
*If Name (Person) = Bashar  $\wedge$  Name (Action) = Enter  $\wedge$  Now (Time)  $\geq$  7:00  $\wedge$  Now (Time)  $\leq$  21:00  $\wedge$  Name (Resource) = Building 1R1*  
*Then allow the person to enter*
- ☞ **Authorization Rule 2:**  
*If Name (Person) = Bashar  $\wedge$  Name (Action) = Enter  $\wedge$  Now (Time)  $<$  7:00  $\vee$  Now (Time)  $>$  21:00  $\wedge$  Name (Resource) = Building 1R1*  
*Then prohibit the person to enter*

Now, the expression of the configuration policies could be as follows:

- ☞ **Configuration Rule 3:**  
*On evt1 = Time is 7:00*  
*Then deactivate the alarm system*
- ☞ **Configuration Rule 4:**  
*On evt2 = Time is 21:00*  
*Then activate the alarm system*

Nevertheless, we intentionally forget to consider the weekends as part of the *closing period* (i.e., 21:00-06:59, Saturdays, and Sundays) in order to present new changes on the mentioned rules. Now, the four above rules are replaced by the following:

- ☞ **Authorization Rule 1:**  
*If Name (Person) = Bashar  $\wedge$  Name (Action) = Enter  $\wedge$  Now (Time)  $\geq$  7:00  $\wedge$  Now (Time)  $<$  21:00  $\wedge$  Name (Resource) = Building 1R1  $\wedge$  Today  $\neq$  (Sun/Sat)*  
*Then allow the person to enter*
- ☞ **Authorization Rule 2:**  
*If Name (Person) = Bashar  $\wedge$  Name (Action) = Enter  $\wedge$  Name (Resource) = Building 1R1  $\wedge$  (Today = (Sun/Sat)  $\vee$  (Now (Time)  $<$  7:00  $\wedge$  Now (Time)  $\geq$  21:00))*  
*Then prohibit the person to enter*
- ☞ **Configuration Rule 3:**  
*On evt1 = Time is 7:00*  
*If Today  $\neq$  (Sun/Sat)*  
*Then deactivate the alarm system*

Furthermore, what if we want to consider public holidays (i.e., “Les Jours Fériés”) in the definition of the *closing period*. For instance, the 14<sup>th</sup> of July is a national day in France. So, another modification is to add new conditions to the existing rules, but in order to do what? What if we needed to include all national days, or others?

By considering the 14<sup>th</sup> of July, we have to following ECA policy expression:

- ☞ **Authorization Rule 1:**  
*If Name (Person) = Bashar  $\wedge$  Name (Action) = Enter  $\wedge$  Now (Time)  $\geq$  7:00  $\wedge$  Now (Time)  $<$  21:00  $\wedge$  Name (Resource) = Building 1R1  $\wedge$  Today  $\neq$  (Sun/Sat)  $\wedge$  Today  $\neq$  14<sup>th</sup> July*  
*Then allow the person to enter*
- ☞ **Authorization Rule 2:**  
*If Name (Person) = Bashar  $\wedge$  Name (Action) = Enter  $\wedge$  Now (Time)  $<$  7:00  $\wedge$  Now (Time)  $\geq$  21:00  $\wedge$  Name (Resource) = Building 1R1  $\wedge$  (Today = (Sun/Sat)  $\vee$  Today = 14<sup>th</sup> July)*  
*Then prohibit the person to enter*
- ☞ **Configuration Rule 3:**  
*On evt1 = Time is 7:00*  
*If Today  $\neq$  (Sun/Sat)  $\vee$  Today = 14<sup>th</sup> July*  
*Then deactivate the alarm system*
- ☞ **Configuration Rule 4:**  
*On evt2 = Time is 21:01*  
*If Today = (Sun/Sat)  $\vee$  Today = 14<sup>th</sup> July*  
*Then activate the alarm system*

In fact, what we are trying to demonstrate is that ECA/CA expressing process gain weight each time considering new conditions, especially if we what to add the summer holidays as well. Any new consideration of conditions is just to clarify more the introduction of new suggested circumstances. Within the possible behavior changes ( $\Delta$ ) (i.e., opening and closing periods), for example we want to consider:

#### **Circumstances:**

*Bashar wants to enter building 1R1, time is 14:00 now and today is Friday.*

#### **New circumstances:**

*Bashar wants to enter building 1R1, time is s 14:00 now and today is Friday 8<sup>th</sup> of August.*

To consider the new circumstances, we need to update the rules by adding new conditions as well. Upon these new circumstances, evaluating the expressed conditions should decide what actions to do (i.e., permissions are given or not (P/P’), and configurations are changed (Config/Config’)).

By observing what we demonstrated, we did not change the expression of permissions (P/P’) neither the expression of configurations (Config/Config’). What we are changing each time to meet the new circumstances by adding the new conditions is to understand the changes ( $\Delta$ ). So, what is missing to optimize the policy expression?

The ECA/CA policy rule misses the **semantics** that express what a behavior change is about. Therefore in order to express the *closing period*, we needed to add additional conditions to help the rule itself understand and consider what “*closing period*” really meant.

As a result, we add more conditions to the policy rules in order to explain that the “*closing period*” meaning involves non-working hours (i.e., 21:00-06:59), weekends, vacations, national-days, etc. From this observation, this is one problem that the current ECA/CA approaches encounter.

Now, we want to introduce new exceptional circumstances that introduce new rules. Let us imagine that reconstruction works will take place during the month of November at building 1R1.

Eventually, this exception implies allowing workers to enter building 1R1 during their working hours (i.e., from 7:00 to 18:00). Hence, the alarm system is deactivated at the reconstruction period. However, Bashar is not allowed to enter the building for safety reasons. As a result, we will need to improve the previous policy by 1) inserting new **five** rules and 2) updating the other old rules (i.e., with the new underlined conditions in order to ensure the policy consistency and avoid any rules conflicts). So, the following nine rules express the new ECA policy:

- ☞ **Authorization Rule 1:**  
*If Name (Person) = **Bashar**  $\wedge$  Name (Action) = **Enter**  $\wedge$  Now (Time)  $\geq$  **7:00**  $\wedge$  Now (Time)  $\leq$  **21:00**  $\wedge$  Name (Resource) = **Building 1R1**  $\wedge$  Today  $\neq$  (Sun/Sat)  $\wedge$  Today  $\neq$  14<sup>th</sup> July  $\wedge$  Month  $\neq$  November  
**Then allow the person to enter***
- ☞ **Authorization Rule 2:**  
*If Name (Person) = **Bashar**  $\wedge$  Name (Action) = **Enter**  $\wedge$  Now (Time)  $<$  **7:00**  $\vee$  Now (Time)  $>$  **21:00**  $\wedge$  Name (Resource) = **Building 1R1**  $\wedge$  (Today = (Sun/Sat)  $\vee$  Today = 14<sup>th</sup> July  $\vee$  Month  $\neq$  November)  
**Then prohibit the person to enter***
- ☞ **Authorization Rule 3:**  
*If Name (Person) = **Bashar**  $\wedge$  Name (Action) = **Enter**  $\wedge$  Name (Resource) = **Building 1R1**  $\wedge$  Month = November  
**Then prohibit the person to enter***
- ☞ **Authorization Rule 4:**  
*If Role (Person) = **Worker**  $\wedge$  Name (Action) = **Enter**  $\wedge$  Now (Time)  $\geq$  **7:00**  $\wedge$  Now (Time)  $\leq$  **18:00**  $\wedge$  Name (Resource) = **Building 1R1**  $\wedge$  Today  $\neq$  (Sun/Sat)  $\wedge$  Today  $\neq$  14<sup>th</sup> July  $\wedge$  Month = November  
**Then allow the person to enter***
- ☞ **Authorization Rule 5:**  
*If Role (Person) = **Worker**  $\wedge$  Name (Action) = **Enter**  $\wedge$  Now (Time)  $<$  **7:00**  $\vee$  Now (Time)  $>$  **18:00**  $\wedge$  Name (Resource) = **Building 1R1**  $\wedge$  (Today = (Sun/Sat)  $\vee$  Today = 14<sup>th</sup> July  $\vee$  Month = November)  
**Then prohibit the person to enter***
- ☞ **Configuration Rule 6:**  
*On evt1 = **Time is 7:00**  
**If** (Today  $\neq$  (Sun/Sat)  $\vee$  Today  $\neq$  14<sup>th</sup> July)  $\vee$  Month = November  
**Then deactivate the alarm system***
- ☞ **Configuration Rule 7:**  
*On evt2 = **Time is 21:00**  
**If** Today = (Sun/Sat)  $\vee$  Today = 14<sup>th</sup> July  $\vee$  Month  $\neq$  November  
**Then activate the alarm system***
- ☞ **Configuration Rule 8:**  
*On evt3  $\neq$  **Month is no more November**  
**Then activate the alarm system***

Giving the above rules, we note that we updated the rules (1, 2, 6, & 7) in order to avoid conflicts and handling the exceptional reconstruction work of November in the new rules (i.e., 3, 4, 5, & 8). On the one hand, the authorization rule (3) aims at preventing Bashar from entering during November to the 1R1 building. On the other hand, the authorization rules (4 & 5) aim at controlling the entering of workers to be only from 7:00 to 18:00 and preventing their entrance outside this range.

Nevertheless, do these rules correspond and conform to the specified security requirements? Surely, no one can answer this question easily without diving inside the rules and the conditions, and potentially be lost. It is obvious that the nine rules are complicated for such simple requirements.

Concerning the alarm system, the exceptional reconstruction work in November requires to deactivate the alarm system during the worker's working hours and to activate it outside these hours and during weekends.

By observing the new ECA policy, what we were doing is just trying to refine the rules to be dynamic. The refinement is by going finer and finer with the details in expressing the rules in order to consider the new exception. In other words, we see that using events introduce more fine-grained rules, which extend the policy size and increase potential conflicts between them. Therefore, a second problem we raise by using ECA is the lack of **abstraction** in expressing the policy rules.

Accordingly, expressing dynamic security management using complex policies is a result of the missing *abstraction* and *semantics* in ECA/CA policies. When semantics are missing, the complexity is remarkable by the size of the rule. In consequence, rules get more complex by adding new conditions in order to provide the required semantics. When abstraction is missing, the complexity is remarkable by the number of the rules. Hence, the policy gets larger by adding new rules to meet the new circumstances (i.e., exceptional reconstruction work).

As the dynamic management of security is a complex problem, one needs a simple methodology to express the policy. Therefore, we imperatively need first a term that bestows both semantics (i.e., meaning to the rules so that they become simpler) and abstraction (i.e., aggregating conditions and circumstances as much as possible to reduce the rules number).

To summarize, we used the previous example to demonstrate our motivation behind 1) Why we need a feedback with *abstraction* to express a dynamic policy, and 2) why we need a feedback with *semantics* to express a dynamic policy. Finally, we introduce the need for an answer to the question: 3) how to express a dynamic security policy that ensures a feedback with both?



# I. Expressing the Policy with Feedback

The innovation brought by PREDYKOT was essentially around the reasoning box, which provides a feedback on the behavior changes ( $\Delta$ ) to the policy (PREDYKOT Posters Co-summit, 2015). It reasons on the collected data (e.g., time now is 7:00 or Bashar is asking to enter) coming from the technical managed environment (e.g., the 1R1 building) to provide this feedback (e.g., closing or opening periods).

We claim that the expression of this feedback must have two features: the meaningfulness (i.e., to reflect semantics) and the abstractness (i.e., to conceptualize the policy and to aggregate rules). For instance, the meaningful feature comes from the feedback “*closing period*” because it indicates that, “*when it is the closing period, the building is closed.*”

Therefore, in what concerns the meaningfulness, one can feel how the following expression: “*when it is the opening period, then allow Bashar to enter the 1R1 building*” is **natural** in term of the human languages (i.e., here it is English) and **high-level** in term of being very close to the defined requirements. By analyzing this expression, we can identify the sentence “*when a meaningful feedback caused by certain circumstances and within certain conditions, then do some actions*”. Consequently, we can better structure this expression as follows:

**When** meaningful feedback (about new *circumstances* happening)

**And** certain conditions (on these new *circumstances*)

**Then** do some actions (to adapt to these *circumstances*)

Now, the first question we need to answer in this section is: what terms could help representing the proposed expression structure?

Moreover, the same feedback (i.e., “*closing period*”) is abstract as well because it does not define any values for the period to which the building will be closed. For instance, the feedback “*closing period*” does not define imperatively that 14<sup>th</sup> of July is included (i.e., it is not a national day for other countries than France). Moreover, the feedback is abstract because it does not specify in what intention the usage of such meaning will be in term of security management (e.g., to allow or to deny persons). For instance, the “*closing period*” does not imply the prohibition of employees to enter the building at the summer vacation (e.g., letting PhD students continue their researches during August). Therefore, the closed meaning for the policy is abstract and does not associate or specify any decision. Finally, the feedback abstraction provides a sort of independency in management. For instance, the policy is interested only in managing the security state of the 1R1 building during the “*closing period*”. As a result, the edition phase of policies should specify for each feedback, the corresponding rules.

Therefore, in what concerns the abstractness, we need to conceptualize this feedback and study what conditions and circumstances could specify and identify its happening, duration, meaning, and usage. At the same time, we need to understand the possible security states caused by receiving a meaningful feedback, and then aggregate the policy rules under each of these states.

Now, the second question we need to answer in the next section is: what conceptual model could help us understand the feedback and its consequences, and how to manage and identify the meaningful feedback?

To summarize, we have the two following questions in order to express the policies in the structure: *When semantics, circumstances and conditions are met, then do some actions*. To answer the first one, we have to find suitable **terminologies** to represent the different semantics, circumstances

and conditions. To answer the second one, we have to propose an engineering *methodology* of analyzing the requirements in order to consider the meaningful feedback (i.e., semantics, circumstances and conditions). This methodology should contain also a modeling approach to conceptualize the relationship between the semantics and the policy permissions and configurations. Answering both questions would fulfill expressing the policy rules with the above-proposed structure.

## I.A. Terminology

Observing failures and other behaviors, whether desired or undesired, in large-scale software or systems of specific domains (telecommunication systems, information systems, online web applications, etc.) is difficult. Very often, it is only possible by monitoring and examining the runtime behavior of these systems through operational logs or traces. However, these systems can generate data on the order of gigabytes every day, which creates a challenge in predicting upcoming critical problems or identifying relevant behavior feedbacks. One can say that there is a gap between the amounts of information a system has and the amount of information a system needs to make a decision (Fülöp et al., 2010).

Nowadays, advancing tools for managing and processing technical events are introduced to the market (e.g., Security Information Event Management (SIEM), Intrusion Detection Systems (IDS), etc.). Hence, the feedback on events represents now an essential phase in order to produce the semantics. However and in order to express feedback on events more efficiently, we need to investigate results from more specialized research fields.

Complex adaptive systems are a research field with intensive studies on the behavior of their actors. The aim is at recognizing highly dynamic behavior changes in order to observe and learn about the unpredictable actors' activities. For instance, studies on *Pervasive Computing* systems have focused on two research trends to handle the dynamic feedback: context and situation awareness. Therefore, it is crucial to define what a context is and what a situation is. Should security management use situations, contexts or both? What is best to use for expressing the meaningful feedback?

To overcome these doubts, we choose to express the feedback using the term *situation*. However, we argue that the *context* represents an undeniable term to use in order to complete our proposed policy rules structure instead of ECA/CE. Thus, using both terms *situation* and *context* we propose the following SCD structure:

**When** *Situation* = meaningful feedback (about new *circumstances* happening)

**And** *Context* = certain conditions (on these new *circumstances*)

**Then** *Decisions* = do some actions (to adapt to these *circumstances*)

The most common definition of the *context information* was provided as “any information that can be used to characterize the situation of entities (i.e., whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves” (Dey et al., 1999, Page 3).

Concisely, we refine what does the *situation* term mean, and what is used for in the literature. Then, we contribute on adding our proper definition and explain how to make the use of it in this thesis. Moreover, we justify our choice of using the *situation* for expressing the semantics. Likewise, we define what does the *context* term mean, and what it is used for in the literature. Then, we contribute on adding our proper definition and explain how to make the use of them in this thesis. Moreover, we justify our choice of using the *context* for expressing the conditions (i.e., associated



with certain circumstances). Finally we define what does the *decision* term mean and what is our contribution to this part of the structure.

## I.A.1. Situation

Different faces of situation could be relevant to use for representing the meaningful feedback. The Centre of National Resource Textual and Lexical of France (CNRTL) continues the initial definition of a situation as “*a set of conditions and circumstances in which one finds oneself*<sup>16</sup>.”

- **At a given time**, the situations semantics could be described as awful, advantageous, critical, cruel, terrible, exceptional, favorable, bad, painful, emergent, fake in which they can be examined, flipped, reversed back into a given or an initial situation.
- **At a given point of view**, the situation can be administrative, financial, legal, physical, military, a property status, a family status, irregular, a marital status etc. “A person inquired straight away on our business situation, and though he was very young, and limited in his means, he paid everything! The look that marked the greatest affection on my sister and me” [Restif La Bret., Nicolas, 1796, p. 113].

Saint Thomas Aquinas, the Italian theologian has described situations as: “Every step, every situation reflects your state of mind and, similarly, carries a spiritual meaning” (CNRTL). From this saying, one can match between human and systems’ elements, mind and requirements, spirituality and behavior. As a result, a new definition may appear “every situation reflects a system’s (element) state against its requirements and, by the same token, carries a meaning about its behavior.”

Situations are expressive and meaningful words that can describe the crucial conditions and circumstances that invite the managed elements’ behavior to change. It is generic enough to express several scales and therefore to be included during the requirements specification, policy specification and expression, configuration, and assessment of technical environments’ behaviors.

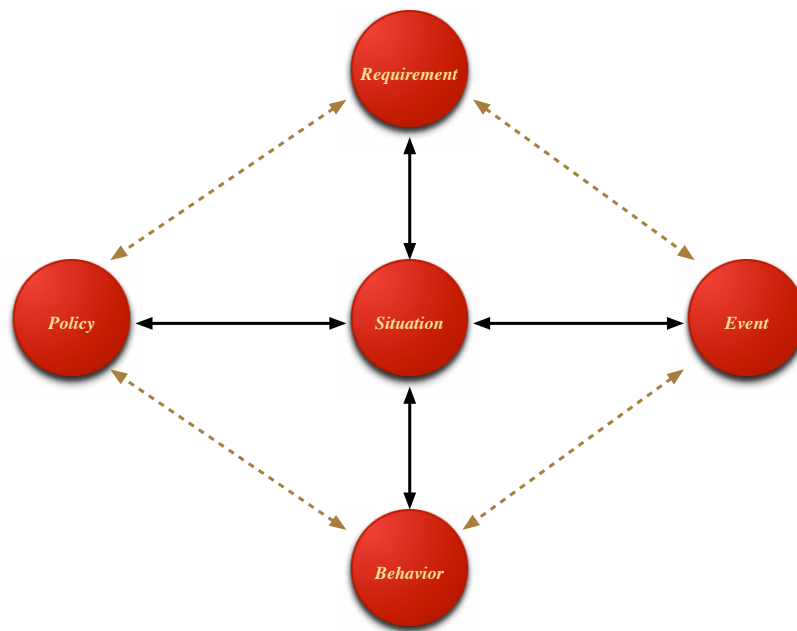
Situations are not limited to defining the state of a person or a task, but they express the state of a system, subsystem, and any member of the system. However, the effort in the definition is to pick up the right word that expresses the interesting situation. One-way is to extract situations and express them using abstract keywords. By analyzing the requirements documents, any keyword that has a significant appearance or importance in the documents can be recognized as a situation, but only if it is possible to define the conditions and the circumstances for a specific subject (i.e., a member of the system).

By definition, the situation is a time-related concept, and it is subject to have a start and an end, which creates a period of time representing the situation lifetime or duration. For instance, in our example, the working environment of the 1R1 building has two situations where the building opens and closes. We define the first situation, as it is expressed in the above requirements, “*opening period*”. This situation starts every day at 7:00 and ends at 21:00 (i.e., outside the weekends, national days, etc.).

---

<sup>16</sup> Collins, Oxford & Larousse Dictionaries





**Figure 4.34: Situation as a core concept.**

In pervasive systems, studies advancement on the situation-awareness domain has proposed a more detailed definition: “*A set of contexts in the application over a period of time that affects the future system’s behavior*” (Yau et al., 2006). Here, a context refers to any instantaneous, detectable, and relevant property of the environment, system, or user, such as location, available bandwidth, and user’s schedule. Pervasive systems collect data to predict or anticipate a situation (Tsai, Liu, Chen, & Paul, 2005). A more technical definition for situation is “*a relevant time frame calculated based on events generated by available sensors*” (Thollot, 2012; Thollot, 2013). Our work is more concerned with defining situations to identify them, once they appear without predictions. However, we hope to derive benefits from predicting future situations. Therefore, we refine the mentioned definition for the purposes of this work as “*a set of contexts detected because of predefined conditions and circumstances within the application over a timeline that affects the current and future system behavior.*”

Ongoing works to distinguish context and situation have been presented by other studies (Boytssov & Zaslavsky, 2013; Janiesch, 2010; Yau, Huang, Gong, & Seth, 2004; Yau, Wang, & Huang, 2003). One definition of the situation for an application software system is “*an expression on previous device-action record over a period of time and/or the variation of a set of contexts relevant to the application software on the device over a period of time. Situation is used to trigger further device actions*” (Yau, Wang, Karim, 2002). A formal definition is proposed as well by considering the situation as problems: “*A situation is a problematic and developing state of a computational element characterized by its context. ‘Problematic’ means that there is a problem of which a system should take care in a given situation and ‘developing’ means that a situation is changeable to another situation or state by system operations for solving the problem*” (Kim & Kim, 2009, Definition 3.1, p. 557). The provided formal definition is  $S = \langle C, H, P, A \rangle$ , where C is the context information, H is the history of actions, P is the problem, and A is the plan (set of actions). A “*situation is a concept of developing a state as well as an understanding context. **Situation is more like a process, while the context is more like data.** That makes it easy and efficient to design and implement dynamic systems based on the concept of situation*” (Kim & Kim, 2009, p. 556).

From the previous definitions of a situation, one can recognize that a situation is mainly defined by its conditions and circumstances. However, this recognition tells only the beginning of the situation. Situations have a duration bounded by the time. Every situation must end at a specific time. Likewise, a situation ends after recognizing certain conditions and circumstances. Moreover, the situation is defined as entity-related, which means that each situation is associated with and owned by a specific entity. For instance, in our example, the working environment of the 1R1 building has two situations that represent the building's state open and close. We define the first situation, as it is expressed in the above requirements, "*opening period*". This situation starts every day at 7:00 and ends at 21:00 (i.e., working days – outside the weekends, national days, etc.). We define the second situation as the "*closing period*" that starts after the end of the first situation (at 21:00) and ends before the starting of the next situation (at 7:00). It also includes weekends. Thus, the "*closing period*" during Saturdays and Sundays starts each Friday's night (at 21:00) and continues until next Monday's morning (7:00). Likewise at the national days (i.e., 14<sup>th</sup> of July), the "*closing period*" is from the night of July's 13<sup>th</sup> (at 21:00) and until the morning of July's 15<sup>th</sup> (at 7:00).

Nevertheless, we recognized a situation with the start and the end of certain conditions and circumstances, but what does a recognized situation represent by itself during its lifetime? The situation represents meanings, semantics. For example, the situation "*opening period*" represents the following semantics: the 1R1 building is open for employees during the period 7:00-21:00 on working days. These semantics are embedded in this situation. Therefore, using situations will simplify the security management, but it is a complex process that we need to manage externally by defining and identifying (or recognizing) the situation. Why externally? Because as we argued before, integrating an *event* inside the policy expression (i.e., ECA) introduces complexity to the policy!

To conclude, situations represent the best candidate to express the behavior of the system. However, it is undeniable that the policy expression needs to consider events and contexts. Once the dynamicity is expressed, the security policy expression can be easier because the knowledge about the environmental behavior it manages and governs, helps defining more consistent and coherent rules with both the requirements and the environment. Studies have frequently promoted the situation notion as the best formalism in expressing and representing the dynamic behavior, because it is closer to the requirements than events and contexts (Loke, 2006).

## I.A.2. Context

Linguistically, The Oxford dictionary defines the **context** as the “set of facts that surrounds a particular event, situation, etc.”

Previously, we have provided the most common definition that associates the *context information* definition with the situation definition as “any information that can be used to characterize the situation of entities (i.e., whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves.” (Dey et al., 1999, Page 3).

A Context is a *fact*. Hence, examples of the information about contexts could be the role, location, identity, state of people (groups or, computational and physical objects), actions, etc. For instance, Emma’s role is a doctor, the doctor’s name is Emma, Emma is running to reach the door, Emma is angry and Emma is at the emergency room.

Any event, situation, or fact that is not surrounded with a clear context is *raw* (e.g., a raw event could be “a person heard a noise”) (Akyildiz et al., 2002; Dorn et al., 2008; Melgani et al., 2001). Any event that is not specified to have a sense, and as a result, is not classified or categorized, would be considered as raw. That is why the example “a person heard a noise” is a raw event with no sense, as we don’t know what person, what noise, and where, etc.

Therefore, the way we view things in our daily lives is affected by their context in order to be specified. The context could be related to several elements that could represent the circumstances and the conditions. However, various refinements of the context definitions in term of awareness do exist, Dey’s definition is widely accepted and adopted to provide a consistent understanding of the subject studied by researchers [INCOME, 2012, Chapter 4, Deliverable 2.1].

In this thesis work, the context is the group of facts (i.e., contexts and conditions) that specifies and refines a situation and a policy rule. For the situations, the context groups the conditions and circumstances that represent their start and end. For instance, the “*opening period*” situation gives the semantics that the 1R1 building is opened within the following context: time is between 7:00-21:00, the days are working days outside the reconstruction works period i.e. other than weekends, national days, vacations, etc., and the months are other than November. The context is recognized using the help of detected events (we will explain the technical relationship at the next section). For the policy rules, the context is the group of facts that are used in the conditions to specify the rules. For instance, the name of the person requesting the access is *Bashar*, the action requested is to *enter*, and the building name is *IRI*.

We did precise what the context for the situation is, but what is the situation for the context in the policy rule? The situation gives semantics to the given and defined context. For instance, the context detected as *someone is running to exit the mall*. The situation gives semantics to this context. This means, that if the situation of the mall is *on fire*, then the person is running away from the fire (i.e., the given semantics). If the situation of the mall is *being robbed*, then the person is suspected as he/she is running away from the crime scene (i.e., the given semantics).

To summarize the *context*, the SCD structure has the following expression: *When Situation And Context Then Decision*. In our contribution, this *context* is of two types: a situation context and a policy context. It is an expressive and a rich source of information extracted from the requirements. By referring to our previous example, for us the *context* represents all the following examples:

- *Bashar wants to enter the 1R1 building, time is 14:00 now and today is Friday.*
- *Bashar wants to enter the 1R1 building, time is 14:00 now and today is Friday 8<sup>th</sup> of August.*
- *The name of the person requesting the access is Bashar*
- *The name of the requested action is Enter*
- *The time now is 7:00, 21:00, 18:00, or 5:00, etc.*
- *The name of the resource, where the access request is controlled, is Building 1R1*
- *Today is Sunday, Saturday, Monday, etc.*
- *Today is a national holiday 14th July, Thanksgiving, etc.*
- *Today is a vacation day Christmas, etc.*
- *There are some reconstruction works this month of November.*
- *The role of the person is reconstruction worker*
- *The alarm system state is active, inactive, etc.*
- *Etc.*

Therefore, the *context* participates in the policy expression at both situation and policy levels. The first level is whenever defining the start and the end of a situation (i.e., helping the recognition of a situation). The second level is whenever defining the conditions of the policy rules.



*As a conclusion on contexts and situations, contexts are used to calculate and identify situations, where situations provide semantics to contexts.*

### I.A.3. Decision

A decision is a rule part that is applied upon the evaluation of relevant situation(s) and relevant context(s). There are two types of the security management decisions: authorizations and configurations. The permissions are the authorization decisions to allow or deny a subject from accessing a requested resource. Defining permissions is associated at least with three types of elements: subjects, objects/resources, and actions. However, other elements types may participate in defining permissions such **as the environment** in where the permission is being given. For instance, an example on the permissions is *allowing Bashar entering the 1R1 building during the opening period*. This example gives the permission of *allowing* the subject 'Bashar' to effecting the action 'entering' on the controlled resource 'Building 1R1' within the "opening period" environmental situation.

Configurations are the management decisions to provision new settings, parameters, etc. The security management involves obligating managed elements to perform certain actions (e.g., activating the alarm systems). Therefore, such security obligations are inclusively part of the security management decisions. Security obligations are high-level expressions, where configurations are technical representations. For instance, alarm systems should be deactivated during the "opening period" situation and activated during the "closing period" situation. Technically, configurations are the actions of setting the alarm on/off called whenever needed.

Finally, our SCD policy permits providing a *default* decision in a special SCD rule. The rule structure is then:

***When** there is no specific situation,  
**And** there is no specific context,  
**Then** give a default decision*

## I.B. Methodology: the SCD Policy Engineering

We contribute with a methodology to engineer security management policies<sup>17</sup>. Our engineering methodology aims at guiding the expression of the dynamic security management policy using the SCD structure, in order to be easily represented later with a technical language. The policy expression methodology we propose aims at analyzing the security requirements to extract the triples: situations, contexts, and decisions. Afterwards, we model the situations relationships and associate each situation with relevant triples. Finally, we express the policy rules using the SCD structure.

### I.B.1. SCD Policy Specification

To specify a dynamic security policy, we extract the SCD triples by analyzing the provided requirements. Given the example that we mentioned, we have initially two requirements. For them, we specified three situations: “*opening period*”, “*closing period*”, and “*reconstruction period*”.

As we mentioned, every situation is defined with start and ending contexts. Each context is a group of circumstances and conditions. The “*opening period*” situation is defined with several starting and ending contexts<sup>18</sup>:

1. **Starting Context:**

- Time now is 7:00
- Today is a working day (i.e. not a weekend, not a national holiday etc.)
- Etc.

2. **Ending Context:**

- Time now is 21:00
- Today is a weekend
- Today is a national holiday
- Today is a vacation day
- Etc.

The situation “*reconstruction period*” is defined with several starting and ending contexts:

3. **Starting Context:**

- The Beginning of November, there is reconstruction work
- Other dates ...

4. **Ending Context:**

- The end of November, reconstruction work should be finished
- Otherwise, extensions ...

---

<sup>17</sup> We refer with the name *Policy Engineering* to the study presented in LANOMS (Zhang et al., 2005): “*Security management of information systems is a complex and daunting task in most organizations. Management policies are then introduced to guide and control the management of information systems, but the management of management policies is also a complex task. Therefore, we propose a new concept—policy engineering, which uses the philosophy and paradigms of established engineering disciplines to address these problems. The introduction of policy engineering will provide a systematic methodology that can be used to guide and control the management of information systems.*”

<sup>18</sup> Nevertheless, the following starting and ending contexts are related to time (period of time), which makes three situations time-related situations. We can also define starting and ending contexts by other identifiers such as the location (space). For instance, we have the situation “*stolen*”. We can identify the location of a shop's items with location patches, using Google coordination (X, Y). The stolen situation's starting context is when the item is no longer inside the shop and it has not been sold.

Logically, the 1R1 building is initially closed and then it is opened for employees to work, then it is closed again, and so on. At November, the building is closed at the night (from 21:00 to 7:00) before the beginning of the reconstruction works. Therefore, the next situation of the “*closing period*” is also the “*reconstruction period*”. Once November is finished, the building gets back to the “*closing period*” situation. Afterwards, the building continues the successive repetition between closing and opening periods.

The “*closing period*” situation as a core situation starts whenever the “*opening period*” situation ends, and ends whenever the “*opening period*” situation starts. Likewise, the same “*closing period*” situation ends whenever the “*reconstruction period*” situation starts, and starts whenever the “*reconstruction period*” ends. Hence, the context that triggers a situation always ends another one. However, as we will see later on the next technical section, the starting and ending contexts are dynamic points. This means, the situation in term of semantics stays the same (e.g., “*opening period*”), but the context may change (e.g., “open day event”). Therefore, the situation will start with different conditions and circumstances but will give the same semantics (i.e. the 1R1 building is open).

Now, the specification of the policy rules context is our methodology’s next step. From the requirements, we identify that there are the following contexts to include in the policy:

- *The name of the person requesting the access is Bashar*
- *The name of the requested action is Enter*
- *The name of the resource, where the access request is controlled, is Building 1R1*
- *The role of the person is reconstruction worker*
- *Etc.*

Finally, the specification of the desired decisions is our methodology’s next step. Our policy decisions are of two types: permissions and configurations. From the example’s requirements, we identify the following decisions (we do not explicit):

- *Allow Bashar to enter building 1R1*
- *Deny Bashar to enter building 1R1*
- *Allow workers to enter building 1R1*
- *Deny workers to enter building 1R1*
- *Activate the alarm system*
- *Deactivate the alarm system*

## I.B.2. SCD Policy Modeling

During the thesis work, we have noticed about the situation notion its expressiveness feature. We found that a situation links the associated entity to the requirements. It reflects the entity’s behavior by showing its current state, it gives a semantic to the context in the policy rules, it also aggregates situations contexts and SCD rules, etc.

Therefore, we decided to model the situation of each entity in which we manage its security. For instance, we are managing the security of the 1R1 building. Hence, we model the situations of the building and we associate to each situation the relevant contexts and decisions.

As changing the situation of the building means changing its states as well, we found that the software engineering model ‘*state diagram*’ is the closest model to our needs. Thus, we consider each state to be a situation and each transition is a starting context of a situation and an ending context to another one. However, adjusting the diagram was not sufficient to include the policy decisions and contexts. Hence, we propose a rectangular box associated to each situation. As a result, we represent the SCD triples and the relationships between them (see Figure 4.35).



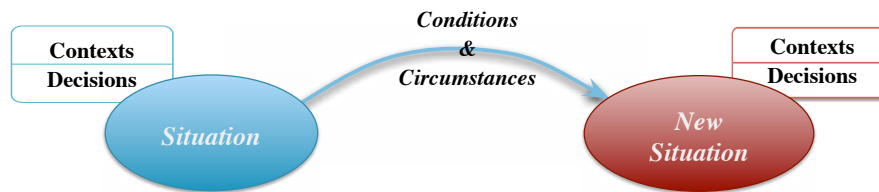


Figure 4.35: Modeling situations using state diagrams

Given the defined SCD triples for our previous example, we can use our model to conceptualize the SCD policy. Figure 4.36, shows three states representing the three defined situations: *opening*, *closing periods*, and *reconstruction period*. Firstly, the previously defined *starting context* ends the situation “*closing period*” and as a result orients the policy towards the contexts and the decisions associated with the new state (i.e., “*opening period*”). Secondly, the *ending context* ends the situation “*opening period*” and as a result orients the policy towards the contexts and the decisions associated with another state (i.e., “*closing period*”). Thirdly, the *starting context* for the situation “*reconstruction period*” starts it and ends the previous situation “*closing period*”. Finally, the *ending context* for the situation “*reconstruction period*” ends it and starts the situation “*closing period*” again. As an observation, the situation “*closing period*” occurs after the “*opening period*” and after the “*reconstruction period*” as well.

As a result of this step, we obtained a conceptual view on the policy that we are willing to express. This facilitates even more the task of expressing the rules in the SCD structure.

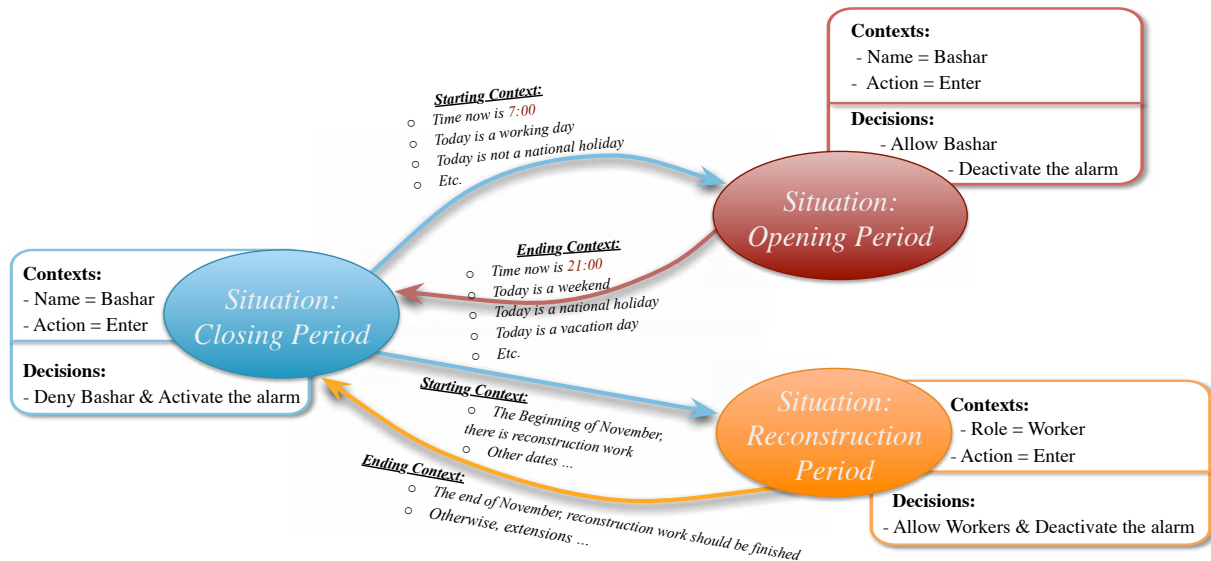


Figure 4.36: Modeling the situations of 1R1 building.



### I.B.3. SCD Policy Expression

Now that we defined the situation, the context and the decision terms, we want to use the SCD structure to express any dynamic security management policies. With the help of the previous state diagram, let us express the example we mentioned earlier:

- ☞ **SCD Rule 1:**  
*When Situation (Building 1R1) = "Opening Period"*  
*And Name (Person) = Bashar  $\wedge$  Name (Action) = Enter  $\wedge$  Name (Building) = 1R1*  
*Then allow the person to enter*
- ☞ **SCD Rule 2:**  
*When Situation (Building 1R1) = "Reconstruction Period"*  
*And Role (Person) = Worker  $\wedge$  Name (Action) = Enter  $\wedge$  Name (Building) = 1R1*  
*Then allow the person to enter*
- ☞ **SCD Rule 3:**  
*When Situation (Building 1R1) = "Opening Period"*  
*And Situation-Status (Building 1R1) = "Beginning"*  
*Then deactivate the alarm system*
- ☞ **SCD Rule 4:**  
*When Situation (Building 1R1) = "Closing Period"*  
*And Situation-Status (Building 1R1) = "Beginning"*  
*Then activate the alarm system*
- ☞ **SCD Rule 5:**  
*When Situation (Building 1R1) = "Reconstruction Period"*  
*And Situation-Status (Building 1R1) = "Beginning"*  
*Then deactivate the alarm system*
- ☞ **SCD Rule 6:**  
*When Any other situations are identified*  
*And Within any contexts*  
*Then Decision is to deny access*

We used five rules to express the predefined security requirements. The first rule is chosen whenever the situation "opening period" is recognized. It is applied if the context matches: *Bashar wants to enter building 1R1*. Upon choosing this rule, the applied decision is to *allow him to enter*. The second rule is chosen whenever the situation "reconstruction period" is recognized. It is applied if the context matches: *a person with the Worker role wants to enter the 1R1 building*. Upon choosing this rule, the applied decision is to *allow him/her to enter*. The third rule is chosen whenever the situation "opening period" is recognized. It is applied only at the context where the situation "opening period" has just begun. Upon choosing this rule, the applied decision is to do an action (i.e., *deactivate the alarm system*). At the opposite way, the fourth rule is chosen whenever the situation "closing period" is recognized. It is applied only at the context where the situation "closing period" has just begun. Upon choosing this rule, the applied decision is to *activate the alarm system*. Exceptionally, the fifth rule is chosen whenever the situation "reconstruction period" is recognized. It is applied only at the context where the situation "reconstruction period" has just begun. Upon choosing this rule, the applied decision is to *deactivate the alarm system*. Whenever there are situations and contexts that do not match the previous five rules, the sixth rule is chosen and the decision by default is to deny all access requests.

In summary, we proposed an analytical approach to engineer the policy expression based on expressing the feedback using SCD triples: situations, contexts, and decisions (actions). We added a modeling approach using the state diagram that helps understanding what are the security requirements in terms of these SCD triples. Both approaches aim at facilitating the policy expression, and consequently will improve its evaluation and enforcement.

As the methodology conclusion, we want to stress the value of our contribution. First, we want to recall the specified security requirements (SR). Table 4.10 demonstrates two columns: the specified security requirements number, and the specified security requirements description.

**Table 4.10: Specified security requirements**

SR	Description
SR 1	<i>Bashar has the right to enter building 1R1 during the opening period</i>
SR 2	<i>The alarm system is deactivated during the opening period</i>
SR 3	<i>The alarm system is activated during the closing period</i>
SR 4	<i>Workers have the right to enter building 1R1 during the reconstruction period</i>
SR 5	<i>The alarm system is deactivated during the reconstruction works</i>
SR 6	<b><i>Deny-all:</i></b> <i>- Workers does not have the right to enter building 1R1 outside the reconstruction period</i> <i>- Bashar does not have the right to enter the 1R1building during the closing period</i>

By empirically comparing the policy expressed using ECA/CA (i.e., eight rules, page 68) and our SCD policy (i.e., six rules), **one can easily remark the difference in the rules number**. More importantly, with less number we expressed simpler rules in term of the conditions complexity. Our rules conditions are simpler and closer to the SR descriptions than the ones expressed with ECA/CA policies.

Our methodology for the policy expression is simple, yet powerful. We externalize the complexity of the events expression to be outside the policy, how? We added the abstract and the meaningful term: *situation*. With the term situation, we bestow the detected context and the specified policy semantics and abstraction. Therefore, we reduce the complexity of expression by using the semantics. Moreover, we reduce the complexity of the policy size by using the abstraction to aggregate the rules.

## II. The Feedback Implementation

We specified our expression of the feedback on the behavior changes to be based on *situations*. We presented that starting and ending points surround each situation. Moreover, we explained that these points are defined using *contexts*. Between the starting and the ending points there is a lifetime duration that we defined as *lifespan*. Hence, we need a management paradigm for situations to identify starting and ending contexts and to maintain their life during the totality of the lifespan.

Accordingly, it is important to point out that the definition of situations implies eventually the definition of contexts. Therefore, the identification process is essential after the definition of situations to assign a concrete managed element in order to distinguish its occurrence. Briefly, the definition of situations creates a powerful abstraction that needs to be assigned a context (i.e., starting and ending points, and managed elements). The assignment process is known as the identification of situations. Situation identification is the outcome of managing situations.

### II.A. Situation Management

Situation management (SIMA) identifies effective methods for situation recognition, prediction, reasoning and control<sup>19</sup>. The SIMA term has been widely used in different domains, especially in critical ones like cyber security, physical infrastructure, pervasive computing, air traffic control, battlefield operations management, disaster and crisis management, homeland security, etc. Its importance comes with domains interested in the detection and processing of situations—normal and, more particularly, abnormal—identified based on events or alarms (e.g., intrusions) (Cochran, 1996), and interested in managing complex dynamic operations, networks, and systems. Modern U.S. defense policy highlights the importance of *situations* by expecting that wars in the future will be characterized by *heightened mobility, increased operational tempo, and more complex and dynamic situations*. Therefore, policy decisions and actions will require an effective methodology for monitoring, awareness, recognition, prediction, reasoning and control of *situations*, partially or wholly, which collectively are recognized in literature as SIMA.

The management of situations aims essentially at managing and controlling distributed heterogeneous information sources, processing real-time (or nearby real-time) event streams, and representing and integrating low-level technical events and higher-level requirements (business goals). Moreover, SIMA includes the task of fusion and presentation of multi-source information that amplifies the human understanding, reasoning about what is happening and what is important in the environments' behavior (Jakobson, Lewis, Matheus, Kokar, & Buford, 2005).

Technologies are generous in providing technical solutions for the SIMA task. The main trends that overlap with this task's objectives are *information fusion, intelligent sensing, sensing grids, situation-awareness, context-awareness, and complex event-processing* architectures (IEEE CogSIMA 2011-2014).

In our contribution, we consider SIMA as a paradigm that aims at defining, identifying and maintaining situations lifespan. Therefore, we define SIMA mainly with three phases in order to implement our feedback.

---

<sup>19</sup> Information rephrased from the following website at 21 July 2015:  
[http://cms.comsoc.org/eprise/main/SiteGen/TC\\_SM/Content/Home/Situation\\_Management.html](http://cms.comsoc.org/eprise/main/SiteGen/TC_SM/Content/Home/Situation_Management.html)

- i) The first phase is to technically design situations definitions. These definitions we already specified during the *SCD Policy Specification* step (i.e., analyzing requirements to recognize keywords that could describe relevant situations such as: *opening period, closing period, intrusion, attack, emergency, etc.*).
- ii) The second phase, which happens once the situation is defined and designed, is the technical identification of situations through starting and ending contexts.
- iii) Finally, the third phase is to maintain the situations alive all along their lifetime duration (i.e., *lifespan*). As far as a situation of a managed element is maintained alive, the SCD policy knows that this situation is still valid.

Therefore, SIMA is a crucial step in our contribution because of its relationship with both the environment and the policy (Jakobson, Buford, & Lewis, 2007). Its outcome is *situations* that represent the core of our contribution. SIMA is a very flexible and abstract paradigm that we specified in this thesis work to respond to security management requirements. Therefore, we need to technically implement this paradigm.

Amit<sup>20</sup> – The Situation Manager is just a conceptual solution, provided by IBM Research Laboratory in Haifa, which proposes a generic architecture to implement the SIMA phases. The Amit solution guidelines what functionalities the software module (i.e., the situation manager) should provide to the interested applications. Moreover, the solution describes the relationship between the situation manager and the events information sources (see Figure 4.37).



**Figure 4.37: General view on the architecture of the Amit Situation Manager.**

Nevertheless, IBM Amit is a high-level conceptual contribution with no implementation. Therefore, we inspire from their solution and contribute with an implementation with respect to their proposition. Our implementation is limited to meet the dynamic security management requirements that we presented during the state of the art.

The conceptual architecture has the objective of monitoring the runtime behavior and receives information about the events' occurrence. Then, it detects the desired situations by means of applications that require reactions. Finally, it reports the detected situations to the subscribers' applications. The Amit – Situation Manager hands over the situation-detection's responsibility from the application level to a higher-level tool. Moreover, it bridges the situations that require reactions and the application. It provides a general solution (i.e., a solution that is practical in many domains) that can express the fundamentals of a situation definition.

Extracted from the article "*Amit – The Situation Manager*" (Adi & Etzion, 2004), we quote the following definition:

*A tool that includes both a language and an efficient runtime execution mechanism aimed at reducing the complexity of active applications. This tool follows the observation that in many cases there is a gap between current tools that enable one to react to a single event (following the ECA: event-condition-action paradigm) and the reality in which a single event may not require any reaction; however, the reaction should be given to patterns over the event history.*

<sup>20</sup> Active Middleware Technology

Nevertheless, we need to implement our own situation manager enlighten by the Amit generic architecture and definition. We believe that implementing the situation including the starting and ending contexts is undoable without using *events*. However, the need to detect and process more complicated events (e.g., attacks) introduces new technologies. In this thesis, we have chosen to follow the complex event processing technology to implement our situation manager. We develop next our motivation to justify our choice.

## II.B. Complex Event Processing

Events are the core elements for understanding a system's behavior and adjusting it to requirements. Events are relatively important phenomena that add fresh news to reality and may eventually participate as a group of events in creating other phenomena within defined contexts or semantics. The following study is extracted based on information from the National Centre of Textual and Lexical Resources (CNRTL) of France. It defines an *event* as "A fact that leads to a situation."

Events may happen across various layers of an organization, as sales leads, orders, or customer service calls. Nevertheless, they may be new items, text messages, social media posts, stock market feeds, traffic reports, weather reports, or other kinds of data. An event may also be defined as a "change of state," when a measurement exceeds a predefined threshold of time, temperature, or other value.

In fact, in reality we can only observe indications, warnings, or signs. For example, a man notices that his family has started to drink more juice, so he needs to add an additional carton of juice to the grocery list. Therefore, the man observed the consumption of his family after the first time the family was out of juice. After the third time that this phenomenon occurred, he reached his conclusion. The example shows that there are two types of events: low-level events or indications and others that are an outcome of observing these events.

Studies were able to achieve the detection of sophisticated events such as hacks and threats using data analytics tools known as the deep packet inspections (DPI) (Porter, 2005; Y. Cao et al., 2013). Moreover, the detection of intrusions into networks and systems was ensured using tools under the IDS category (intrusion detection systems) (WT Work - NIST, 2003). Finally, the security incidents and malwares are discoverable with tools called SIEM (Security Information and Event Management) (Gabriel et al., 2009). All IDS, DPI and SIEM are tools with the objective of detecting complicated processed events.

Nevertheless, we require in this thesis work to identify semantics and not only technical events (otherwise, we would use ECA). For instance, we give the following example: "a person is running to exit the store". Simultaneously, the store manager is pressing the emergency button. Consequently, a call to the nearest police station was automatically performed. Panics were recognized from observing the other clients inside the store. The reader would intuitively conclude that this is a robbery scene. We come with an understanding out of composing all these events together (i.e., a semantic: there is an ongoing robbery in the store).

We dress the semantic with what we called a *situation*. Event processing tools (e.g., IDS, DPI, SIEM, etc.) are not designed to provide such semantics. Therefore, we choose the complex event processing technology because, in addition to processing events, it aggregates, composes, coordinates and correlates events to provide complex events. Moreover, it redoes all these operations on these complex events to produce more complex events, and so on until constructing the beginning of an understanding, which is the beginning of a situation.

Therefore, among all possible technical solutions to ensure SIMA and implement the situation manager, we justify our choice of using the complex event processing technology. It is an event-driven solution that processes events and reprocesses events to produce more *complex* events. It aims at helping the applications to express the detailed behavior changes through resulting in an aggregated feedback. A *complex* event is not a sophisticated signification of the occurrence of events groups. *Complex* refers to the difficult effort we put on processing several events in order to help producing a meaningful feedback.

The first introduction of the concept of “*complex event processing*” was in August 18, 1998, at Distributed Systems at Stanford University:

*Complex event processing is a new technology for extracting information from distributed message-based systems. This technology allows users of a system to specify the information that is of interest to them. It can be low level network processing data or high level enterprise management intelligence, depending upon the role and viewpoint of individual users. It can be changed from moment to moment while the target system is in operation. (Luckham & Frasca, 1998)*

Complex event processing (CEP) applies to a wide range of studies on business processes, artificial intelligence, network administration, security of information systems, etc. Therefore, commercial, academic, and free tools are available for different aims and with powerful and advanced features.

On April 23, 2007, Tim Bass from TIBCO Software Inc. introduced TIBACO as an example of CEP tools. TIBCO defined CEP as:

*An emerging network technology that creates actionable, situational knowledge from distributed message-based systems, databases and applications in real time or near real time. CEP can provide an organization with the capability to define, manage and predict events, situations, exceptional conditions, opportunities and threats in complex, heterogeneous networks. Many have said that advancements in CEP will help advance the state-of-the-art in end-to-end visibility for operational situational awareness in many business scenarios. These scenarios range from network management to business optimization, resulting in enhanced situational knowledge, increased business agility, and the ability to more accurately (and rapidly) sense, detect and respond to business events and situations. (Vermesan & Friess, 2013, p. 91)*

CEP is an event processing loop that starts by combining information (data) from multiple heterogeneous sources to infer composition of events or events patterns, known as *complex events* (CEs). The CEP objective is to identify CEs and to associate them, if possible, with meanings or semantics. A CEP-based system should be capable of monitoring, processing (i.e., filtering, coordinating, composing, and aggregating events into CEs), reasoning, and analyzing (Adi & Etzion, 2003). CEP systems are rule-based systems with representation languages (i.e., event processing languages [EPL]) to express CEP rules that supervise, and may control, other systems.



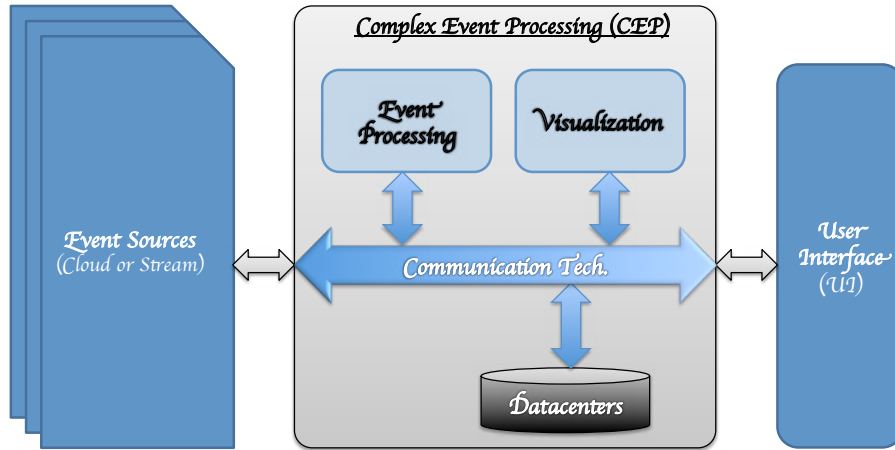


Figure 4.38: General view of the CEP architecture.

Using CEP requires a functional architecture to help organizations understand and organize the system requirements based on analyzing the behavior. Figure 4.38 presents the common architecture that each CEP-based agent usually respects.

CEP relies on a number of techniques to process CEs, including event-pattern detection, detecting relationships between events (such as causality, membership, or timing), and event correlation (a technique for making sense of a large number of events and pinpointing the few events that are really important in that mass of information).

Processing the complex events aims to monitor and analyze the information collected (data) about events, and their meaning, in order to derive conclusions that help to understand the behavior. The data might be observed from structured and organized sources (e.g., datacenters), converged sources (e.g., event streams), or event vast sources (e.g., clouds).

The event processing systems are embedded in monitoring and reasoning engines, decision support systems, and reporting systems (e.g., activity logs, historization, and audit). Inside sources, event-processing systems locate events in two forms: time-based and storage-based. Storage-based events are partially ordered or disordered inside the cloud. Events could be sequenced based on timelines inside *event streams*. A special technology used in event stream processing (ESP) is *time windows*. Normally, these windows are movable and dynamic with time, and therefore they are known as “*sliding windows*.”

Nevertheless, what is a cloud event, and what is a stream event? When event cloud processing is relevant, and why event stream processing is more interesting for our contribution?

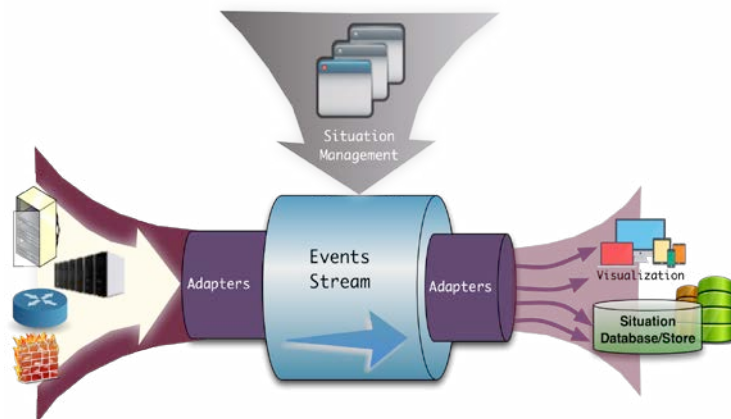
Events stream is a time ordered sequence of events in time, while events cloud is unordered. For example, an events stream contains the following sets:  $\{\{t_0, \text{“a person is running toward the exit of the store”}\}, \{t_0+1, \text{“the store manager pressed the emergency button”}\}, \{t_0+2, \text{“panic is detected inside the store between clients”}\}, \{t_0+6, \text{“the nearest policy station is informed”}\}, \text{etc.}\}$ . On the other hand, events cloud with the same content will have the following unordered sets:  $\{\{t_0+6, \text{“the nearest policy station is informed”}\}, \{t_0+1, \text{“the store manager pressed the emergency button”}\}, \{t_0, \text{“a person is running toward the exit of the store”}\}, \{t_0+2, \text{“panic is detected inside the store between clients”}\}, \text{etc.}\}$ . We find that events cloud is storage-based and similar to the unstructured data in databases. One needs to use unordered data for the ease of data insertion, but with no concern for the data retrieval efficiency. For instance, we can use events cloud for the archiving process and historization. Cloud events are also relevant to store a huge amount of data, which is the famous *big data* (Costa et al., 2013). Moreover, analyzing cloud events is interesting whenever there are unknown patterns to



discover through machine learning and expert systems (e.g., in medical and biological data) (Fülöp et al., 2010).

Events stream is time-based and reputed with its strength and its efficiency in term of retrieving events rapidly (Cugola & Margara, 2012). Moreover, event stream processing helps the momentarily identification of events. This helps create time-related reasoning on the occurrence of events. Therefore, ESP meets our requirements in creating the starting and ending points, and inclusively establishing the situation lifespan.

In Figure 4.9, the relationship between SIMA and CEP (i.e., based on ESP) takes a bidirectional dependency form. Situation management is about defining situations through specifying the conditions and the circumstances of the behavior that start and end them. Afterwards, SIMA identifies situations through the CEP. The CEP tools provide the ability to keep an eye on the environment's behavior through predefined rules. As one of these tools, the event stream processing (ESP) contains adapters (mainly two, an input and an output ones) that observe and detect interesting behaviors according to the defined rules and aggregate them in a stream of events. In the stream, the role of SIMA is to configure the ESP with rules that aggregate, collect, coordinate, and compose events into CEs. The latter will then be used to identify situations.



**Figure 4.39: Situations and complex-events relationship in event stream processing (ESP).**

We found an open source solution that helps us implementing our situation manager using the ESP technology. Esper proposes a technology and technical framework to manage the processing of events. This is an extraction from the company definition of the technology:

*Esper is an Event Stream Processing (ESP) and event correlation engine (CEP, Complex Event Processing). Targeted to real-time Event Driven Architectures (EDA), Esper is capable of triggering custom actions written as Plain Old Java Objects (POJO) when event conditions occur among event streams. It is designed for high-volume event correlation where millions of events coming in would make it impossible to store them all to later query them using classical database architecture.*

*A tailored Event Processing Language (EPL) allows expressing rich event conditions, correlation, possibly spanning time windows, thus minimizing the development effort required to set up a system that can react to complex situations.*

*Esper is a lightweight kernel written in Java, which is fully embeddable into any Java, process, JEE application server or Java-based Enterprise Service Bus. It enables rapid development of applications that process large volumes of incoming messages or events.*

In Esper, each events stream is related to a type of events. For instance, fire incidents are sent to a specific stream. Warnings about potential attacks are sent to another events stream.

Esper EPL is what we use to express all *complex events*. The language syntax is very close to the SQL language. Thus, it contains all the common operations: group by, having, from, select, mathematical operations (e.g., count, max, etc.), where, etc. We present the two following EPL structures to introduce the syntax:

```
Select eventStreamName.AttributeName as alias,  
MathOperations(eventStreamName.AttributeName) as alias, etc.  
From eventStreamName.win:time(value of minutes, seconds, etc.) as alias,  
Where eventStreamName.AttributeName1 = eventStreamName.AttributeName2
```

In Esper CEP, we use EPLs to retrieve events from events streams (i.e., instead of tables in databases). Each event contains attributes with values that are similar to the columns of the tables. We can apply mathematical operations on the retrieved values (e.g., min, max, count, etc.). EPL also allows the retrieval of events within a sliding window (i.e., win:time). The sliding window allows the retrieval of events within a window of time defined for example by minutes or seconds. The term *sliding* means that the events retrieved within the 10 second wont be the same as the next coming 10 seconds. It is at the moment of firing the query that we count 10 second before and so on. Finally, we can add the usual logical conditions to perform the join between streams or attribute values.

Using Esper EPL, we can also define the patterns. Patterns allow the concept of subscription to a type of events for a period or time or forever. At the following structure, we define a pattern to retrieve events from an event stream every time the conditions are met:

```
Every eventStreamName (AttributeName =value, AttributeName > value, etc.)
```

However, we can limit the events retrieval by defining a timer to stop for example after one hour (i.e., 60 minutes. Therefore, we add the following line to the pattern:

```
Where timer:within(60 minutes)
```

## II.C. Adaptable Situation Manager

Our situation manager (SM) aims at ensuring the implementation of all our SIMA defined phases. Therefore, it has three functionalities to accomplish each of the SIMA phases. We claim our SM to be adaptable as we configure and reconfigure situations without stopping its functionalities. Hence, situations can be defined and redefined whenever it is necessary.

Our SM respects the Amit guidelines in term of language and architecture. However, Amit is a conceptual architecture and an abstract language. The latter expresses situations with the relevant events and defines the concept of the *lifespan* (i.e., what we use to draw the duration of a situation) that is bounded by an initiator (i.e., our starting point) and a terminator (i.e., our ending point) (Adi & Etzion, 2002).

What we retain from the Amit guidelines on the architecture is the fact that the situation manager (Adi, Botzer, & Etzion, 2002) aims at observing the behavior changes to provide a semantic feedback to subscribed applications (i.e., which is exactly our main objective using SIMA). However, as SIMA phases vary from an application domain to another, the Amit Situation Manager does not impose any implementation. Hence, as we did specify our required SIMA phases, we need then to implement our own SM functionalities to ensure these phases.

Likewise, the abstract language in Amit aims only at describing situations. However, there is no specific representation language that is proposed to technically express our situations. Thus, we need as well to use our own language to express situations.

Our technical implementation of the SM architecture is a CEP-based solution, namely built on the Esper technology. Our technical expression and identification of situations using the SM architecture is ensured using Esper EPL language (presented earlier), which respects the Amit language definition (Adi & Etzion, 2004).

The SM architecture schematized in Figure 4.40 consists of three main components. The first one is the Context Manager, which is a CEP-based agent responsible for defining all the interesting CEs for the Situation Manager (SM). The second component is the SM itself, which is responsible for retrieving the starting point (SP) and ending point (EP) of each predefined or configured situation (i.e., requested by the administrator). Basically, the SM is also an intelligent CEP-based agent that provides semantics. Finally, the third component is the situation database, which is the store of current situations to be consumed by the relevant applications (i.e., in our contribution, it is the PDP agent) during the evaluation of the dynamic security policy when a decision is required. All these components cooperate to detect, identify, and deliver situations, which are a meaningful feedback about behavior changes, to the authorization system (precisely the PDP).

Based on the technical architecture in Figure 4.40, our SM components are responsible of delivering three functionalities to ensure the SIMA (Situation Management) phases as the following:

- (1) We contribute with the conceptual and technical edition of situations and contexts as the first functionality. The conceptual edition is through designing situations and the relevant contexts to define situations' SPs and EPs. In term of the technical edition, our SM architecture is adaptable in the sense of accepting dynamic changes on configurations (i.e., reconfigurations) during the runtime. This functionality is valid for both agents (context and situation managers) for the definition of relevant complex events (i.e., including detecting all raw events) and relevant situations (i.e., recognizing the SP and the EP).

- (2) Based on the provided situation design and configuration, our SM architecture can easily identify situations. The identification process contains two steps: identifying events to create complex events, and identify SPs and EPs to create situations.
- (3) Finally, our SM maintains the situation of a managed element during its lifespan by storing it in a database (i.e., keeping the PDP aware about the current situation of a managed element as far as this situation is valid).

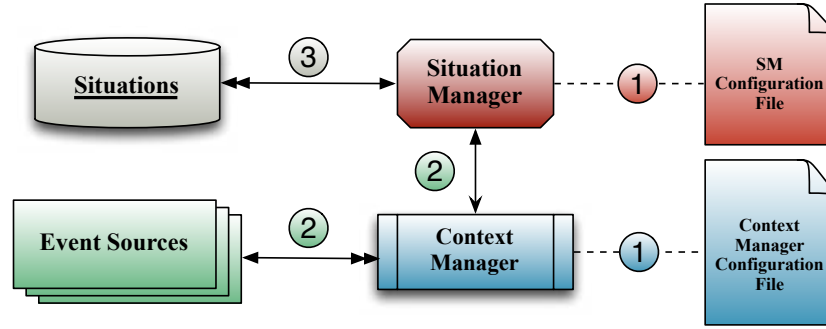


Figure 4.40: Our SM architecture.

## II.C.1. Situation Design & Dynamic Configuration

Some studies concerning the situation decision in the literature (Costa, Mielke, Pereira, & Almeida, 2012; Heckmann, 2006; Barford, 2010; Yau & Liu, 2006), aim at helping the situation identification (technical detection or recognition) through conceptually drawing the situation.

Within the same objective, we have our own definition of the relevant situations for the SCD policy. Therefore, we contribute on a specific technical design for our situation definition.

Each situation belongs to an entity and it appears and disappears in the constraint of time. The SP is defined when one of many conditions and circumstances (i.e., stating contexts) at a specific time allows the appearance of this situation. The EP is defined when one of many conditions and circumstances (i.e., ending contexts) at a specific time allows this situation to disappear. Thus, each situation has a lifetime characterized by its duration (i.e., between the situation's birth and death). This duration is known in research as the "*lifespan*" (Adi, Botzer, & Etzion, 2002).

Situations at the higher level are abstract because they reflect descriptive information (e.g., "*opening period*"). Their strength is at the lower level when attached to entities (i.e., "*opening period of the building IRI*"). Therefore, we stress that each situation is concerned by a unique entity (e.g., any managed element is a possible entity: *Bashar*, *workers*, *alarm system*, and in our example it is the *IRI building*). It is a many-to-many relationship, where a situation may belong to many entities, and an entity may have several situations. Briefly, each situation is characterized by the following elements (see Figure 4.41):

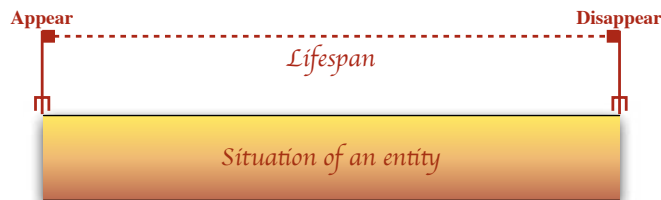


Figure 4.41: general design for a situation.

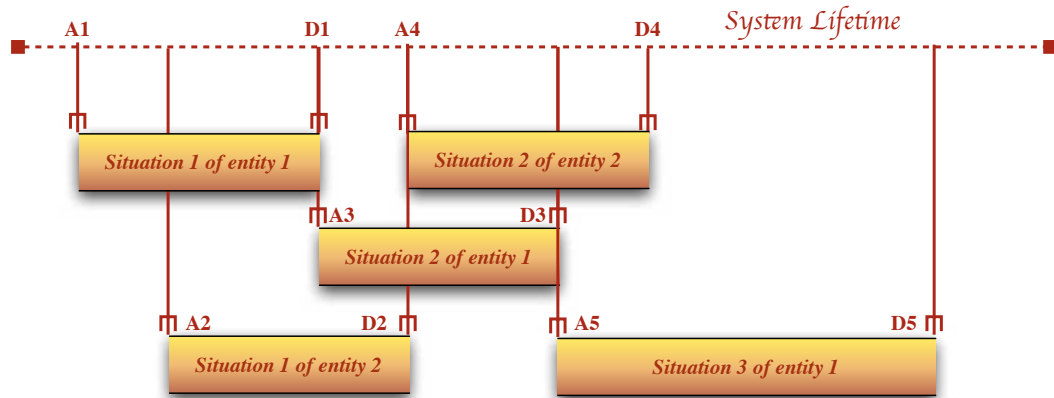
1. **Situation Name:** for example, *opening period* and *ending period*.
2. **Appearance:** the moment of detecting one of the *starting contexts* for a situation.

3. **Disappearance:** the moment of detecting one of the *ending contexts* for a situation.
4. **Lifespan:** the duration between the appearance and disappearance in which the situation will be preserved for consumptions.
5. **An entity:** the relevant managed element that we intend to secure.

Figure 4.42 shows a view on the sequential and instantaneous happening of several situations. It is possible for two situations or more to occur at the same time if they belong to different entities. In term of the entities-situations relationship, **there is no entity without a situation**. Moreover, there should be an initial situation for each entity. Thus, the default situation for all entities is “nominal,” which means that the entity is within its normal behavior and activities. To the limit of the thesis work, the managed situations we experienced are *conflict-free*, with the condition of respecting our policy engineering recommendation in term of SCD policy specification.

In this thesis work, we consider only one situation for one entity at a given time. However, situations-overlapping is possible when these situations belong to different entities. Therefore, we necessarily need more research effort on considering more situations for one entity. Situations then should not cause any contradictions in term of authorization or configuration decisions.

In Figure 4.42, let’s imagine entity 1 is the alarm system and entity 2 is building 1R1. We suppose that the alarm system has three situations: (1) Activated (2) Dysfunctional, or (3) Deactivated. We keep the same situations of building 1R1: (1) Open or (2) Close. Therefore, we can imagine the alarm system to be dysfunctional during both the opening and closing periods of building 1R1. However, as we mentioned before, this overlapping does not affect the security management decisions. There will be always one decision for each SCD policy evaluation because there is one matching for each rule. Hence, there is no overlapping between the situations of the same entity.



**Figure 4.42: Interrelationship between situations and entities**

Now, we designed the desired and relevant situations including defining their SPs and EPs in form of complex events. The question is how to represent technically this design in a way that the CEP-based agents (context and situation managers) can understand? Moreover, how to permit the administrator to dynamically update the technical representation of these situations and complex events without interrupting our dynamic security management system (i.e., in order to keep respecting PBM recommendations)? For instance, the French government was considering the city of Toulouse as part of the Zone A during the academic year 2014/2015 (i.e., the winter vacations start from the 7<sup>th</sup> of February). However, from this academic year 2015/2016 the city of Toulouse will be considered as part of the Zone C (i.e., the winter vacations starts 20<sup>th</sup> of February). Hence, the technical definition for the situations’ SPs and EPs has to be changed for both the opening and closing periods, in order to consider these new contexts.

Therefore, we need to dynamically (re) configure both the context and the situation managers. This means, redefine the CEs that express the contexts for the SPs and the EPs. Using dynamic configurations, we feature the Situation Manager and the Context Manager agents with adaptable upgrading of situation definitions, or totally adding new situations.

We manage the (re) configuration of both agents externally and without interrupting their functionality. We recognize and read the situation's configuration files in real time whenever a change is acquired. We applied this configuration strategy to the Context Manager as well to capture the events participating in identifying the SPs and EPs.

To represent this dynamic configuration, we propose a new configuration language to express the necessary Esper EPLs for both context and situation managers. Hence, to configure a situation in the Situation Manager (SM) we define two parameters: situation type and situation listener to instances. Likewise, we configure the Context Manager with two parameters: complex event types and complex events listeners. Thus, We propose a uniform representation for the configuration file for any CEP-based agent to be structured as the following and expressed using Esper EPL language:

```
IF
    StatementName = EPL Statement
Then
    StatementName;WhatListener;WhichAction=WithWhatParameters
```

Any of our *Update Listeners* by default recognize three actions:

- ☛ *Send*: we implement this action inside the listener of each CEP-based agent (i.e., situation or context managers) to send simple events **internally**. Therefore, all the agent's software objects communicate internally using Esper simple events. For instance, when the SM receives the beginning of the situation “*Opening Period*” (SP), the relevant update listener sends this event to the SM again as a request to create the situation (i.e., we call this event: *situation event*). As an example for the use of this action in the Context Manager, we use the *send* action to aggregate events. For instance, we detect raw events from many sensors at different locations. After processing these raw events we create a complex event. We send this complex event to be listened by the relevant software object. Briefly, we use the *send* action to exchange events internally between the update listeners, because each listener is interested with specific types of complex events. We give the following example:

```
Select (e1, e2, e5, ...)win:time(6 sec) From eventStream1 //Listener1 sends CE1
```

```
Select (CE1, CE3, C6)win:time(1 min) From CEStream3 //CEListener2 recieves CE1
```

- ☛ *Forward*: This action is similar to the *Send* one, but it aims at generating stream events to be sent **externally** (i.e., outside the relevant CEP-based agent). For instance, the communication between the Context Manager and the Situation Manager is managed using this action. Upon the listening of SPs or EPs, the Context Manager forwards a complex event to the Situation Manager.
- ☛ *Show*: This action aims at exposing and logging the relevant events to the user GUI (i.e., the administrator). This also keeps track of situations or complex events.

Our CEP-based agents listen to complex events and react with these three updates using Esper *Update Listener*. However, we want to allow future upgrades to these agents. Therefore, the



configuration file may contain any name of action. Though, if the declared action is not implemented inside the relevant *Update Listener*, the action unsurprisingly will not be executed.

Nevertheless, it is possible to integrate a new type of listeners other than the *Update Listener* in Esper<sup>21</sup>. Moreover, actions can be implemented inside the listener directly or called from an external implementation (i.e., using plugins technologies like: OSGi bundles or POJO).

At the Context Manager side (Figure 4.43), we demonstrate the following example for the complex event statement that represents the SP of the situation “*opening period*”. The statement contains the conditions and circumstances that permit the SP to be valid. Then, it associates a listener to this complex event statement. It configures this associated listener to update the SM using the *forward* action. Hence, upon the matching of this statement by a specific complex event, the associated listener will forward all the information of this detected CE. The destination of this forwarding is the SM and the type name of the *complex event to send* is SP event. Inside the Context Manager configuration file, the concerned part is:

```
#IF
ComplexEventStatement.OpeningPeriod= EPL Statement;
#Then
ComplexEventStatement.OpeningPeriod="
select * as OpeningPeriodSP from TimeDate_Events.std:lastevent(),
where (timeTaken.roundFloor('hour') == '7' and timeTaken.roundFloor('min') == '00')
and timeTaken.getDayOfWeek() not in ('5', '6')
and (timeTaken.getDayOfMonth() != '13' and timeTaken.getMonthOfYear() != '6')
and timeTaken.getMonthOfYear() != '10';22
ListenerName=DynamicComplexEventsListener;
Forward="ThisCEto::SituationManager::SendAs::SituationEvent;"
```

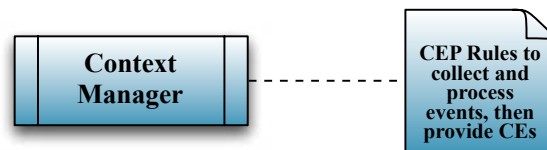


Figure 4.43: Context Manager configuration file

At the SM side (Figure 4.44), we demonstrate the following example for a situation statement inside the SM configuration file. The configuration simply aims at detecting all events from the type “*Situation Events*” forwarded by the Context Manager. Having an event from this type means the beginning of a new situation (i.e., inclusively means the ending of another). Therefore, we define the action *process situation* to check if the SP is related to the “*opening period*”, then the reaction will be to create the situation. Otherwise, if the SP is related to the “*closing period*”, then the reaction will be to end the situation.

```
#IF
SituationStatement.OpeneingPeriod= EPL Statement;
#Then
SituationStatement.OpeneingPeriod="select * from SituationEvent.std:lastevent()
as se where se.message like '%SP%' ";
ListenerName=DynamicSituationsListener;
ProcessSituation=OpeneingPeriod
```

<sup>21</sup> Esper provides the ability to integrate new listeners at runtime. We did not need this functionality in our implementation. More information is available inside the Esper guide’s StatementAwareUpdateListener.

<sup>22</sup> We may clarify more these four earlier condition statements. In fact, the first conditions statement verifies the current time is exactly equal to 7:00 am. Otherwise, the next condition statement verifies today is not a weekend (Saturday = 5, and Sunday = 6). If not, the next condition statement verifies today is not the 14<sup>th</sup> of July. Finally, the last condition statement verifies that the current month is not November (i.e., = ‘10’).



The action “*process situations*” exists only in the Situation Manager agent. It is applied upon the detection of events from the type “*situation event*”. Hence, the configuration file Context Manager does not contain this action. This action is not one of the default three actions: send, forward, and show. Therefore, the Situation Manager should contain the implementation of this action in order to be executed.

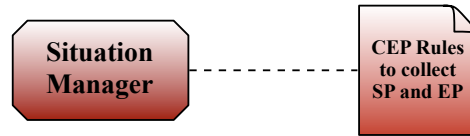


Figure 4.44: SM Configuration File

To summarize this functionality, for one situation “*opening period*” we needed to express the configurations at two locations: Situation Manager and Context Manager. At the Context Manager level, we configure the contexts that help identifying the SPs and EPs. At the SM level, we configure the link between the situations and their relevant SPs and EPs (i.e., identified and forwarded by the Context Manager). We stress the interest of this identification is the aggregation of the policy rules, thus reducing its complexity. By observing the ECA/CA rules expressed earlier, this situation and its context group the conditions expressed in the rules 1 & 6. Hence, we were able to use the situation name only “*opening period*” in our expressed rules 1 & 3. We externalized the complexity of the dynamic behavior (i.e., expressed by the conditions) to be managed outside the policy, but we keep its dynamic effects on the policy decisions (i.e., using situations and contexts as we will see at the underlying section “orienting mechanism”).

## II.C.2. Situation Identification

Technically speaking, this SM functionality aims at identifying the previously designed and configured situations. From this point, we aim at linking the provided technical design with the technical implementation of situations using the CEP techniques.

As abstract terms stating the businesses interesting to applications or systems (Costa, Guizzardi, Almeida, Pires, & Sinderen, 2006), situations can be identified straight from continuous data sources (e.g., motion detection of “walking” or “running” for knowing the room’s situation, or a temperature measure for forecasting a weather situation or determining a fever situation). Situations are dynamic, with varied durations; sequential; and interleaved to express the application, system, or network behaviors. Therefore, all transitions between the entities situations need to be controlled and identified (Ye, Dobson, & McKeever, 2012).

Figure 4.45 illustrates our approach of using Esper ESP to aggregate events in order to create CEs using *patterns* and sliding windows. Patterns are special structures of CEP rules that define the operation to process events (aggregation, coordination, etc.) within windows of time (i.e., for each pattern). To explain better through an example, some airports can track checked bags in case they get lost. Bags pass by four checkpoints equipped with wireless sensors and antennas.

An ESP collects all events coming from these points and keeps interesting events in the event stream. Based on CEP rules, the ESP processes the stream. Supposedly, one CE is identified when a bag is not on the conveyor belt anymore. The technique to detect that is through the ESP analyzing the event stream and coordinating four signals from the antennas. In case the last signal is not received or there were no events informing that the bag had passed by, then the CE is triggered. This CE is a candidate, alone or with other CEs, that defines the start of a situation (i.e., baggage lost).

Using CEP tools, the identification of situations is done by using CEP rules in SQL-like queries. Based on these rules, a CE identifies predefined conditions and circumstances expressing the behavior.

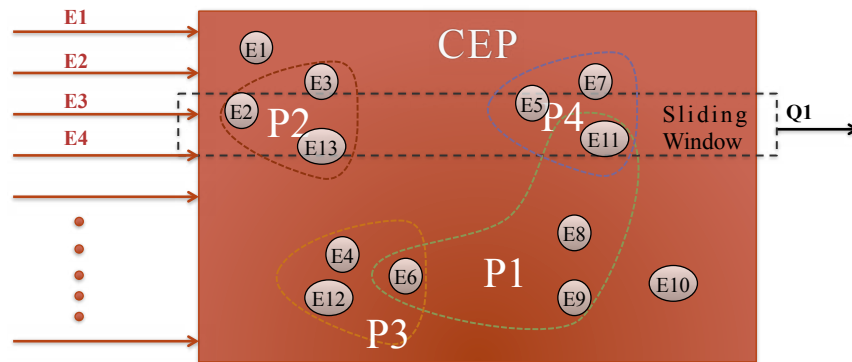


Figure 4.45: Stream processing using CEP rules.

In Figure 4.46, we link the situation design (i.e. proposed earlier in Figure 4.41) with the technical identification approach using ESP (see Figure 4.45). We already defined that there are two points bounding each entity's situation: a starting point (SP) (i.e., expressing its appearance) and an ending point (EP) (i.e., expressing its disappearance). Both points represent conditions and circumstances inside the environment, and they are identified by one or more CEs. For one entity, the previous situation's EP may or may not match the SP's CEs. Likewise, the next situation for the same entity may or may not match its SP with the EP of the current situation. However, the situations happening for one entity are certainly successive. The lifespan is defined by the duration between the EP and the SP. Moreover, during the lifespan of a situation, the events and the CEs do not stop expressing the behavior. Therefore, patterns are used also to ensure that there is no suspension or maybe a different EP or SP. For instance, the situation of gigantic access to a smart grid usually ends by suspending access requests to reduce the number of connections. However, the fact of losing the connection entirely ends the situation as well (i.e., this is a pattern defined in parallel to the main situation's identity).

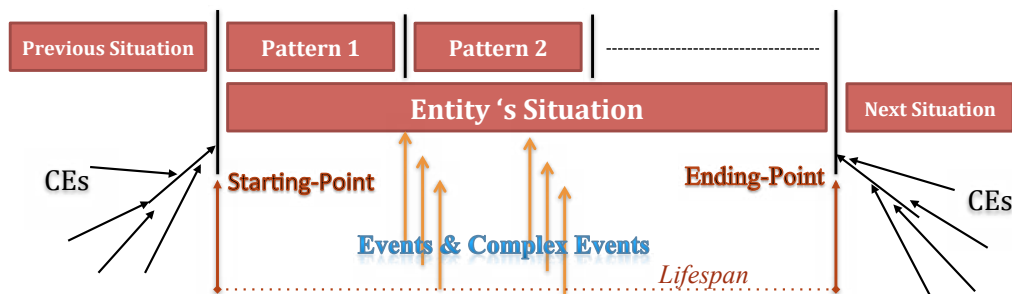


Figure 4.46: Situation identification technique.

The following query represents a pattern written using Esper EPL. The result of this query is a complex event that defines the SP of the situation *gigantic access*.<sup>23</sup> The situation SP starts whenever the number of different accesses detected becomes more than 15 within 5 minutes. Another possibility, SP may start whenever there is ten denied tries for the same access. We express these CEs using the following Esper EPL patterns:

<sup>23</sup> In networks, it is defined as the frequent number of attempts to access (a) resource during a short interval of time.

◆ CE Rule 1:

```
SELECT * as GiganticAccessSP
FROM pattern -- monitors the different access requests placed within a 5 min window
    [every access=AccessEvent ->
        (every(timer:interval(5 min) and not AccessType(id!=access.id)))]
HAVING (Count(*) > 15) -- if there are no 15 different requests, then no gigantic access
```

◆ CE Rule 2:

```
SELECT * as GiganticAccessSP
FROM pattern --monitors same access requests placed within a 5 min window
    [every access=AccessEvent -> ((every(timer:interval(5 min)) and
        (PolicyDecision(request.id==access.id) == deny)))]
HAVING (Count(*) > 10) -- if there are no 10 retries, there is no gigantic access
```

The EP could be a manual intervention from the network administrator. To keep our automatic solution, CE3 would be then to observe the administrators activities and end the situation upon doing a specific action (e.g., clicking on a button called “back to normal access”). Another way around is by using a policy obligation to remove the configuration “*block-access*” (i.e., provisioned automatically when this situation starts). CE4 is to observe the policy obligations and end the situation whenever there is a configuration to remove the “*block-access*”. Hence, this EP terminates the situation *gigantic access* by the happening of CE1, CE2, or both.

◆ CE Rule 3:

```
SELECT * as NominalAccessSP -- Conceptually it is the GiganticAccessEP as well
FROM AdminActivities.std:lastevent() as AA -- getting last activities made only
Where AA.Action == ClickOnButton(“back to normal access”)
```

◆ CE Rule 4:

```
SELECT * as NominalAccessSP -- Conceptually it is the GignaticAccessEP as well
FROM PolicyObligations.win:time(10 sec) as PO -- creating a sliding window to get last
    obligations within the 10 seconds from calling the query
Where PO.Config == “remove ‘block-access’ setting from edge-router XYZ”
```

We can imagine the following SCD policy rules that participate in this situation:

◆ SCD Rule 1:

```
When Situation (Network XYZ) = “Gigantic Access”
And Beginning of the situation (Network XYZ) = “Gigantic Access”
Then “block access” of the edge router XYZ
```

◆ SCD Rule 2:

```
When Situation (Network XYZ) = “Nominal Access”
And Ending of the situation (Network XYZ) = “Gigantic Access”
Then don’t “block access” of the edge router XYZ
```

Moreover, we can identify situations with even simpler EPLs than patterns in some scenarios. For instance, we can identify our earlier defined situations: *opening and closing periods*. We remind the SP of the “*opening period*” situation is one of the starting contexts (i.e., time now is 7:00, today is not a weekend, today is not a national holiday, there is no reconstructions this month (it is not November), etc.). Thus, the following CE1 expresses the starting contexts. As for the EP, the ending contexts are represented with the CE2 (i.e., whenever one of the following is valid: time now is 21:00, today is a weekend, today is a national holiday, today is a vacation day, there is reconstruction works, etc.).<sup>24</sup>

<sup>24</sup> We want to precise that Esper Tech defines Date-Time functions result with numbers that start by zero. Hence, the months are 0-11 and the days of the week are 0-6.

◆ **CE Rule 1:**

```
SELECT * as OpeningPeriodSP -- Conceptually it is the ClosingPeriodEP as well
FROM TimeDate_Events.std:lastevent(),
Where (timeTaken.roundFloor('hour') == '7' and timeTaken.roundFloor('min') == '00') and
timeTaken.getDayOfWeek() not in ('5', '6') and
(timeTaken.getDayOfMonth() != '13' and timeTaken.getMonthofYear() != '6') and
timeTaken.getMonthOfYear() != '10'
```

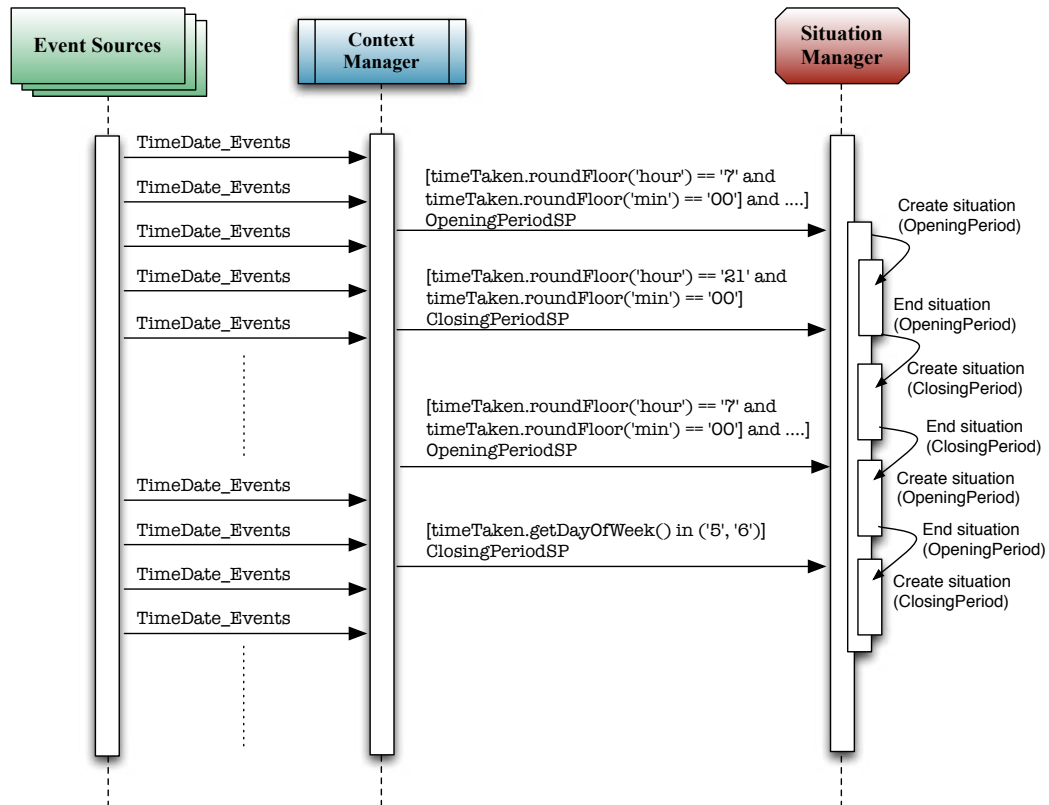
◆ **CE Rule 2:**

```
SELECT * as ClosingPeriodSP -- Conceptually it is the OpeningPeriodEP as well
FROM TimeDate_Events.std:lastevent(),
Where (timeTaken.roundFloor('hour') == '21' and timeTaken.roundFloor('min') == '00')
OR timeTaken.getDayOfWeek() in ('5', '6') OR
(timeTaken.getDayOfMonth() == '13' and timeTaken.getMonthofYear() == '6') OR
timeTaken.getMonthOfYear() == '10'
```

The technical recognition of complex events in ESPER is possible using *Listeners*. A *listener* is a monitoring function for complex events in a specified stream. Each listener can be associated with CEs in order to detect their occurrence. We are using *update listeners*. It is one of the Esper listener types that allow doing reactions upon the detection of CEs.

Therefore, we attach CE1 and CE2 to an update listener inside the Context Manager. Upon the identification of one of them, the update listener will notify the SM by generating a new CE3. This CE3 includes two messages to the SM: the starting of one situation and the ending of the other one.

In conclusion of this SM functionality, we want to give a bigger picture to the relationship between the SM agent, the Context Manager agent, the situations and the event sources. The Context Manager is the one detecting CEs that identify the SPs and EPs. Upon their identification, the Context Manager generates a new CE to the SM to inform the starting SP (or the ending EP) of a situation (cf. the sequence diagram, Figure 4.47).



**Figure 4.47: Sequence diagram of the interactions between event sources, the Context Manager, and the situation manager (including the starting and ending of each situation)**

### II.C.3. Situation Preservation

We preserve a situation by keeping it alive from its starting points until its ending points (i.e., within its lifespan). Therefore, the preservation functionality for the SM aims at answering the following questions:

- i) How to create a situation or how a situation appears?
- ii) How to maintain the situation during its lifespan?
- iii) How to remove a situation or how a situation disappears?

In Figure 4.48, we link each question with the relevant step that answers it. First, the Context Manager identifies the CEs composing the SP of a situation entity and informs the SM about it. This SP detection marks the real starting point of the processed situation. However, the situation at this moment is not created yet (i.e., physically) in our dynamic security system. Second, the SM detects the beginning of the situation and reacts by updating a java properties file (i.e., situations database) and adding the name of the situation with its value (e.g., Building 1R1 Situation = “*Opening Period*”). Third, the Context Manager identifies the CEs composing the EP of a situation entity and informs the SM about it. This detection of the EP also marks the real ending point of the processed situation. However, the physical ending of the situation is marked when the SM reacts by updating the java properties file and removing the value of the situation and adding the new value (i.e., as we mentioned, the EP of the situation is SP for another. Therefore, the update process could be considered also as replacing the situation of building 1R1 with a new one).

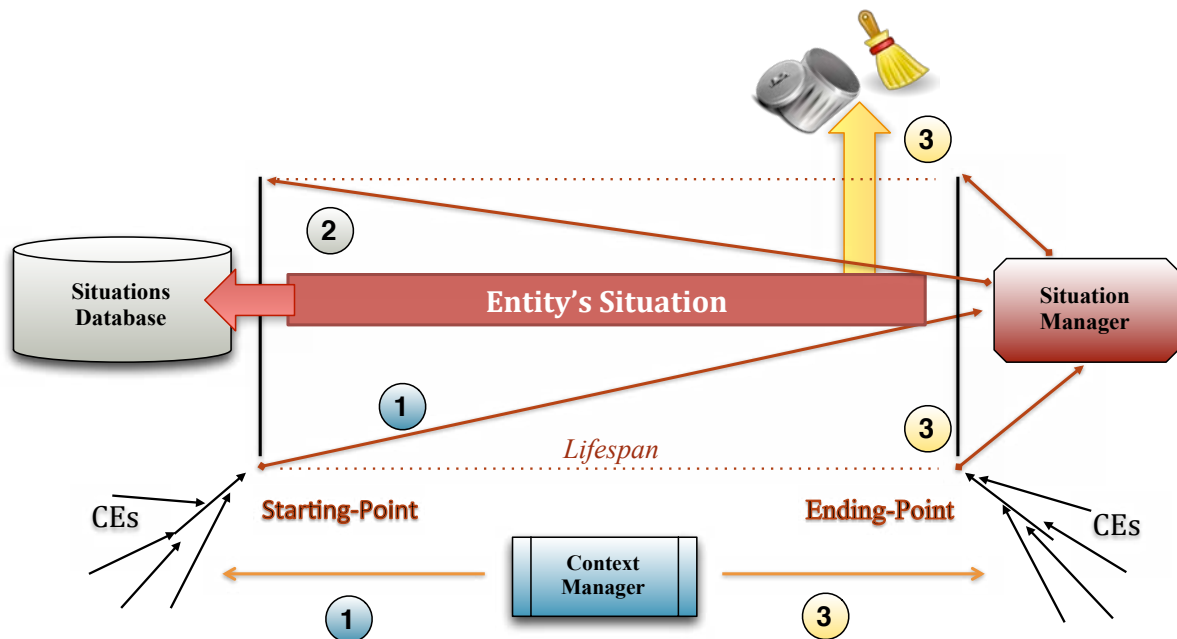


Figure 4.48: Situation preservation process

# III. Representing the SCD Policy-Expression

In Policy-Based Management, the management of the policy passes through a lifecycle. The first step of this lifecycle is the expression, the analysis, and the specification of the policy (i.e., what we presented to this point). The next step in the policy lifecycle is to represent the policy expression using a language. This language needs to be enforceable (i.e., there is a security management system that is able to enforce its decisions upon its evaluation). The final step then, is to implement and to enforce this represented policy using a technical architecture. (Zhang, Liu, & Wang, 2005)

Based on our presented state of the art, we highlighted that ABAC is a generic approach to represent any security management policy using the notion of *attribute*. We found OASIS XACML as the only standard policy language that respects ABAC recommendations. Therefore, we believe it's the best candidate as a technical language to represent our SCD policy expression:

<b>When</b>	<i>Situation</i>	= Attributes
<b>And</b>	<i>Context</i>	= Attributes
<b>Then</b>	<i>Decisions</i>	= Attributes

## III.A. Dynamic XACML Policy oriented by Situations

ABAC XACML is a generic, flexible, and abstract security policy language standardized by OASIS. XACML is used to describe general access and usage control requirements in terms of constraints on attributes. Specifically, attributes could be any characteristics of any category such as the subject, the resource, the action, or the environment in which the access request is made. Attributes have an identifier, which is a Uniform Resource Name (URN), and a data type, also identified by a URN. Considering attributes is what makes the language very flexible. Moreover, XACML is an XML-based language, and it is natively extensible.

Thanks to category tags, XACML version 3.0 (XACML-V3.0, 2013, OASIS Standard) is no longer limited to elements (subject, resource, action, and environment). Attributes are manipulated through predefined data types and functions. Aggregation allows the use of abstraction (i.e., being close to requirements) to express groups of rules. We use this feature to aggregate rules by situations and with respect to business requirements.

As a result, we are able for each one of the SCD triples to create an attribute category, and therefore each triple can have a corresponding representation in XACML. However, we need to define the SCD-XACML relationship: how these SCD attributes give dynamicity to the XACML policy?

### An Orienting Mechanism:

By orienting the policy using situations, we mean providing the policy decision point (PDP) through the specified XACML policy, which includes situations as attributes, the ability to give dynamic authorization decision at each access request.

Based on the XACML categories, the representation of the stable security policy is made possible using situations. The situations are defined as new category of attributes that aggregates the rules of the policy. Hence, each relevant situation to the management of security is associated with authorizations (conditions and decisions) and configurations (i.e., advice and obligations).



In Figure 4.49, the XACML policy structure presented in Chapter 2 is associated with the SCD expression structure. This association is to reflect where the situations, contexts and decisions are stressing the XACML policy.

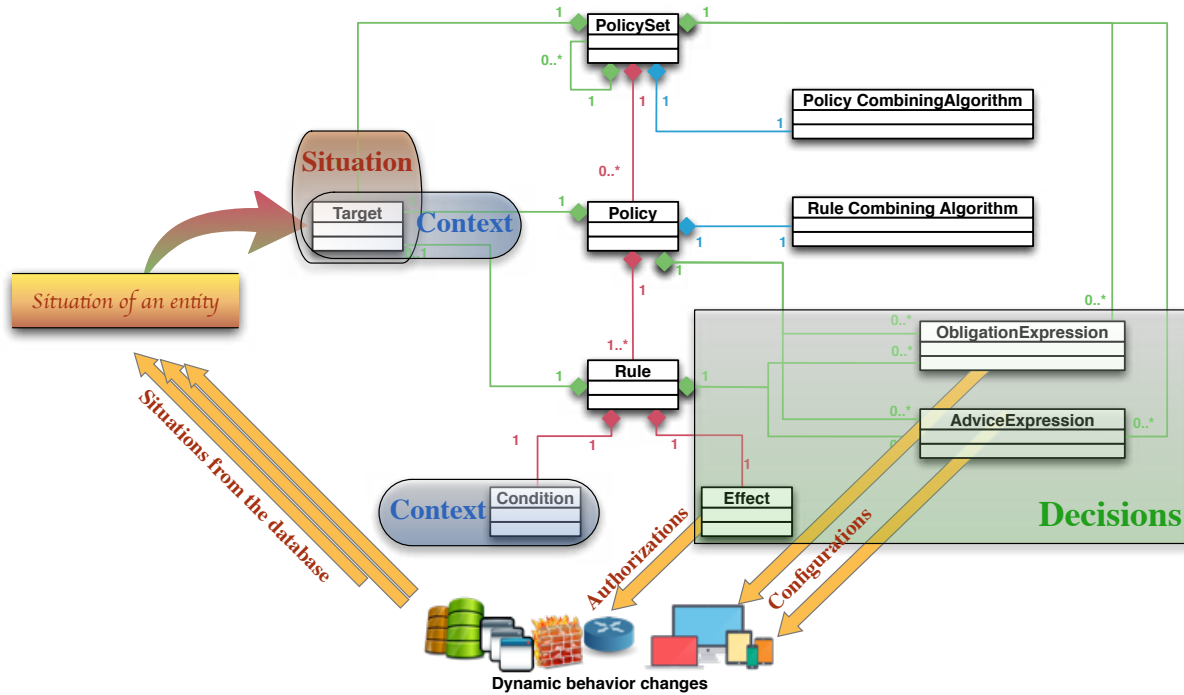


Figure 4.49: OASIS XACML v3.0 a security management policy oriented by situations

The situations influence gets to the XACML policy from one door, which is the *target*. The XACML security policy is composed of policy sets in which there are several policies. We define any policy set, policy, or rules to be targeted by situations. We have chosen the target as the place where to represent situations because the target has access to control three levels using the following XACML elements: the policy set, the policy and the rules. Hence, one situation targeting a policy set is then orienting the decisions of several policies. Another situation targeting a policy is then orienting the decisions of all its rules. Finally, one situation targeting a rule is then orienting its conditions. As a result, by putting the situations at the level of the target we can orient the XACML policy from all its possible levels.

Now, the context as we mentioned earlier represents the condition side of the policy. Therefore, we have two locations where we can represent conditions: inside the target and inside the rules' conditions. Representing the context at the target location specifies the situation. For instance, the context (*beginning of situation*) specifies that the situation "*reconstruction period*" has just started. On the other hand, we represent the context at the rule's condition location to specify in what condition the decisions should be delivered. For instance, the contexts ("*Name (Person) = Bashar*" and "*Name (Action) = Enter*") specify that only if the person's name is Bashar and the requested action is to enter, the decision of *allowing the access request* will be provided. As for the situation, it gives semantics for all contexts in both locations. For instance, the first context *beginning of situation* does not have sense if we don't know that the situation is "*reconstruction period*". Now, we understand that it is "*the beginning of the reconstruction period*". For the second context, allowing Bashar to enter does not have sense if we don't know that the situation is "*opening period*", which means that building 1R1 is opened.

Finally, the remaining element to represent using XACML is the *decisions* among the SCD triple. XACML has two elements that can represent the decisions: *Effect* and *Obligations*. We use the



*Effect* element to represent the authorization decisions, because the outcome of this element covers the possible authorizations: allow or deny. For configurations, we need to specifying: *who should do what*. Using the XACML obligation expressions we can specify what to do. For instance, the alarm system should be activated. However, we use the XACML advice expressions to specify *who* will do the obligation (i.e., by default, it is the requesting PEP who should apply the obligation).

As a result, we represented the following structure using XACML:

**When** Situation  
**And** Context  
**Then** Decisions

As a result of using XACML target, we are able to orient the policy at several levels. We are able now to represent our example's expressed SCD policy. Using XACML, the security management of the 1R1 building based on the specified security requirements will have the following representation:

**Policy:**

**Targeting** Situation (Building 1R1) = "Opening Period" **For**  
**Rule 1:**  
**IF** Name (Person) = Bashar & Type (Action) = Enter & Name (Building) = 1R1  
**Then** Permit  
**Targeting** Situation (Building 1R1) = "Reconstruction Period" **For**  
**Rule 2:**  
**IF** Role (Person) = Worker & Type (Action) = Enter & Name (Building) = 1R1  
**Then** Permit  
**Targeting** Situation (Building 1R1) = "Opening Period" **For**  
**Rule 3:**  
**IF** Situation-Status (Building 1R1) = "Starting-Point"  
**Then** Permit -- the continuity of this situation  
**Apply** Obligation: Deactivate the alarm system  
**Targeting** Situation (Building 1R1) = "Closing Period" **For**  
**Rule 4:**  
**IF** Situation-Status (Building 1R1) = "Starting-Point"  
**Then** Permit -- the continuity of this situation  
**Apply** Obligation: Activate the alarm system  
**Targeting** Situation (Building 1R1) = "Reconstruction Period" **For**  
**Rule 5:**  
**IF** Situation-Status (Building 1R1) = "Starting-Point"  
**Then** Permit -- the continuity of this situation  
**Apply** Obligation: Deactivate the alarm system  
**Targeting** Any Situation **For**  
**Rule 6:**  
**IF** Any Conditions  
**Then** Deny

The enforcement of the previous XACML representation allows the management of the 1R1 building's security. We managed conflicts using the strategy of denying all other situations that are not targeted in the policy (rule 6). By comparing the SCD policy expression with the XACML policy representation, we highlight the exact number of rules. We succeeded in proving the concept of our policy expression using situations, contexts, and management decisions (i.e., permissions and configurations).

In conclusion, our policy is a stable set of rules expressed in a flexible way that allows external services, namely decision-making, to move between the rules in a selective manner to give a dynamic security management decisions (i.e., changing the decision with regard to the environment behavior changes, which are expressed as situations). Stability implies not changing the rules. So that upon the start of a situation, or during its lifespan duration (e.g., “*opening period*”), the decision-making process will only start considering this situation whenever the policy evaluation takes place. This flexibility requires a generic expression methodology to express *situation-oriented rules*. That is why we choose to express the stable security policy using an attribute-based approach.

# Chapter's Conclusion

---

The Chapter presents the dynamic stable security policy to be conflict-free. The policy is written in a comprehensive way in order to cover all the desired security requirements. Being comprehensive to get flexible expressions means that the processing of conflicts has been considered at an upper level that is closer to organizations. As we are using situations as a convergence term linking the requirements with the policy and the behavior (i.e., aggregate events and rules), our contribution assumes that conflicts become a matter for these situations.

Nevertheless, it has been explained that having situations for the interest of security management defines them as conflict-free situations. That is, the identification of situations is concerned with judging an instantaneous access request in order to be authorized or not (i.e., with sending the appropriate configurations to apply the judgment decision). Therefore, the situations are only interesting to be consumed at the policy evaluation phase (i.e., at the access request moment). Hence, the situations are concerned about the current request for the current subjects to use currently available objects at the current time for the current environment. For this analysis, the conflicts are not subject to cause misjudgments of the security policy at the decision-making moment.

However, even if situations got conflicted in a future analysis when requirements change, the solving of conflicts at the high level (where the conflicts are semantically more understandable) is much easier in terms of management than going through, technically, the low-level rules or configurations.

We presented a dynamic security policy that is stable. We express the policy based on analyzing the system at the construction or design phase and following software engineering practices. Our stable security policy holds the expression of flexible SCD rules that give upon their evaluation (during an access request) dynamic decisions (i.e., authorizations and configurations). The dynamicity of our expression is provided thanks to the integration of the Situation Management paradigm. The technical implementation for SIMA was ensured through implementing an adaptable situation manager based on the complex event processing technology. Finally, the technical representation of our SCD expressed policy is ensured using XACML version 3.0.

Figure 4.50 summarizes our contribution on the policy expression in this chapter. We presented two edition processes, where both are based on the specified security requirements. The first one is to express and represent the policies using our SCD structure. The second is to express contexts in order to identify situations. Through the chapter we demonstrated at several sections our understanding of the triple relationship (Situation, Context, and Decision). We proposed the management of situation as a conceptual paradigm to manage, identify and define situations. We presented our implementation on software modules that process contexts to deliver the relevant situations for our stable policy. Finally, we represented technically both sides of edition.

Nevertheless, the yellow arrow in the same figure shows that the SCD policy considers the situations during the policy evaluation, but we did not present how? Therefore, there should be a technical relationship between the presented XACML policy and our implemented situation manager architecture.

In the next chapter, we demonstrate how we implement this relationship in order to implement the policy and enforce its decisions with consideration to different situations and contexts.

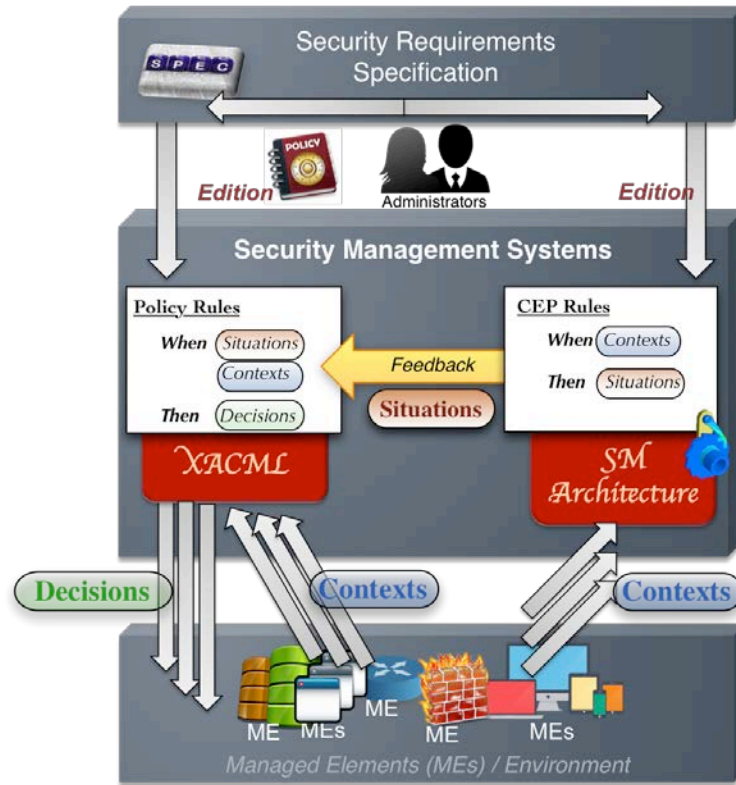


Figure 4.50: Concluding the policy expression contribution

# Chapter V

## Unified Adaptive Architecture

---

*The implementation of a security management policy entails always an architecture that applies its decisions (i.e., enforcing authorizations and setting configurations). Therefore, we contribute in this Chapter on implementing our SCD dynamic and stable policy with an adaptive and unified architecture. We intend to unify because we have different models to control a policy (outsourcing and provisioning). Thus, we need both models together in order to apply our SCD policy decisions (configurations and authorization). Moreover, the implementation of the situation management needs to be connected to the policy implementation. Also, we need to add adaptability to our architecture to perform dynamic reconfigurations. As the managed elements technology advances, their (re) configurations need to be dynamic and change frequently.*

*As a result, we use an event-driven architectural solution to unify the two policy control models and also the situation manager architecture. Moreover, we added to this architecture a component-oriented architecture that allows the dynamic (re) configurations.*

---

### Summary

- I. Interrelations implementation**
  - SM-PDP Relationship
  - PDP-PEP Relationship
  - Master PEP Conceptual Architecture
  - PEP-SM relationship
- II. The Implementation of a Unified Architecture**
  - An event-driven architecture
- III. SCD Unified & Adaptive Architecture**
  - SCD Unified Architecture
  - Adding the adaptability feature
  - Adding adaptability to agent implementations

# Chapter 5: Unified & Adaptive Architecture

*“The family is constantly changing, as each member changes. Some changes we recognize as developments, and the pleasure they bring usually makes us more adaptable. Some changes threaten, or disappoint other members, who may try to resist the change, or punish someone for changing.”*

Terri Apter

Columbia World of Quotations

Retrieved November 19, 2014, from Dictionary.com

## Introduction:

In this Chapter, we pursue our contribution on proposing a framework for the dynamic management of security. We succeeded in expressing the security management policy using the SCD structure:

<i>When</i>	<i>Situation</i>
<i>And</i>	<i>Context</i>
<i>Then</i>	<i>Decisions</i>

We represented this expression then using the 3<sup>rd</sup> version of the OASIS XACML standard language. Although we presented how to identify and manage situations and contexts using the Situation Manager (SM) architecture, we did not present the implementation of our stable SCD policy. Thus, we did not implement the consideration of the identified situations into the policy decisions. Studies state that the policy implementation entails two main steps: 1) the evaluation of the security management policy to make decisions, and 2) the enforcement of these decisions (i.e., applying the policy on the managed elements) (Sloman, 1994; Zhang et al. 2005; Mitropoulos & Douligeris, 2006; Bergstra & Burgess, 2007).

We presented through Chapter 3 the different policy management approaches. We found that the Policy-Based Management (PBM) approach meets these two steps by defining two management agents: 1) the Policy Decision Point (PDP), which is responsible of evaluating the policy and providing relevant decisions, and 2) the Policy Enforcement Point (PEP), which is responsible of enforcing PDP decisions into the managed environment.

Nevertheless, we draw our motivation of this Chapter in Figure 5.51. Even though we were able to identify and preserve situations to be consumed during the orientation mechanism, we did not present how? Therefore, we need to implement the following bidirectional relationships in order to complete our thesis contribution:

1. The **SM-PDP** relationship implementation: the SM architecture presented the last chapter is a set of independent software agents that works on preparing situations to be consumed by other agents (e.g., the PDP in our thesis work). Thus, the first direction of this relationship is **to implement** the SM architecture providing situations to the PDP in order to consider them during the decision-making process (i.e., the result of evaluating the SCD policy). The other direction is an optional one. The PDP may have the possibility of changing the situations to get back (or go further) to a more desirable situation. Therefore, we need **to implement** the PDP giving decisions to change the current situation(s) of one or different specific managed elements.

2. The **PDP-PEP** relationship implementation: there are different implementations in the literature for both PBM agents PDP/PEP (Boutaba, 2007). However, we need **to implement** different communication ways from the PEP to the PDP in order to provide the PDP with the necessary context (i.e., access-request-related) for the decision-making. As a result, we need **to implement** the PEP(s) to consider the situational decisions in both forms: authorizations and configurations.
3. The **PEP-SM** relationship implementation: the SM architecture ensures three functionalities where one of them is recognizing the context for the Starting Points and Ending Points. The PEP(s) may participate in creating the beginning or the ending of a situation. For instance, the SM architecture made the situation “*under attack*” of the managed network available for the PDP. The PDP gives the decision of applying the configuration “*block all accesses*” to the edge gateway at the managed network. Once the PEP applies this decision, the SM architecture defines a new situation for the edge gateway as “*closed*”. Hence, we need **to implement** the SM architecture to be able to reach the PEP(s) activities (i.e., applied security management decisions). The other way of the relationship is **to implement** how the PEP(s) can have the possibility of including the *beginning of situations* (i.e., the SP as a contextual information) inside their communication with the PDP. For instance, the PEP can include in its communication with the PDP that the status attribute of the situation “*opening period*” has just started in order to deactivate the alarm system (i.e., refer to the policy, p. 101).

Still, the above three relationships are strongly required to work in parallel, at the same time. These relationships as described complete one another, which means we cannot provide coherent decisions without relevant situations to be provided, we cannot apply effective decisions without considering the managed elements situations, and we cannot ensure the PEP to communicate with the PDP without the PEP having the necessary contexts. Hence, our final contribution in this thesis work is to unify all of the implementations for the three mentioned relationships, into a single security management architecture.

Giving the previous introduction, this Chapter is structured with the following sections: the first section presents in depth our propositions to implement the three relationships using the same scenario already used in the previous chapter. We provide henceforth a big picture on the required architecture to be implemented that unifies all the three relationships. Then we propose our implementation of this unified architecture. Finally we conclude our implementation by summarizing results and findings on the architecture side of our contribution.

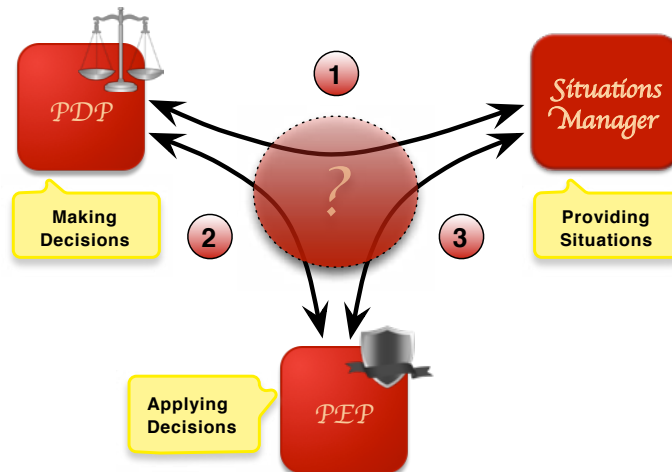


Figure 5.51: View on the necessary implementations to apply our SCD policy



# I. Interrelations implementation

At this point, our current possible options for technical implementations are the SM architecture, which we previously presented in Chapter 4, and the XACML technical PBM architecture that conforms to the XACML language version 3, which was also presented in Chapter 3.

These solutions can be adapted to participate in implementing only the first two following relationships (cf. Figure 5.51):

- i. In order to implement the SM-PDP relationship, we have to create a technical link between the policy evaluation process and the situation identification process.
- ii. While the XACML architecture ensures one PBM implementation, we have to ensure that the situation oriented decisions, which are the result of the previous relationship, are applicable using this implementation.
- iii. For the third relationship, we need to link the PEP configurations coming from the PDP obligations toward the situation manager (e.g., change the situation of a particular entity). Moreover, the Context Manager of the SM architecture can provide interesting contexts, not only to the Situation Manager, but also to be included in the PEP requests.

Therefore, we need to implement all these relationships and unify their functionalities to allow concurrent interactions between all our software agents.

## I.A. The SM-PDP Relationship

Given the orientation mechanism presented in Chapter 4, we demonstrated the need for situations at the moment of evaluating the SCD policy in order to provide a dynamic decision-making process. Hence, we link the XACML access control architecture to the SM architecture to ensure this need. It is essential to note that the work does not change the XACML architecture, and at the same time, the implementation respects the Amit abstract architecture.

Following our analysis of the XACML architecture, we found that there is one logical position where we can connect the two architectures. The PDP is interested about the semantics reflected by the situations happening. In other words, the PDP needs complementary information about situations of the managed elements to make decisions. Therefore, we found the Policy Information Point (PIP) is the most suitable point to connect both architectures.

As a result, we reuse the workflow of the architecture defined in Chapter 2 until the step 6 (Figure 2.8). Afterwards, the consumption of the identified situations starts. That is, when the context handler requests the attributes (i.e., situations with other attributes) from a PIP, in parallel, the SM should have already stored the interesting situations for the PDP inside the situation database. Once the situations are stored, the PIP returns the situations, and the PDP decisions will take effect according to the behavior. After fulfilling steps 12 and 13, the SM detects the new changes (if any) and identifies the new situations caused by the PDP decisions. Then, the SM stores them to be taken into consideration when arriving at step 6 again.

## I.B. The PDP-PEP Relationship

Following this combination of both architectures in Figure 5.51, we clearly find that the PDP-PEP relationship is partially considered. The XACML architecture ensures only the authorizations decisions through the request/response standard and synchronized message exchanges.

Nevertheless, we find that implementing the PDP-PEP relationship with consideration to the SCD decision defined in the previous chapter must include three possibilities of exchanges: providing context, applying security management decisions, and decisions requests/responses.

### I.B.1. Providing the Relevant Contexts

In Figure 5.52, the PEP agent is responsible of informing the PDP about the relevant contexts occurring inside the managed elements. We highlight two words: relevant and contexts. Given the SCD policy example in the previous chapter, we have three situations: opening, closing and reconstruction periods. The configurations expressed and represented for this example are applied once the beginning of each situation is detected. Therefore, we need the PEP to inform the PDP about the beginning of each one of the three situations (i.e., the relevant context) once it happens. Upon their happening, a relevant configuration should be applied: activating or deactivating the alarm system (i.e., the applying PEP agent is not necessarily the one sensing the context). We call the PEP agent at this part of the relationship a *sensor*.



Figure 5.52: PEP agent as a sensor

### I.B.2. Asynchronized Security Management Decisions

In Figure 5.53, the PEP agent is also responsible of applying any received decision from the PDP. Hence, the PEP should know and be able to apply this decision. We call such agent an *actuator* that acts by applying the decision once it arrives (i.e., asynchronized). The sent decisions from the PDP to the PEP are often called configurations (e.g., activate/deactivate the alarm system, configure the edge gateway to block all accesses, update the current authorization policy with new permissions, etc.). We call *provisioning* the action of sending the configuration from the PDP. It is requesting the PEP to perform the action of ‘*configuring*’ different managed elements.

However, in order for the PDP to react by giving new security configurations, another (or the same) PEP should work as a *sensor*. Thus, the PDP first receives events expressing certain context from any PEP that works as a *sensor*. Upon these context changes, the PDP evaluates the SCD policy and sends relevant decisions to the targeted PEPs that work as *actuators*. Then, these PEPs are expected to perform whatever required actions expressed in the SCD policy (i.e., we consider that the targeted actuators already understand these actions and holds their technical implementations).

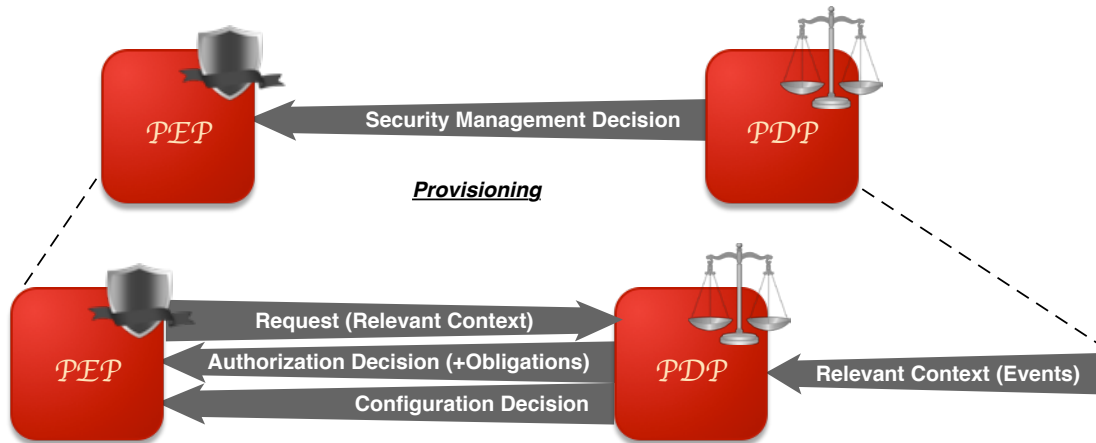


Figure 5.53: PEP agent as an actuator

### I.B.3. Synchronized decisions requests/responses

In Figure 5.54, the PEP agent is also responsible of sending the access request manifested by a user and synchronously waits the decision as a response of the access request. We see this agent as a node of synchronized communication and we call it a *smart node*. Upon the receiving of a special kind of events (i.e., that we call an access request) from the PEP, the PDP evaluates the relevant context(s) accompanied with the event and returns a decision. We defined the decision as an authorization decision reflecting the permission and/or an obligation decision reflecting the necessary configurations to be provisioned (i.e., before or after the authorization decision).

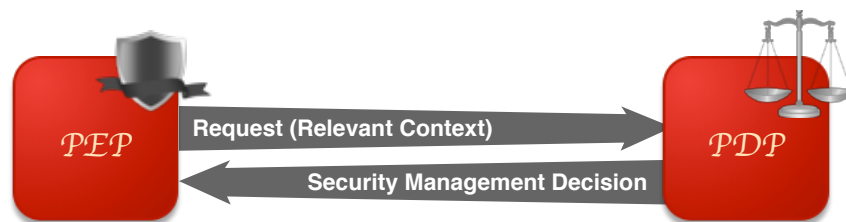


Figure 5.54: PEP agent as a smart node

## I.C. Master PEP Conceptual Architecture

Although we obtained our new combined architecture (cf. Figure 5.55) that supports the situations orienting mechanism, we only ensure the enforcement of a dynamic security policy in terms of authorization decisions only (i.e., XACML synchronized authorizations request/response). To meet our security management requirements, we are required to include and implement security configurations along with these dynamic authorizations. Thus, we need to implement and ensure the three previous functionalities: providing the relevant contexts, applying security management decisions, and ensuring decisions requests/responses.

Now, XACML only supports the smart nodes implementations through its standard request/response communication paradigm. Therefore, all XACML PEPs can be only smart nodes. However, we want to keep using the standard and thus make use of the XACML communication paradigm, but we want also to implement our presented dynamic security management system.

Therefore, we contribute in implementing a software agent that we call “**Master PEP**”. This agent is able to accept all three communications presented earlier, and therefore perform the three

required functionalities. Hence, our *Master PEP* can behave not only as a smart node but also as a sensor and an actuator.<sup>25</sup>

In Figure 5.55, we present our Master PEP architecture. This agent is able to perform functionalities 1, 2, and 3 and therefore be a sensor in 1, an actuator in 2, and a smart node in 3. Hence, the PDP receives the situations through the PIP and use their semantics to know where to orient the request once it arrives inside our stable and dynamic SCD policy. When the Master PEP sends a normal XACML request, the PDP synchronically responds with a response after evaluating the policy. The evaluation is oriented automatically by the provided situations. This part is original with innovation at the PEP agent level.

However, when another PEP detects a relevant context (e.g., an intrusion), the latter forwards the context to the Master PEP. **The Master PEP by its role redoes the original part. Now, the received decision of the PDP to the Master PEP might or might not correspond to the same PEP that informed about the new context (i.e., works as a proxy).** In case of the positivity, the communication between the Master PEP and the PEP is then synchronized. Otherwise, the communication is asynchronized. As a result, the Master PEP itself manages the synchronized communication and this produces functionality 3 (i.e., the Master PEP is a smart node). The asynchronized communication ensures the functionalities 1 and 2 (i.e., the Master PEP becomes either a sensor or an actuator).

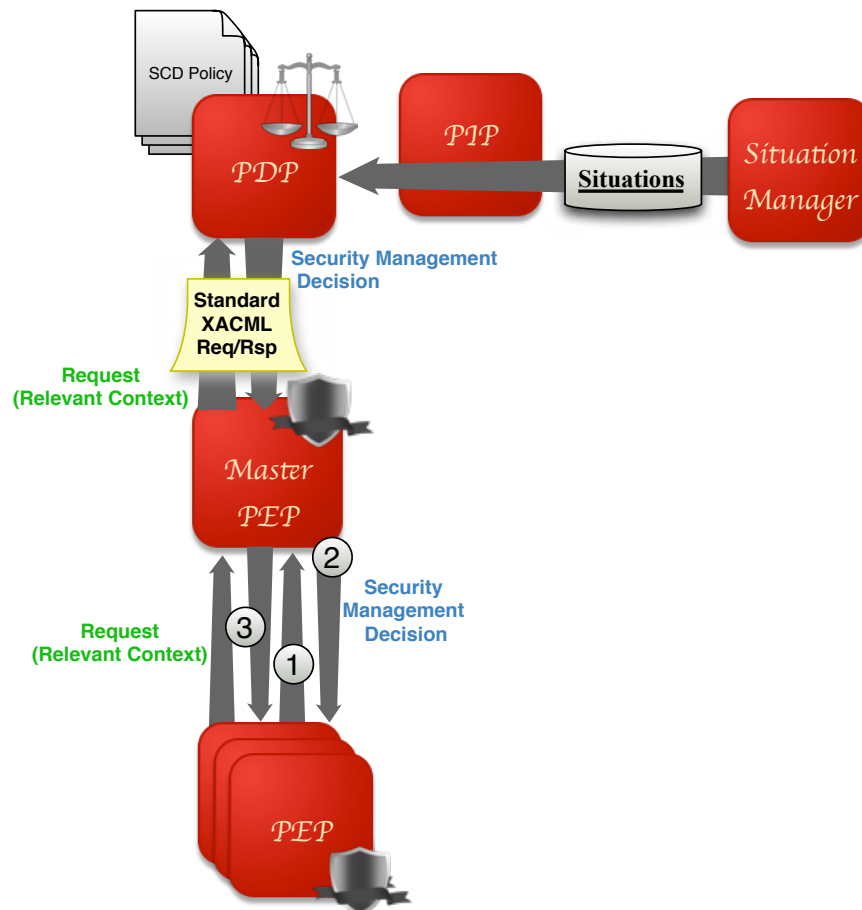


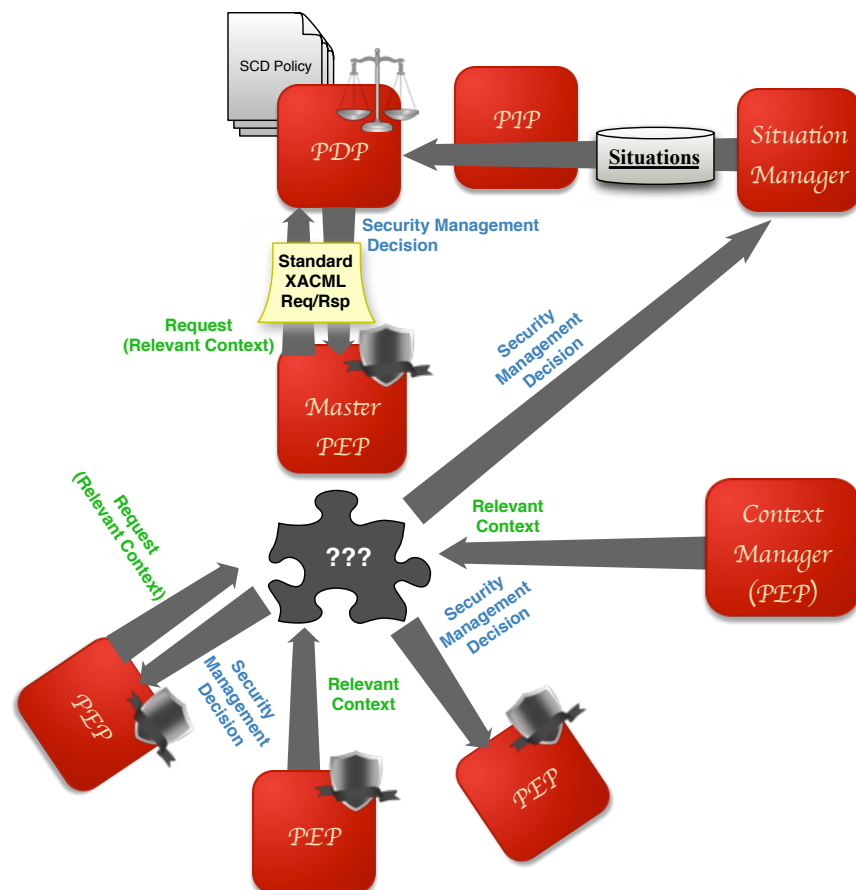
Figure 5.55: Conceptual architecture of a Master PEP

<sup>25</sup> We present the technical implementation of the Master PEP inside the upcoming underlying sections.

## I.D. The PEP-SM relationship

We noticed that the Context Manager might also recognize certain common contexts (such as attacks, intrusions, etc.) in order to define the SPs and EPs. Hence, the context *beginning (or the ending) of situation* can be seen as a PEP **sensor** to inform the PDP (e.g., the beginning of the opening period in our SCD policy). As a result, we figured that the Context Manager works as a PEP in this case for the Master PEP (i.e., works as a sensor). Moreover, the Master PEP (or the Context Manager PEP) may apply configurations from the PDP in order to change the situation of certain entities (i.e., we can imagine some entities with high security-sensibility or privacy-sensibility, therefore their situations need a permission in order to be changed). So, the Context Manager receives the Master PEP configuration of obligating the SM to change the situation of a specific managed element, and it transfers this configuration to the SM in form of SP or EP (we also accept that the SM works as a direct actuator in this case and considers the configurations as an SP or EP). Thus, we need to find a convergence technique to unify all types of agents: sensors, actuators and smart nodes.

In Figure 5.56, we represent our last contribution as a missing puzzle piece. This puzzle piece we seek should unify the management of security using our SCD policy through the three mentioned agent types. At the same, the unifying solution should allow decoupling the agents by permitting many communications at the same time, and therefore be able to organize all the exchanges. Thus, next section presents our implementation of such architecture.



**Figure 5.56: The missing puzzle piece of our Master PEP architecture**

## II. The Implementation of a Unified Architecture

We want our architecture to implement, but not limited to, the scenario we used in the previous chapter. In Figure 5.57, we present how our earlier expressed and represented SCD policy (i.e., during the two edition processes: policies and situations) should be applied and implemented.

Therefore, we need our architecture to be able of:

- Having synchronized a request/response authorization from/to Bashar.
- Provisioning asynchronized configurations to activate/deactivate the alarm system.
- Exchanging contexts between the different parties: Bashar, alarm system, context manager, master PEP, and the 1R1 Building.

As a result, we need to find the technology that allows us to cover the following points:

- Implement different types of agents (mainly, sensors, actuators and smart nodes), and permits a unified environment where all types can exist together. Moreover, this implementation should permits to implement the grouping of all agents' feature into one agent (i.e., Master PEP).
- Implement all different types of mentioned communications: provisioning (relevant contexts and configuration decisions) and outsourcing (access requests). Moreover, the implementation should allow the necessary representation for these exchanges.

Therefore, the following underlying sections are seeking to implement a convergence architecture that allows covering the two previous points.

Finally, we contribute with adding a new value to our unified architecture. The last subsection of this section proposes a dynamic methodology of implementing security management configurations.

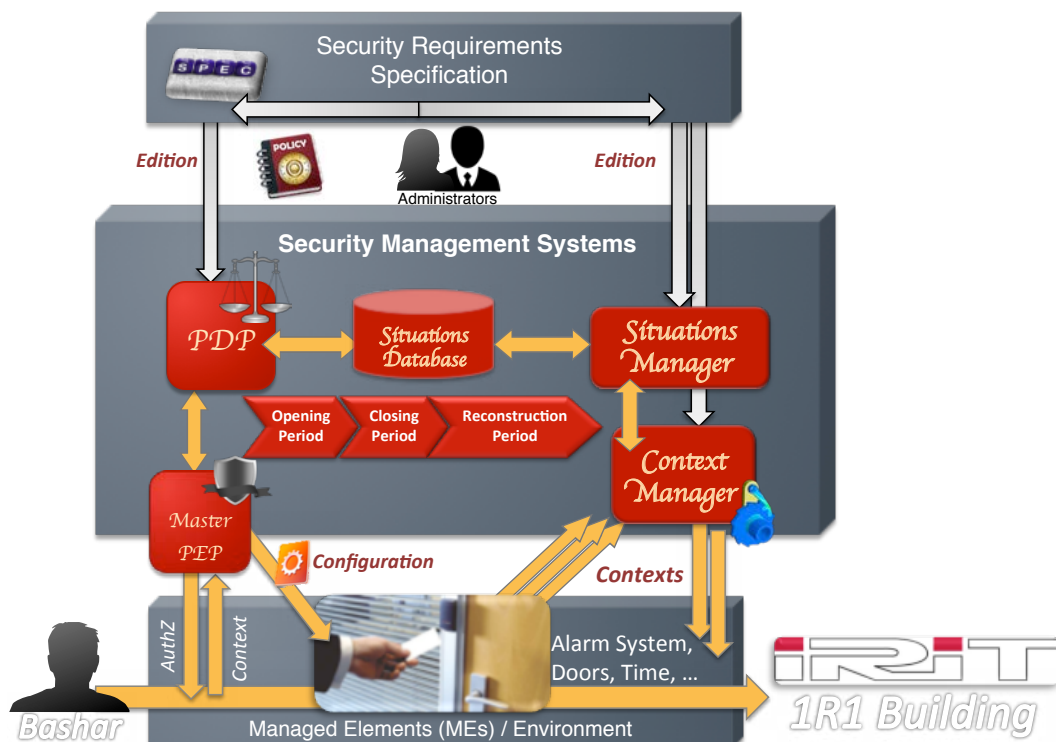


Figure 5.57: Conceptual example of a unified architecture



## II.A. An event-driven architecture

An event-driven architecture disseminates to relevant agents (human or software) what events are happening inside or around the technical managed environment. The relevant agents evaluate disseminated events, and may take actions if necessary. Therefore, it is crucial for the situation manager to be aware of the contexts (i.e., simple and complex events) so that SM can apply semantics to their occurrences. We used the events stream technology to receive and send all relevant events. As we are already using an event-driven architecture for the situation manager, we want to optimize our contribution by integrating this usage to be part of one architecture. But why?

We used the events stream to group contexts into one place where several agents can select relevant contexts to their operations (i.e., the context manager, the situation manager, and the Master PEP). We indicated that there are some contexts that are relevant for the Master PEP (e.g., beginning of the opening period) and others that are interesting for the context manager (e.g., the number of access requests per 5 minutes). Moreover, we indicated that there are different agents communicating with each other using different communication manners. Therefore, we need a converging solution that could 1) integrate all these agents and communications 2) and employ the streams of events as well into a unique unified solution.

We found that the *enterprise service bus* (ESB) technology meets these expectations. The ESB objective is to exchange messaging with wide acceptability of differences in used technologies and communications. ESB aims conceptually at grouping all different event-based agents and allowing the usage of each other services on-top of a unified way of communications (Etzion & Niblett, 2011, Pages 16, 308, 310; Chen, 2006). Therefore, this is what replaces conceptually the missing puzzle in Figure 5.56.

Although the concept is abstract, we can specify more our requirements and our technical implementation for a specific ESB. Thus, we present our implementation by answering the following questions, where each one is a separated subsection:

1. How to implement the three types of agents?
2. How to implement the different communication types between them?
3. With the situation manager architecture event streams and the bus technology, how to organize and structure the exchanging messages mechanism?

Nevertheless, we start first by going more in-depth about “what is the *enterprise service bus*?”

*Event-driven architectural styles* free the event notifications from having to be sent to a specific destination (component, agent, etc.). In security management, the *enterprise service buses* (ESBs) are event-based systems that aim at loosening the coupling of space, time, and synchronization and at providing a scalable infrastructure for information bus exchange. These systems are driven via event subscriptions and streams/channels. Events streams are technically provided through two types: queues and topics. The relationship between the architecture components is defined through subscriptions. The parties in this relationship are either producers/consumers and/or publishers/subscribers.

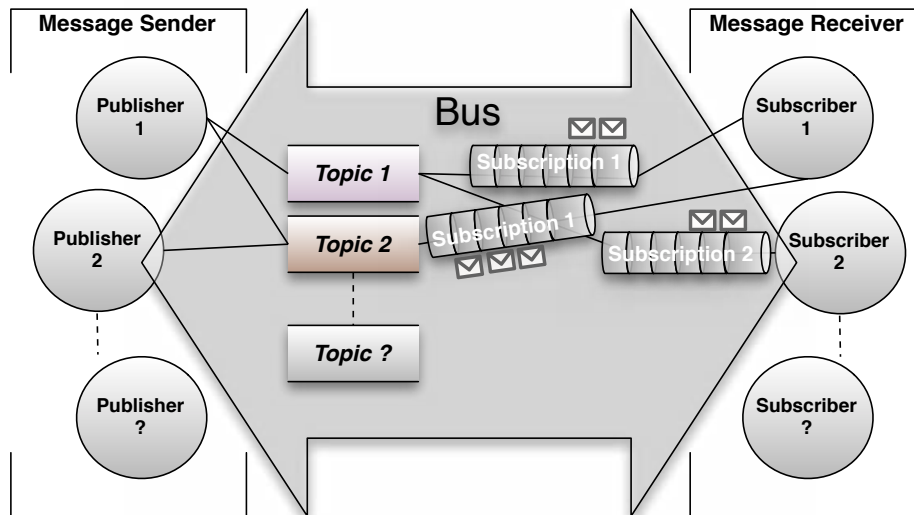
The Producer/Consumer model is used when the producer needs to provide a service to a preregistered consumer. Therefore, there should be a sort of contract or engagement between both parties in the form of peer-to-peer to guarantee this service.

However, the Publisher/Subscriber model is used without a predefined contract. It is sufficient for a subscriber to be interested in a channel of events, and then subscribes to start receiving



notifications from the publisher (Mühl, Fiege, & Pietzuch, 2006, p. 3). Channels in publishing/subscribing technique are either structured based on their events content (e.g., medical information) or classified based on their topics (e.g., policy instructions, sensor readings, etc.).

We chose to use topic-based channels because they result in creating fewer event streams for our architecture (see Figure 5.58). Moreover, they are decoupled from the concrete scenario we are implementing. At the opposite, the content-based channels create multi-channels and require preliminary awareness of the events content we are processing.



**Figure 5.58: Topic-Based Enterprise Service Bus**

In our architecture, each publisher should send to a specific preconfigured topic. Each topic is associated with many subscriptions, where each is related to only one subscriber. For instance, Publisher 1 can publish a message (i.e., contains an event) to topic 1. The message is placed in all subscriptions queues (i.e., subscriptions 1 & 2). The topic 1 subscribers can consume the message based on retrieving rules that they specify (i.e., message filtering and processing rules). For instance, the rule could be to consume the received message only if it is from publisher 1. However, both subscriptions 1 & 2 contain the same number and order of messages.

Finally, the power of an event bus in the ESB-based system is their conceptual flexibility (Schmidt, Hutchison, Lambros, & Phippen, 2005). Thus, there is no restriction in term of connections count. Therefore, an agent may have several connections to the bus. Moreover, no limits in term of agents number connecting to the bus. Hence, we need to manage the bus system with respect to the infrastructure limitation (i.e., in order to maintain the expected performance). In addition to managing the bus connections, exchanging messages should be universal for all parties. That is, inside the bus every agent type has different exchange reason (configuration, authorization, etc.), so we have to unify the representation of messages to be understandable by all these parties. Hence, we need to define a common language to exchange messages, but with conformity to our complex event processing in order to accept our contexts representations (see Chapter 4). However, the common language is only to represent our messages content, but the content is always an event. Moreover, as our architecture is an event-driven one, receiving and sending a message is an event.

## II.A.1. Agents Implementations

In Figure 5.59, we implement our three identified types of agents as the following:

- 1) **Sensors:** software agents configured to monitor contexts and publish their observations to be consumed by the other agents. Therefore, the sensors are implemented to work as *publishers* into specific topics of our bus.
- 2) **Actuators:** software agents able to apply the policy decisions by performing actions (i.e., configurations, etc.) on the managed elements. In order to receive these decisions, the agents need to listen by monitoring the policy decisions. Therefore, the actuators are implemented to work as *subscribers* to specific topics of our bus.
- 3) **Smart nodes:** software agents configured to monitor relevant contexts (namely, access requests) and send them in form of authorization requests. Then, they are required to apply the decisions coming in form of authorization responses from the PDP. These agents are implemented to work as both *publishers* and *subscribers*. However, these agents follow the synchronized communication paradigm that we defined as *outsourcing*. Therefore, we need to maintain the relationship between publishing and subscribing. So, we implemented the smart nodes and developed each one to consider a sort of *session-like* solution. After each time the smart node publishes an access request, the smart node needs to be maintained a subscription session to receive the response to this request. Thus, we create synchronized communications with the bus.

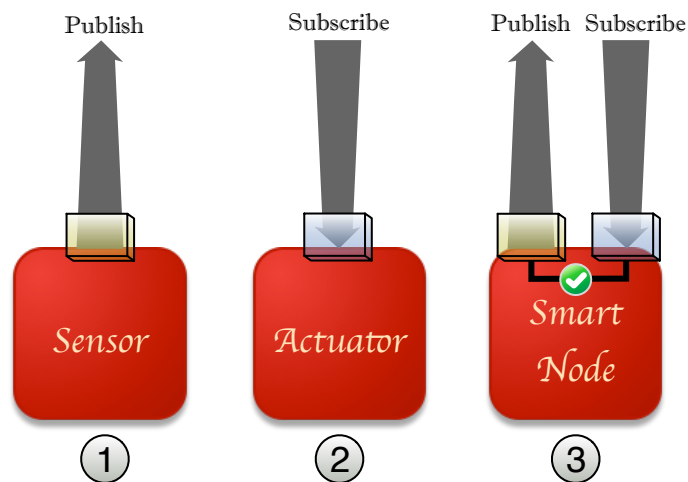


Figure 5.59: Implementation of the three agent types.

We still have to implement our Master PEP agent. This agent as presented conceptually in Figure 5.55, ensures that security management decisions are provided in both a synchronized (outsourcing) and an asynchronized (provisioning) communication paradigm. Therefore, Figure 5.60 shows our implementation by combining all the three previous implementations. The Master PEP needs to subscribe to the bus topics in order to 1) receive access requests, and to 2) receive relevant contexts. Moreover, the Master PEP needs to publish to the bus topics in order to 1) provision directly policy decisions, and to 2) respond with authorization decisions to the waiting smart node.

Now as we implemented our agents, we need to specify the implementation for the communications between them. In next step, we present different possible communications and their implementations.

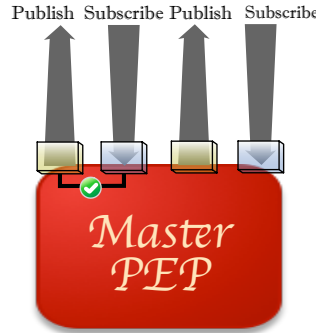


Figure 5.60: The Master PEP Implementation

## II.A.2. Inter-Agents Communications

In Chapter 3, we presented during the architecture state-of-the-art two policy control models to describe the communication between the PEP and the PDP: outsourcing<sup>26</sup> and provisioning<sup>27</sup>. We argued that any security management system is required to accept both models in order to perform configurations and authorizations.

Therefore, in our contribution, we implement three communication modes between agents: outsourcing to perform authorizations, provisioning to perform configurations, and finally we contribute with a hybrid model for both configuration/authorization at the same time. We present each communication mode using our same example presented in Chapter 4.

In Figure 5.61, we present our implementation for the outsourcing communication paradigm, which is a synchronized model of controlling our SCD policy. After the SM identifies the situation “*opening period*”, the PEP agent fixed on the door (i.e., it should be able to control its movement through, for instance, a badge system) publishes requests to the common topic with the Master PEP, whenever a person (Bashar) approaches to the door and presents his/her badge card. Through its subscription to the same topic, the Master PEP receives Bashar’s access request and performs a standard XACML outsourcing communication (i.e., request/response) with the PDP. Once the PDP gives the situation-oriented decision (i.e., influenced by the “*opening period*” situation, which is obtained by the PIP), the Master PEP publishes the authorization decision (i.e., allowing Bashar to enter building 1R1) to a common topic. Through subscribing to the latter topic, the PEP smart node agent (i.e., controlling the door system) gets the response, which holds the authorization decision. Finally, the PEP enforces the PDP decision of allowing Bashar to enter by opening the 1R1 building door.

As a result, we were able to implement the authorization of the SCD policy, such as the rule:

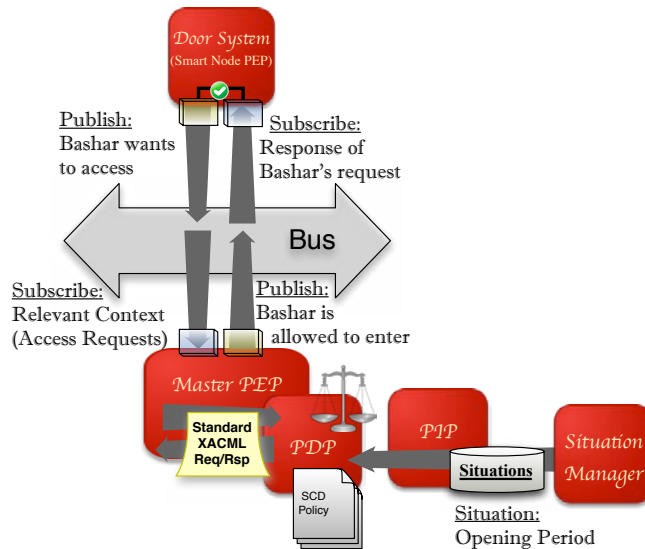
**Targeting** *Situation (Building 1R1)* = “*Opening Period*” **For**

**Rule 1:**

**IF** *Name (Person)* = *Bashar* & *Type (Action)* = *Enter* & *Name (Building)* = *1R1*  
**Then** *Permit*

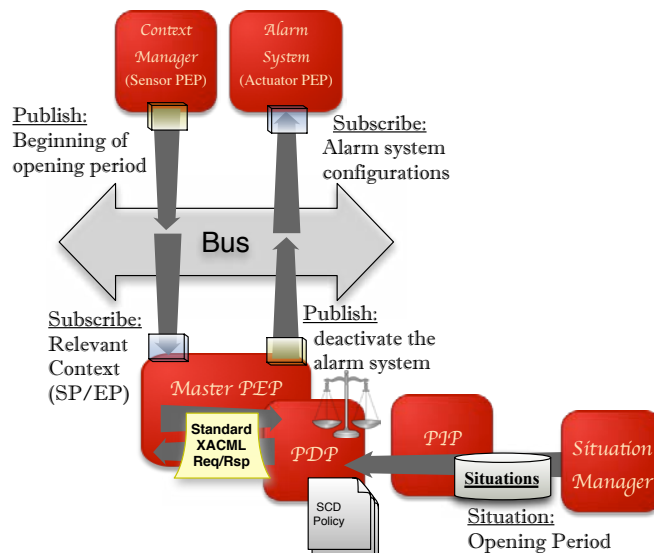
<sup>26</sup> Work on outsourcing presented in COPS (Durham et al., 2000, RFC 2748; “COPS-ODRA/DRA” Salsano, 2001 & 2002))

<sup>27</sup> Work on provisioning presented in COPS-PR (Chan et al., 2001, RFC 3084))



**Figure 5.61: The outsourcing SCD policy control model implementation**

In Figure 5.62, we present our implementation for the provisioning communication paradigm, which is an asynchronized model of controlling our SCD policy. After the Context Manager<sup>28</sup> (i.e., PEP as a sensor) publishes the beginning of the “*opening period*” situation to a common topic between the Situation Manager and the Master PEP, the Situation Manager agent identifies and stores the situation “*opening period*”. Through its subscription to the same topic, the Master PEP receives the context (“*opening period*” SP) and **performs a standard XACML outsourcing communication (i.e., standard XACML request/response paradigm) with the PDP**. Once the PDP gives the situation-oriented decision (i.e., influenced by the “*opening period*”), the Master PEP publishes the configuration decision (i.e., deactivate the alarm system) to a common topic. Through subscribing to the latter topic, the PEP actuator agent (i.e., alarm system controller) gets the configuration as an obligation to do. Thus, this PEP actuator calls the program that deactivates the alarm system.



**Figure 5.62: The provisioning SCD policy control model implementation**

<sup>28</sup> The Context Manager is a smart agent that publishes and subscribes to the messages bus. It subscribes to events and publishes new complex events as new contexts, SPs, and EPs.

As a result, we were able to implement the configuration of the SCD policy, such as the rule:

**Targeting** *Situation (Building IR1) = "Opening Period" For*

**Rule 3:**

**IF** *SP (Situation) = "Opening Period"*

**Then** *Permit -- the continuity of this situation*

**Apply** *Obligation: Deactivate the alarm system*

Finally, we present our implementation for the hybrid communication paradigm, which combines both the outsourcing and the provisioning communication paradigms and we use synchronized and asynchronized models for controlling our SCD policy.

Supposing that the security checks to authorize Bashar of entering the building is verified at two doors, and not only one as in Chapter 4. Hence, we can facilitate the entering of Bashar by controlling the access once (i.e., at the main entrance 'N1'), and then configuring the next door (i.e., the Lab door 'N2') in the way to his/her office. As a result, we need to implement the hybrid side of the following SCD policy rule:

**Targeting** *Situation (Building IR1) = "Opening Period" For*

**Rule 1:**

**IF** *Name (Person) = Bashar & Type (Action) = Enter & Name (Building) = IR1*

**Then** *Permit*

**Apply** *Obligation: Open the next door (N2)*

In Figure 5.63, After the SM identifies the situation "opening period", the PEP agent fixed on the door (i.e., it should be able to control its movement through, for instance, a badge system) publishes requests to the common topic with the Master PEP, whenever a person (Bashar) approaches to the door and present his/her badge card. Through its subscription to the same topic, the Master PEP receives Bashar's access request and performs a standard XACML outsourcing communication (i.e., request/response) with the PDP.

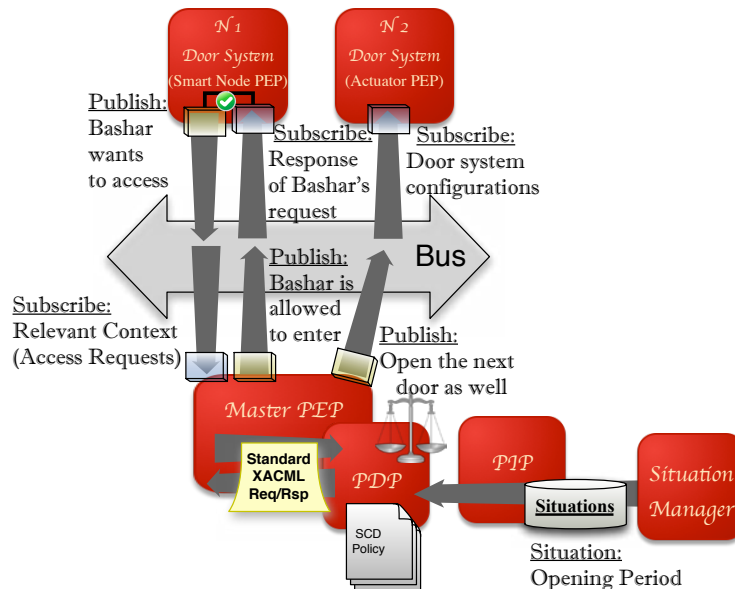


Figure 5.63: The hybrid SCD policy control model implementation

Once the PDP gives the situation-oriented decision (i.e., influenced by the "opening period" situation, which is obtained by the PIP) this time consists of both configuration and authorization, the Master PEP publishes to a common topic the decision, we will explain in the next step how the Master PEP identifies the destination and how to divide the security management decision. Through

subscribing to the later topic, the PEP agent (i.e., as a smart node controlling the system of the door N1) gets the response, which holds the authorization decision. This PEP enforces the PDP authorization decision of allowing Bashar to enter by opening the 1R1 building door N1. At the same time, through subscribing to the same topic, the PEP actuator agent (i.e., controlling the door system N2) gets the configuration as an obligation to do. Thus, this PEP actuator calls the program that opens the door (i.e., until Bashar passes through N2).

Nevertheless, these exchanges between agents are realized based on topics, as we explained earlier. Hence, we have to manage and define these topics. Moreover, we have to organize and structure the use of each topic and the messages circulating inside each one. Therefore, we still need to answer in the next subsection to the following remained questions:

1. What are the used topics, the relationship between the topics and agents?
2. How do we define topics and configure agents to know what topic to use?
3. What are the types of messages permitted inside the topics? Who can retrieve each message?
4. How do topics link agents between each other? How does each agent discover the other agent?
5. How does the Master PEP agent analyze the XACML PDP responses and send each decision to the suitable agent?
6. How to know which message goes to which agent and how to organize the exchanges? How to know how to read the structure of the message ?
7. How does the Master PEP agent know the other existing PEP agents and which PEP is responsible for which managed elements?

### **II.A.3. Managing Agents Exchanges**

We defined the different types of agents and how we implemented each type of them. We also presented the different communications possibilities between these agents. Now, we need to explain on what these communications are built on. Therefore, we need to explain how the agents are exchanging information and how we manage and implement these exchanges.

Whenever we implement an agent, we download a configuration file that contains 1) the available topics names, 2) the description of each topic, 3) and finally the address for connecting to the bus (i.e., for publishing and/or subscribing). However, how we define topics?

A topic means “*a matter dealt with in a text, discourse, or conversation; a subject*” (Oxford Dictionary, British & World English). Technically speaking, a topic-based communication is a modular approach towards a content creation, where content is composed around specific topics. Within a topic, we can mix contents and reuse them in different common contexts. Thus, we use a topic to unify several contents, our communications are done based on several topics, and we use the bus to unify topic-based communications into a unique event-driven architecture.

Therefore, each functionality we provide treats a problem, and each problem needs a topic. Thus, we define the main functionalities that our unified architecture provides: the processing of complex events (i.e., to calculate the SPs and EPs) and the making of security management decisions (i.e., sending authorizations and configurations). Thus, we have mainly two topics used: *complex events* and *policy decisions* topics.

The *policy decision* topic is used to exchange authorizations (including requests) and configurations (i.e., security management decisions). Thus, the participating agents in this topic are mainly:



- 1) The Master PEP as a publisher for security management decisions and as a subscriber for new authorization requests
- 2) The smart node PEP agents as publishers for authorization requests and as subscribers to receive the security management decisions
- 3) The actuator PEP agents subscribes to this topic waiting for configuration obligations coming from the Master PEP, and finally
- 4) The Context Manager (or directly through SM) agent as a subscriber for receiving policy obligations about changing the situations status (start or end)

The *complex events* topic is used to exchange detected contexts. We defined the context as the processed events (e.g., motion detection sensors detect 0 person in the room *alpha*, the context is the room *alpha* is empty). Thus, we identify the participating agents to be:

- 1) The Context Manager agent as a publisher to this topic for contexts in form of SPs/EPs for both agents the SM and the Master PEP
- 2) The SM then working as a subscriber for this topic to identify situations
- 3) The Master PEP working as a subscriber for this topic to obtain relevant contexts with the policy, for example, the startings/endings of “*opening period*”.

Nevertheless, the processing of the complex events requires applying operations on the simple unprocessed events (e.g., sensor readings). Hence, we need another topic where all technical sensors can send their *raw events* in order to be processed by the Context Manager. For instance, in the 1R1 building, we may have motion detection sensors associated with the alarm system. Thus, the readings coming from these sensors are not processed yet, and therefore are sent to the *raw events* topic.

We mentioned earlier that the Master PEP needs to know the available other PEP agents (sensors, actuators, and smart nodes). Thus, we need a sort of “social milieu” to exchange identities of each connected agent<sup>29</sup>. Therefore, we use a similar approach to the COPS-PR management approach in recognizing connected PEPs. That is, the PEP initiates the connection with the PDP only once at the first time, and then, the PDP starts provisioning the relevant configurations to this PEP whenever needed.

Likewise, all of our agents must publish to our last topic (called “*Connections initiation*”) at least once. The published message contains at least two attributes and their two values: the PEP ID and its functionality(ies). For instance, the *alarm system* actuator agent publishes once to the initiation topic that its ID is the *ActPEPI* and its first functionality *AlarmControl* (i.e., it is possible to have more than one functionality). The Master PEP retrieves all these information by subscribing to this topic and stores them as a form of list (each PEP and its information). The Master PEP maintains its subscription to the topic active as long as it is running in order to be updated whenever a new PEP agent is connected.

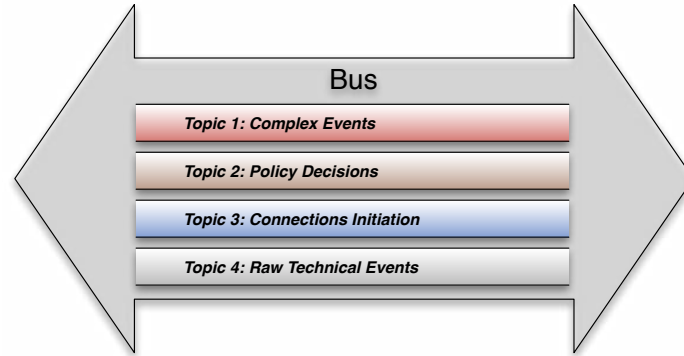
Finally, the Master PEP keeps in touch with registered PEPs once a security management decision concerns one of them. In case the decision contains an authorization, the Master PEP replies by publishing to the PEP who first sent the request. In case the decision contains configurations, the Master PEP browses its list and matches the configuration with the available functionalities. Once a match is found, the Master PEP asks the matched PEP to perform the configuration. But how?

---

<sup>29</sup> Ideally, we have to manage the identities of agents. Thus, we have to verify that an agent has the identity it publishes. Hence, we need to apply mechanisms such as the Public Key Infrastructures (PKIs). However, this is not in the scope of this thesis and we will explain it in a future work.



Now, we have four topics used in our architecture illustrated in Figure 5.64. However, we still need to present how we manage the exchange of messages in term of: knowing who sent the message, how to recognize the messages elements, and when to know that a message is relevant to an agent. That is, all agents subscribing to a topic will receive all messages. However, they can write their own subscription rules to filter messages and get what is interesting for them.



**Figure 5.64: Our available topics for the messages bus.**

There will be many messages (e.g., configurations) sent from the Master PEP to the other PEP agents. Thus, we need to define a structure for the messages in order to know how to recognize the messages elements. Moreover, with numerous configurations sent, the PEP agents need to know which one of them need to perform a retrieved configuration.

In Figure 5.65, we answer to all the remaining questions with concern to the exchanges of our agents. First, all our topics in the bus can only contain text messages. Topics are queues that hold a stream of messages (i.e., known as channels) sorted by the arriving order. Thus, we need to explain how we exchange events inside text messages.

We unify the message format and provide a formal structure. Therefore, any publishing and subscribing operation needs to respect this structure. Moreover, defining a format for messages permits controlling and securing the exchange. So, we identify each message by identifying each source agent who is sending the message.

All software agents of our contribution are connected to the Message Bus technology to exchange events (i.e., implemented using ActiveMQ Bus). We define the following standard Backus-Naur Form (BNF) specification to be respected by all messages<sup>30</sup>. Each event is therefore structured in a MESSAGE as following:

```

<Message> ::= <ID> <Event>

<ID> ::= <Sender> [<Receiver>]
    <Sender> ::= < Single-Attribute >
    <Receiver> ::= < Single-Attribute >

<Event> ::= <Attribute-List>
    <Attribute-List> ::= <Single-Attribute> [<Attribute-List>]

<Single-Attribute> ::= <Attribute-Name> <Attribute-Value>
    <Attribute-Name> ::= function
                        | sender-ID
  
```

<sup>30</sup> IETF standards: RFC733, RFC822, RFC2616 & RFC2811.

For more information <http://marvin.cs.uidaho.edu/Teaching/CS445/grammar.html>

```

| destination-ID
| configuration
| authorizationDecision
| ...
<Attribute-Value> ::= ActPEP1
| AlarmControl
| Activate
| Deactivate
| Allow
| Deny
| ...

```

Therefore, each topic contains messages that have two elements: ID and Event. The ID is the identification of the message and it is composed of both the sender and the receiver (destID) identities (i.e., the destination ID is not a required field in the message, as sensors for example are not required to know who will use their readings). On the other hand, the event is structured in form of sequences of an attribute and its value.

An example on the messages that are sent to the connection initiation topic only once from each new agent:

**<Message>** ::= senderID ActPEP1 function AlarmControl

The following message is an example on a normal message. It is sent from the Master PEP to the alarm system in order to deactivate its functionality:

**<Message>** ::= senderID MasterPEP destID ActPEP1 configuration deactivate

We defined the allowed structure and content of our messages. Now, we need to present how an agent knows that a sent message into one of its subscribed topics needs to be retrieved. To do so, we use the ID value to filter the messages in a subscription relationship through rules. Thus, the rules allow the agents to retrieve only interesting messages from the topic queue.

In Figure 5.65, we illustrate an example on the management of exchanges. The Master PEP publishes to the topic “*policy decisions*” three types of messages (i.e., for the Master PEP these messages are decisions and for the receiving agents the messages are events): 1) configurations, 2) authorization (AuthZ) response on a previous request, and 3) provisioning a new context(s). The Master PEP and all other agents are associated with a configuration file that contains: the ActiveMQ URL, the topics names, and description on each topic. Moreover, all agents subscribed to the topic will be able to receive all messages. Therefore, they have the possibility of using rules to filter messages in order to receive only the interesting ones.

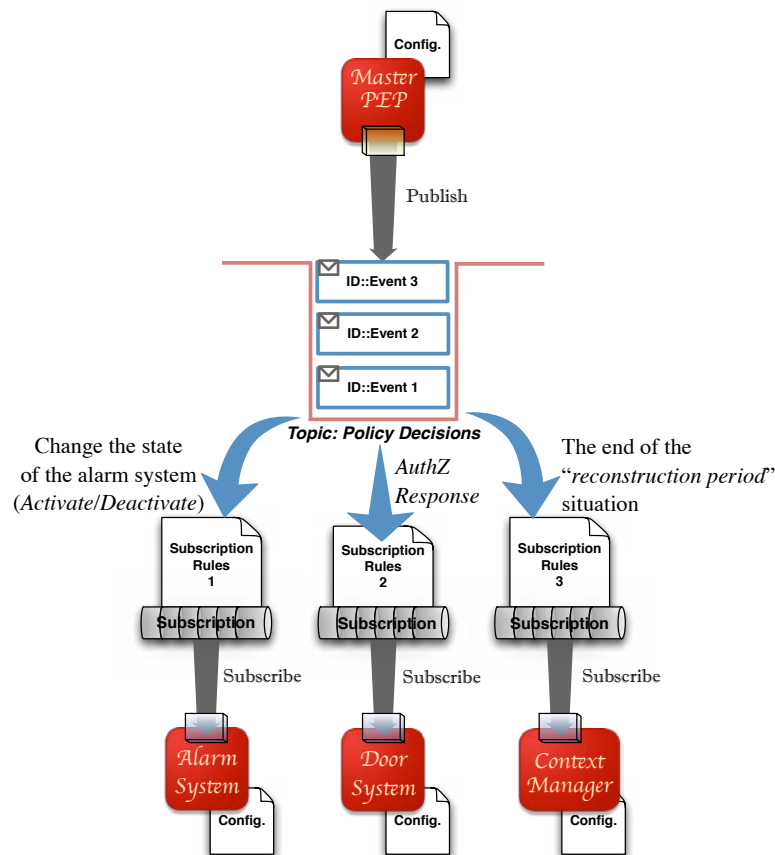
First, let’s assume that the current situation of building 1R1 is “*reconstruction period*”. Therefore, the Master PEP sends configurations at the beginning of this situation to deactivate the alarm system (i.e., using the earlier configuration message). The alarm system based on its subscription rules will get the messages with destID equal to ActPEP1. Now, this agent can respond to the policy obligation by applying the configuration “deactivate the alarm system”.

Second, the Master PEP receives an authorization request from the door system asking to allow Bashar to enter. The Master PEP publishes the authorization response to the topic “*policy decisions*” that says to deny Bashar to enter (i.e., the current situation is “*reconstruction period*”). Thus, the door system will deny Bashar’s request and the door should stay closed.

Third, the situation “*reconstruction period*” ends normally at the 30<sup>th</sup> of November as we specified. Let’s assume that the workers have finished the reconstructions before this date. Therefore, this new context is detected by the Context Manager and forwarded to the Master PEP. The latter can

take a security management decision to keep the building closed until the agreed time comes. Thus, the Master PEP sends a new context “end reconstruction period” to the responsible agent. This is an end-point for the situation “*reconstruction period*”. So, we have two possibilities. Either to declare this functionality for the Context Manager, then it will transfer it through the topic “complex events” to the Situation Manager. Otherwise, we can directly include the Situation Manager in this process and consider this decision as an ending-point (EP) to the situation “*reconstruction period*”. Thus, it will remove this situation from the database.

Indeed, it is more efficient to connect the Situation Manager directly. However, we keep both possibilities because not all contextual decisions from the policy may end or start a situation. Some decisions participate only in the SP or EP and therefore needs the Context Manager to process the event (i.e., the policy decision) with others in order to create the expected SP or EP.



**Figure 5.65: Example on the management of exchanges**

Following these techniques, we can now present our complete event-driven XACML architecture that aims at identifying situations, orienting the dynamic security policy decisions by these situations, and unifying the enforcement of these decisions by giving the possibility to outsource dynamic authorizations and provision adaptive configurations.

## III.SCD Unified & Adaptive Architecture

We presented our architecture in several steps. First, we presented a conceptual view on our architecture (the Master PEP architecture) with the necessary agents, the relationships between them and the possible communication paradigms to be implemented. Second, we present the technology we are using based on an event-driven architecture (i.e., messages bus implemented using Active MQ). Third, using this technology, we implemented the different agents, their inter-communications, and their exchanges techniques.

### III.A. The SCD Unified Architecture

Now, we want to make a coherent presentation to the complete architecture in order to demonstrate its unification feature (i.e., unifying the three presented interrelations). We claim that our SCD unified architecture is generic and can give a solution to implementing any dynamic security management system (i.e., with respect to our earlier expression methodology).<sup>31</sup>

In Figure 5.66, we present the full view of our SCD unified architecture. For the ease of explanation, we divide the architecture presentation into parallel management phases: 1) initiation in dark blue, 2) situations management loop in violet, 3) decision-making management loop in yellow (i.e., authorizations and obligation decisions), and 4) decision-application in light blue (i.e., enforcing authorizations and applying configurations). The green arrows mean subscribing (collecting messages) and the orange arrows mean publishing (sending messages).

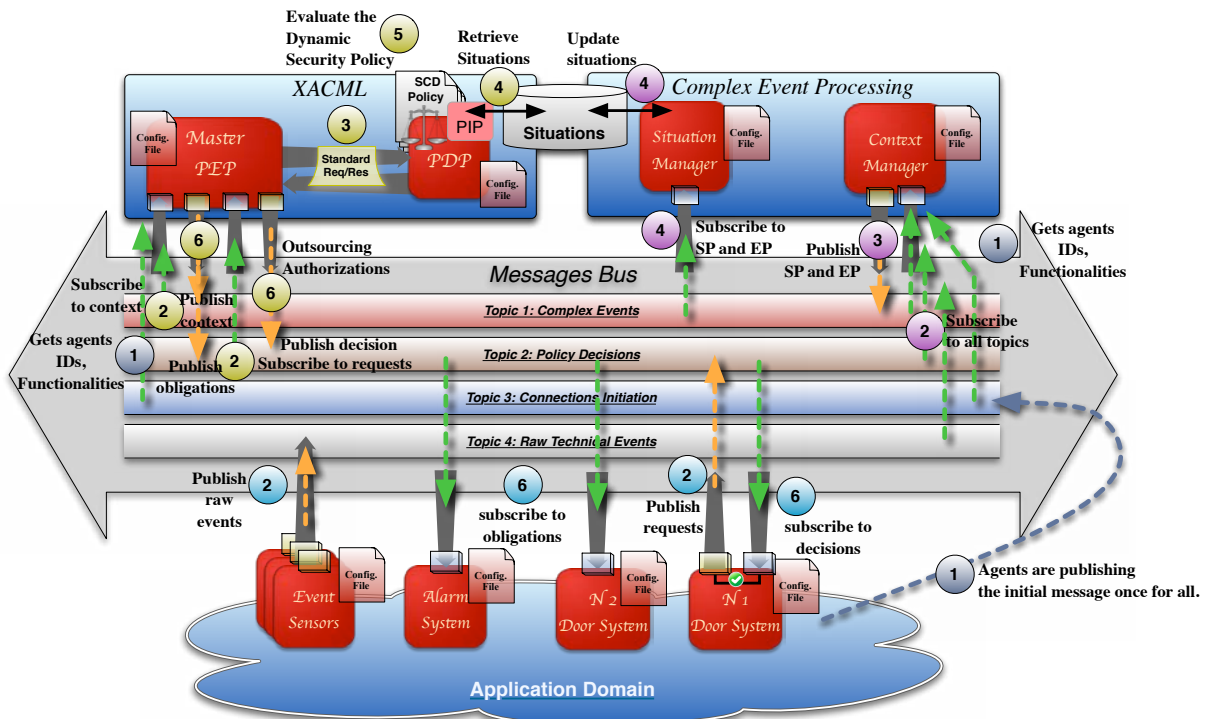


Figure 5.66: The full view on the SCD Unified Architecture

Nevertheless, all steps are indispensable one for another in our architecture. The architecture contains four topic-based streams—JMS messaging streams (Java Messaging Service) built inside a

<sup>31</sup> We will make proof of this claiming in the next Chapter

messages bus implemented using the Apache ActiveMQ framework. The framework launching steps are the following (i.e., after expressing and representing all presented steps in Chapter 4):

- i. **Messages Bus:** we launch the bus on a server and we can administrate and monitor the bus through the local link: `tcp://localhost:61616`. However, the agents configuration file contains the remote link address (`http://IP.Address:Port/`).
- ii. **Situation Management & Policy Management Architecture:** We launch the agents for both management loops at the same time: Master PEP, Situation and Context Managers, and PDP (including its PIP).
- iii. **Management Agents:** We launch finally all the agents that are controlling the managed elements: alarm system, door systems (N1 & N2), technical sensors and any other event sources.

The architecture implementation passes by six parallel steps (1, 2, 3, 4, 5, and 6). Having the same step with the two or three different colors means that the step is done in parallel in several phases (e.g., steps 2 and 6). We present our implemented architecture functionality by first presenting the initiation step. This is the first step that occurs after each time of launching our framework:

- 1 All the software agents publish their initiation message described earlier with their ID and functionality. The Master PEP and the Context Manager collect this information and build two lists holding all the management agents (i.e., alarm system, door systems, etc.). However, this step can be repeated whenever a new management agent is added to the managed environment. For instance, we deploy a fire alarm system. Hence, the new management agent needs to do the self-identify.

After the initiation step, we have the second step that passes in parallel by three phases:

- 2 The Context Manager subscription starts monitoring all types of events circulating inside the exchanging channels of the three topics: 1, 2, and 4.
- 2 The Master PEP subscription rules allow to retrieve messages from two topics 1 & 2 that concern messages with relevant context to the policy or access requests.
- 2 The management agents start their functionalities either as a sensor, actuator, or a smart node. For instance, technical sensors publish their raw readings to the topic 4. The door system N1 starts sending access requests (if any).
- 3 After the management agents start publishing their readings, the Context Manager can now identify complex events. If any context matches the SP or EP for the Situation Manager, the Context Manager sends this SP or EP to the topic 1.
- 3 The Master PEP subscription rules should allow the retrieval of any relevant context through topic 1 or any access request in the topic 2. Upon any detection, the Master PEP sends a standard XACML request to the PDP to check the SCD policy decision.
- 4 At this point, the Situation Manager subscription rules allow to retrieve SPs and EPs. In case the message is relative to an SP, the *situation manager* will create a new instance in the database to mark the start of an entity's situation. Otherwise, it will update the database and delete the situation corresponding to the received EP. This is the final step concerning the situation management loop. Afterwards, steps 2, 3, & 4 in violet continue to be repeated as long as the framework is working.
- 4 Through its PIP, the PDP gets the new situations relevant to any XACML request made by the Master PEP.
- 5 The PDP evaluates the XACML request based on the SCD policy and with consideration to the new situations. As a result of this evaluation, the PDP sends back the XACML decision to the Master PEP.



- 6 Once the Master PEP receives the PDP response, it processes the response finding the destinations of this response and divide the decision to publishable messages. Hence, if there is inside one XACML response only one-authorization-decision to a pending request (synchronized) from a smart node (e.g., the door system N1), the Master PEP sends the decision response. Otherwise, if there are any obligations, the Master PEP obligates each destination (i.e., actuator or smart node agent retrieved from its initiation list; any of the alarm system, the door system N1 and the door system N2) to apply configurations on their managed elements.
- 6 Through their subscription rules, the management agents retrieve messages concerning configurations and authorizations. The actuators receive configurations from the Master PEP and apply them on their managed elements. The smart nodes receive authorizations and enforce them on the accesses requestor. The smart node, however, may receive configurations as well accompanied with the authorization decisions. Then, the smart node needs to apply the configurations as well on their managed elements.

In conclusion, we presented our architecture that unifies different types of agents and different policy control models with the help of event-driven architecture and technologies. Our complete architecture is able to implement successfully any policy expressed and represented using our SCD methodology (explained in Chapter 4). We claim our architecture to be generic in the sense of being applied on whatever information system case study. We proof this claiming in the next chapter by providing, expressing, and implementing different use cases.

Nevertheless, we said that our architecture is adaptive, but our agents are static and cannot be upgraded. Any changes to the PEP agents require a phase of recoding and stopping our security management system. Thus, we present in the next section how we can conceptualize and implement adaptive PEPs that accept dynamic upgrading of security management configurations.

## III.B. Adding the adaptability feature

Now, security configurations are possible with the new XACML v. 3.0 obligation expressions. However, the obligations are included inside the policy in a static manner, and upgrading the configurations to adapt to technological advancement requires the interruption of the security management system and the modification of the security policy, which contradicts with what we are presenting in this thesis. Therefore, we first present how to tackle the adaptability problem of security configurations, and then we present the adaptability solution unified with the dynamic XACML architecture.

For instance, if the technologies used for deploying the alarm systems are renewed, the technical configurations should be changed. However, the policy itself does not require us to change it because the configurations are externalized and dynamic. Therefore, the security management system will not be interrupted and we can only change the implementation of this configuration, but not the representation inside the policy.

### III.B.1. Defining Adaptability Notions

In systems and networks management, security management solutions should meet the requirements during behavioral changes to any given situation inside the environment. A system is believed to be “adaptive” as it is able to meet the security requirements whenever the conditions and the circumstances are changed. Consequently, any dynamic security management system is said to be *adaptive* whenever it performs security changes to the managed elements in compliance with identified situations. Therefore, the previous situation awareness permits DSMS to identify situations.

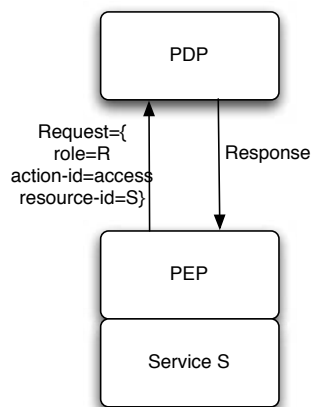


Upon undesired changes, the DSMS should apply adaptable security configurations. This process is defined in literature as the *adaptation* process.

Formally speaking, the adaptation process aims at changing a system to respond to new monitored changes inside its environment. Given the multitude of definitions related to this concept, we have chosen to retain one that is both generic and reduces ambiguities by its formal approach (Subramanian & Chung, 2001). In addition, we fix the objective of applying the adaptation approach to fulfill a dynamic adaptability of the system (Fox & Clarke, 2009). Therefore, in the light of our contribution, the adaptation of an authorization system is the ability to change this system, because it no longer meets the dynamic behavior of the environment. The new situations (i.e., representing the dynamic behavior) inside the environment consequently represent a new environment—suggesting a new requirements definition—may also cause the creation of a new environment. To meet the requirements connecting the system with its environment, the monitoring process notifies the system about the new changes. Consequently, the creation of a new system takes place by dynamically adding or modifying the old one (i.e., without interruption to its functionality) with the aim of either meeting the new requirements or adapting the new environment’s situations to the requirements (Cheaito, 2012, p. 43).

### III.B.2. Current XACML PEPs

We present a simple scenario to illustrate the need of adaptability at the PEP side. Let us consider a policy that grants subjects with role *R* to access service *S*. The PEP has been designed to catch any request to service *S* and transform it into an XACML request that it sends to the PDP (Figure 5.67). When someone with role *R* requests access to service *S*, the decision of the PDP sent to the PEP only contains the effect (PERMIT in this case – Figure 5.68, a). At some time, the security administrator wants to log the granted accesses. She modifies the rule by adding an obligation to inform the PEP having to log granted accesses (Figure 5.68, b). However, this obligation did not exist when the PEP was designed, and then the PEP does not provide any functionality to enforce this obligation.



**Figure 5.67: Messages exchanged between the PEP and the PDP**



```
<Response>
  <Decision>Permit</Decision>
</Response>
```

(a) Response sent by the PDP when no obligation

```
<Response>
  <Decision>Permit</Decision>
  <Obligations>
    <Obligation ObligationId="urn:siera:example:obligation:log-access:file"
      FulfillOn="Permit">
  </Obligations>
</Response>
```

(b) Response sent by the PDP including an obligation

**Figure 5.68: Responses sent by the PDP**

As a consequence, we can formulate the following requirements about the ideal PEP:

- **Req1:** *The PEP must be able to enforce any XACML decision.* Whatever the response of the PDP is, the PEP must be able to enforce it.
- **Req2:** *The PEP must be dynamically adaptable.* The policy can change over time and new requirements may appear. The PEP must be able to adapt at runtime.
- **Req3:** *The PEP must be XACML v3 compliant.* The XACML Request/Response protocol must not be changed.
- **Req4:** *The PEP must provide interfaces to be integrated to an application/service.* In this example, the PEP controls the access to service S. The PEP transforms specific service S data into an XACML request and enforces the Permit/Deny decision into service S. Thus, there must be an interface to allow programmers to integrate the PEP with service S.
- **Req5:** *The adaptation feature must not complicate the task of PEP programmers.* Currently, requesting a decision from a PDP is implemented by methods like `getDecision()`. Using our adaptive PEP must not be more complicated.

### III.B.3. A Dynamically Adaptable XACML v3 PEP

We present in this section the description of our adaptable PEP architecture. We have followed a service-oriented component approach that takes advantage of integration and dynamicity from service-oriented architectures and of reusability and dependency management from component-oriented models.

The service-oriented architecture promotes modeling solutions in terms of provided services described by contracts. It is based on the idea of composing applications by discovering and invoking available services to accomplish some tasks (Papazoglou et al., 2007). The general pattern of service-oriented solutions consists in service providers, service consumers, and a service registry. Service providers publish services at runtime and service consumers' requests for a specific contract. Different service providers can offer the same service, and the consumer can choose one of them based on the contract—since the framework allows service consumers to be dynamically notified of new registered services and unregistered services.

Component oriented programming focuses on making reusable, logical blocks of software that implement one or more interfaces. A component-oriented software is then an assembling of components. The notion of interface is very similar to the notion of service interface, and component-oriented programming has been used to implement services. A service can be implemented by one or more components.

Adaptable enforcement of security management configurations needs a dynamic mechanism in order to be compatible with the existing heterogeneity between different technical infrastructures. Therefore, exchanging between components, plugins, or any other software modules on top of a technical environment, the service-oriented component (SOC) creates a dynamic approach aimed at delivering loadable and unloadable services to complex environments with consideration for managed elements' heterogeneity (Cervantes & Hall, 2004). The approach combines the advantages of the service-oriented architectures (SOA) in terms of integration and dynamicity with the advantages of component-oriented models (SOM) in terms of reusability and dependency management (Laborde, Barrère, & Benzekri, 2013). The following subsections explain both the SOA and the SOM approaches. Afterwards, the framework used to implement SOC in the thesis contribution is presented.

### **III.B.3.I Service-Oriented Architecture**

The Service-Oriented Architecture (SOA) aims at providing solutions through modeling contracts of described services. To accomplish some tasks, SOA essentially offers discovering and invoking available services to compose applications (Papazoglou et al., 2007). Any service-oriented solution generally consists of a service provider, a service consumer, and service registry patterns. Service providers publish services at run time and service consumers' request services for a specific contract. Different service providers can offer the same service, and the consumer can choose between them based on the contract, since the framework allows service consumers to be dynamically notified of new registered and unregistered services.

Briefly, SOA software architecture is based on the key concepts of an application front-end, service, service repository, and service bus (Sprott & Wilkes, 2004). A service consists of a contract, one or more interfaces, and an implementation. A service in SOA is an exposed piece of functionality with three properties:

- i. The interface contract to the service is platform-independent.
- ii. The service can be dynamically located and invoked.
- iii. The service is self-contained. That is, the service maintains its own state.

### **III.B.3.II Component-Oriented models**

Programming using a component-oriented approach means building reusable logical blocks of software (i.e., components) that implement one or more service interfaces. A component is a contract with the realizer. Any component-oriented software represents components assembly. A service can be implemented by one or more components. An interface is a contract with a client, describing a set of behaviors provided by a component object. It defines a list of operations, their signatures, and semantics.

### **III.B.3.III OSGi Framework**

One way to implement service-oriented component architecture is using the OSGi framework. Created initially for Java-based systems, OSGi offers a framework for modular systems. OSGi gives the possibility of declaring *dependencies* between individual modules. It allows users to manage the lifecycle and dynamically adapt each component of the system.

The term “OSGi” is only a specification. However, the implementations can vary and are mostly taken by Equinox, Apache Felix, and Knoplerfish. The OSGi implementations are based on the JVM and provide mechanisms for service management, component definition, execution, control, and life cycle management of modules.

The bundle is the most essential OSGi concept. It is the component that builds and draws the OSGi structure. At the deployment level, an OSGi bundle is interpreted as a Java jar file. Nevertheless, there are major differences between the “bundle jar file” and the regular one. One can recognize them by referring to the OSGi specific manifest definition and some OSGi specific classes.

Another important concept of OSGi is *services*. Services specify the interaction between the bundles. Two sides represent these services: (a) the definition of interfaces and (b) the registration to an implementation that performs these interfaces. Service registration is stored within a service directory. As with SOA, services make OSGi-based systems more decoupled and independent. Moreover, OSGi makes it possible to adapt components and to interchange between them during the runtime.

Finally, any OSGi framework offers a platform to manage the lifecycle of bundles. Every bundle holds its own OSGi configuration. The configurations are mainly the *dependencies* needed to run the bundle. The OSGi framework knows through the configuration files the order of execution, and therefore the lifecycle management is applied to each of the components with a given order. Remarkably, the two main classes that help the management of the lifecycle are the “Activator” and the OSGi “interface”.

### III.B.3.iv An adaptive PEP Architecture

Our architecture drawn in Figure 5.69 consists of the following components:

- The core PEP is the corner stone of our architecture. This component:
  - Calls service `getDecision` to get a PDP response from a PDP access point
  - Calls service `applyObligation` if any in the response of the PDP.
- Install obligation modules at runtime.
- A PDP access point implementation. This component allows the PEP to contact a PDP from various methods.
- Obligation implementation components. Each obligation component publishes the urn of the obligation it implements allowing the CorePEP to differentiate them.

All these components have been implemented in the OSGi framework.<sup>32</sup> Making the OSGi framework cooperate with a pure Java application is a hard task (Hall, Pauls, McCulloch, & Savage, 2011) (e.g., they use different class loaders). However, PEPs must be integrated into applications, which may not be pure OSGi. Thus, we have created a Java class `ExtensiblePEP` that launches a customized OSGi environment and communicates with the `CorePEP` seamlessly from any Java application.

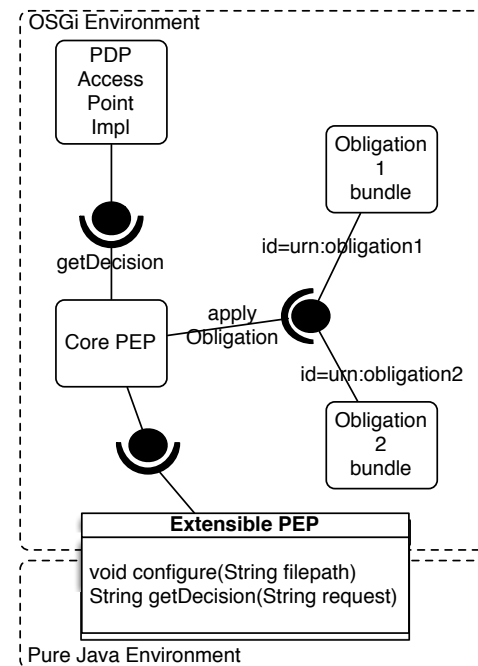


Figure 5.69: Our adaptive PEP architecture

### Adaptation specification

The corePEP is able to install new obligation components. However, a mechanism is required to inform the PEP about (a) what component implements what obligation and (b) how to install and configure it. We have decided to provide this information in the policy itself. XACMLv3 provides the concept of advice that provides supplemental information to the PEP. In addition, advice can be ignored by the PEP. Hence, advice is a good way to provide information about obligation components.

The second benefit of sending obligation components information inside advices is that this information is sent in PDP results alongside obligations. Thus, no extra protocol is needed.

<sup>32</sup> OSGi, <http://www.osgi.org/Main/HomePage>, last access April, 2014

We propose to employ two advices for enforcing obligations:

- The first advice, called installation advice, provides information about where to download the component. It must have an id that equals to *urn:siera:management:obligation*. This advice must have attribute *urn:siera:obligationId* set to the URN of the obligation. Attribute *urn:siera:bundle:file:location* indicates the location of the component and attribute *urn:siera:bundle:file:name* contains the name of the component file. For example, the advice of Figure 5.70 states obligation *urn:siera:example:obligation:log-access:file* is implemented by component *FileAccessLog\_1.0.0.201306191231.jar* that is available in directory *obligation-store*.
- The second advice, called configuration advice, provides the configuration of the component. The *adviceId* must be the same as the *obligationId*. Attributes within this advice are specific to the component. In our example, the component logs requests and results in a file. The advice sets the path file.

```

91 <ObligationExpressions>
92   <ObligationExpression ObligationId="urn:siera:example:obligation:log-access:file" FulfillOn="Permit">
93   </ObligationExpression>
94 </ObligationExpressions>
95 <AdviceExpressions>
96   <!-- Advice for installing the component -->
97   <AdviceExpression AdviceId="urn:siera:management:obligation" AppliesTo="Permit">
98     <AttributeAssignmentExpression AttributeId="urn:siera:obligationId">
99       <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
100         urn:siera:example:obligation:log-access:file
101       </AttributeValue>
102     </AttributeAssignmentExpression>
103     <AttributeAssignmentExpression AttributeId="urn:siera:bundle:file:location">
104       <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
105         obligation-store/
106       </AttributeValue>
107     </AttributeAssignmentExpression>
108     <AttributeAssignmentExpression AttributeId="urn:siera:bundle:file:name">
109       <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
110         FileAccessLog_1.0.0.201306191231.jar
111       </AttributeValue>
112     </AttributeAssignmentExpression>
113   </AdviceExpression>
114   <!-- Advice for configuring the component -->
115   <AdviceExpression AdviceId="urn:siera:example:obligation:log-access:file" AppliesTo="Permit">
116     <AttributeAssignmentExpression AttributeId="urn:siera:example:obligation:log-access:file:path">
117       <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
118         /var/log/siera/file.log
119       </AttributeValue>
120     </AttributeAssignmentExpression>
121   </AdviceExpression>
122 </AdviceExpressions>

```

Figure 5.70: Extract of the obligation expression for the “auditing” functionality

As a result, the global process of the PEP is the following (Figure 5.71):

- *Step 1* – The CorePEP receives the result from the PDPAccessPointImpl. For each obligation, the CorePEP tries to call service *applyObligation*.
- *Step 2* – If the service is not available, the CorePEP looks at the installation advice for that obligation. It downloads the component file.
- *Step 3* – The CorePEP installs the component in the OSGi environment and sends the configuration advice to the component.
- *Step 4* – The CorePEP calls service *applyObligation*.

This approach fulfills our requirements since:

- **Req1** – The PEP can be upgraded with a new obligation component if needed.
- **Req2** – Our PEP is adaptive. Obligation components are installed and configured at runtime.
- **Req3** – We use XACMLv3 features only.
- **Req4** – The PEP can be easily called from any Java program using method *getDecision()*.
- **Req5** – Adaptation is managed transparently.

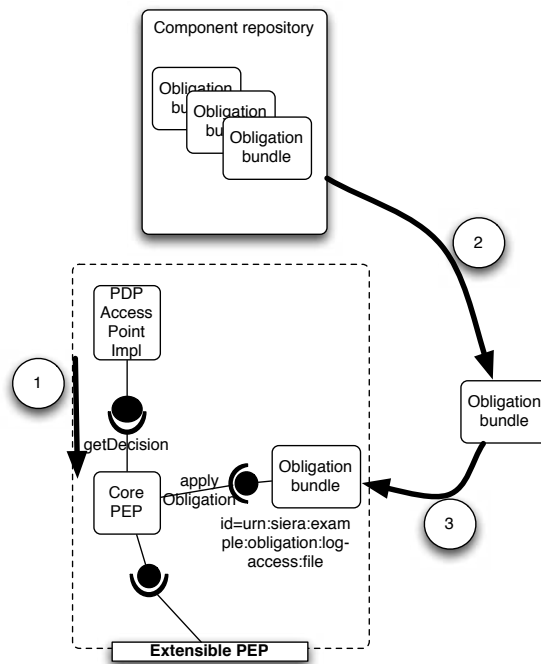


Figure 5.71: Process for upgrading the PEP

### III.C. Adding adaptability to agent implementations

Using our previous presentation to the adaptability feature, we can upgrade any agent with this feature by instantiating the Java class Extensible PEP inside this agent. Then, we can use the functionalities presented earlier upgrading new configurations (see Figure 5.72).

Nevertheless, in our unified architecture we identify three types of agents that can accept the upgrading feature: actuators, smart nodes, and Master PEPs. Thus, in Figure 5.72 we can replace the “Agent” by any of these three types. In case of the actuator, the alarm system is now decoupled from the deployed infrastructure technology. Hence, any upgrading to the configurations can be programmed and added to the agent without stopping our security management system. Likewise for the smart node “door system N1”, the agent accepts new configuration upgrades.

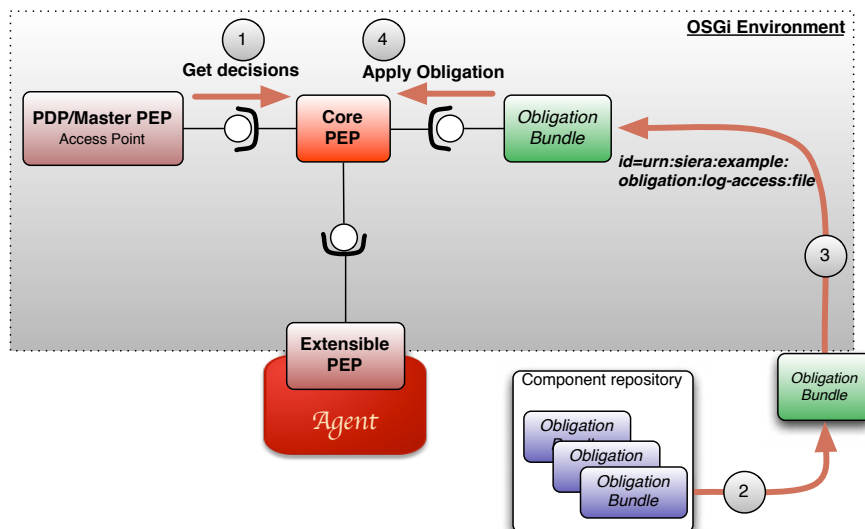


Figure 5.72: Upgrading agents with adaptability

Finally, upgrading the Master PEP allows having a local PDP, an adaptive PEP actuator to apply dynamic configurations directly, and an adaptive PEP smart node to use requests/response locally and apply dynamic configurations directly. By this upgrading of the Master PEP, we are closer to the MAPE agents (Monitoring, Analyzing, Planning, and Executing). However, we ensure the monitoring, the analyzing and the execution functionalities, but we still need to implement the planning functionality (i.e., as a future work).



# Chapter 5 Conclusion

We have presented in this chapter our dynamic security management system that expresses authorizations and configurations that need special enforcement. The existence of two policy control models to enforce the policy causes the implementations to choose the outsourcing or the provisioning approach. Therefore, our usage of both authorizations and configurations invited a unified enforcement of the dynamic security policy. Moreover, the need for configurations to adapt to technological changes invited us to implement an adaptive policy enforcement point.

As a result, we introduced a unified architecture with adaptive enforcement that respects the dynamic security policy. In Figure 5.73, we summarize the outcome of applying our contribution to the architecture. Supposing that we have six situations. Specific events represent the behavior changes of the managed elements and change the nominal situation 1 to either start (SP) situation 2 (s2) or situation 3 (s3). In case of s2, the fact of having an access request (rq1) would get the denial response and may be accompanied with obligations that start situation 4. However, being in s3 will grant rq1 the permission to access and may also apply some obligations to start situation 5. Otherwise, configurations might be provisioned during s3 to start situation 6.

Nevertheless, the red arrows represent events coming from the environment that express either behavior changes of managed elements or access request from the latter. The blue arrows represent the dynamic synchronized authorization decisions that resulted from evaluating the dynamic security policy with the orientation of situations. Finally, the green arrows represent the adaptive enforcement of configurations. These configurations could be either synchronized (i.e., when accompanying the peer-to-peer authorization decisions) or asynchronous (i.e., when provisioned to the managed elements) (Figure 5.73).

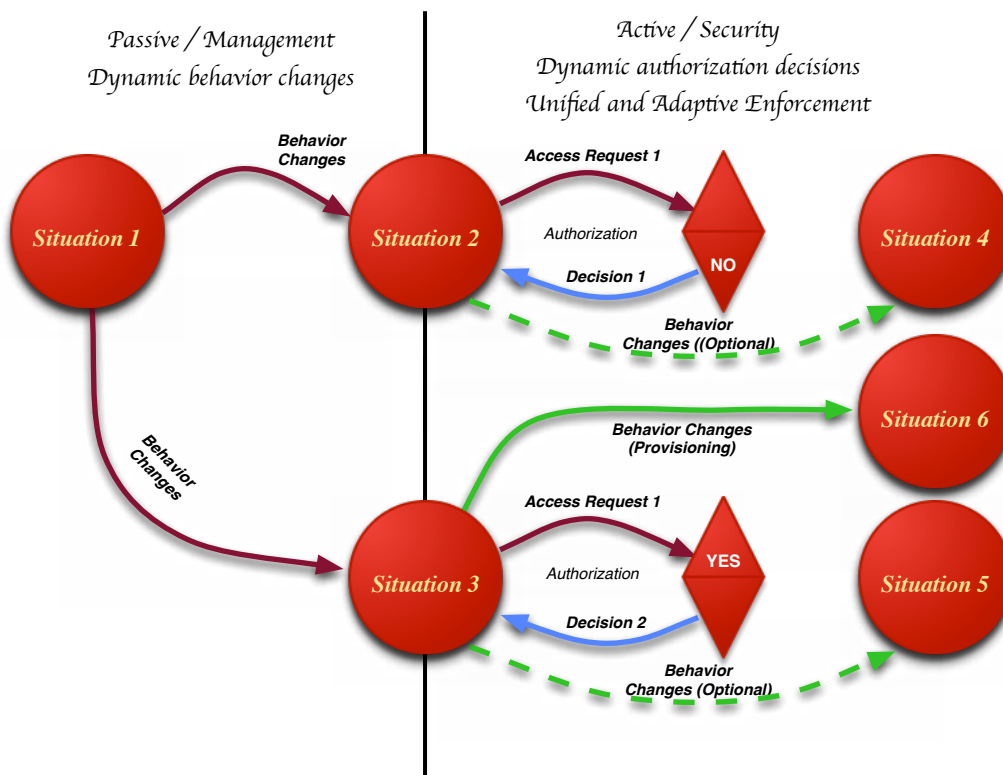
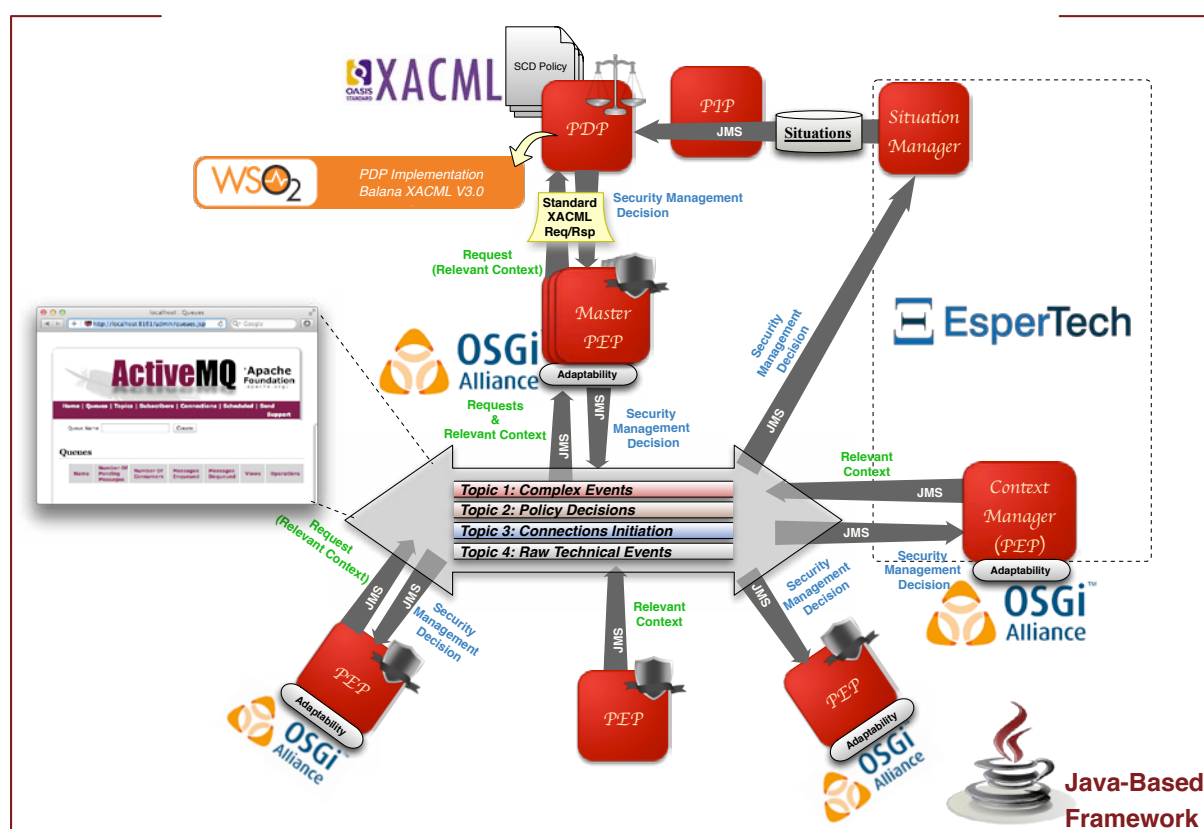


Figure 5.73: State diagram representing our DSMS enforcement logic

In conclusion, Figure 5.74 demonstrates the implementation of all the interrelations we presented between the PDP, the PEPs, and the SM architecture using our unified architecture. We are able to implement different communication paradigms (i.e., outsourcing, provisioning, and a mix of them) between all necessary agents: actuators, sensors, and smart nodes. Moreover, we implemented and managed a uniform exchanging approach upon our messages bus. We introduced and implemented a new smart PEP agent that we named “Master PEP”. We used this agent as a control point of exchanges between all connected agents and the situational-oriented decision-making process.

Finally, we presented our complete architecture in-depth and we added a new feature to our actuating agents that we described as “adaptability”. We defined what is adaptability and how we implement and upgrade PEP agents with dynamic configurations. Then, we demonstrated how we could use these upgrading into these agents and the possibility of having local PDPs (i.e., inside each adaptive PEP). Technically, we implemented this architecture using different available technologies, but all agents are implemented using Java language. The ESB messages bus is implemented using apache Active MQ (i.e., it provides an interface to monitor subscriptions, queues and messaging exchanges). The SCD policy and the authorization response/request are represented using the standard OASIS XACML version 3.0. The PDP is implemented based on an existing open source implementation from WSO2. We modified the Balana’s implementation in order to meet our requirements (i.e., Balana implementation eases the handling of the new obligation expressions of the XACML v 3.0) [Balana – XACML Info, August 2012]. The Situation and Context Manager are implemented using the Esper technology (i.e., implementing complex event processing engines and their EPL configuration language) [ESPER Tech. Inc., 2006]. All messages exchanging (publishing or subscribing) are implemented using Java Messaging Services (JMS). The adaptability feature is implemented using OSGi framework for Java (namely, using the bundles).



**Figure 5.74: Conclusion view on our SCD unified architecture**

# Chapter VI

## Proof of Concept: Three Different Scenarios

---

*We expressed and implemented our contribution on the SCD policy. The framework, which we have demonstrated through one trivial example in the previous two chapters, does not prove its “generic nature”. Therefore, we focus in this chapter on proofing this generic concept by choosing different scenarios in term of problem nature. The first scenario tackles exceptions with highly dynamic environment. Fast reactions and bypassing security should be tolerated to save lives. The second scenario tackles managing authorizations of virtual parties accessing one shared resource. The dynamic state of this resources implied having dynamic authorization decisions. The third scenario tackles pure management problem. Organizing and scheduling the submission of jobs require being efficient in term of energy usage. Thus, managing the security of the job submission process, should not only ensures the process efficiency, but its respect to the security policy.*

---

### Summary

#### **I. First Scenario:**

- Security Management of Healthcare Information Systems
  - Case Study: Break The Glass

#### **II. Second Scenario:**

- Security Management of Virtual Organizations
  - Case Study: Workflow Management

#### **III. Third Scenario:**

- Security Management of Job Submission in grid computing
  - Case Study: Energy Management

#### **IV. Framework Evaluation**

# Chapter 6: Proof of Concept: Three Different Scenarios

---

*“It is very unfair to judge any body’s conduct, without an intimate knowledge of their situation. Nobody, who has not been in the interior of a family, can say what difficulties of any individual of that family may be.”*

*Jane Austen*

*Columbia World of Quotations*

*Retrieved November 10, 2014, from Dictionary.com*

---

## Introduction:

It is difficult to prove the generic nature of an architecture and stating it is applicable to any system in which its objective is ensuring dynamic security management. In this chapter, therefore, we seek to prove our framework (i.e., language and architecture) can be applied to three examples of our choices.

The idea of proposing the first scenario is to demonstrate a specific security problem in healthcare known as *Break the Glass*. Through this problem, we are able to present bidirectional relationship on how situations orient security decisions, and how security decisions configure situations. Thus, we present the added value of using a meaningful feedback, of using this feedback to orient the decision-making, and of *optionally* using the decision outcome (i.e., provisioning asynchronized configurations) in stimulating changes to this feedback (e.g., put an end to an ongoing situation or start a new one). Moreover, we demonstrate with such highly dynamic environment how it is possible to express dynamic authorization policies to consider exceptions (e.g., emergencies) without modifying or updating the rules.

The idea of proposing the second scenario is to demonstrate the use of our framework into a security-oriented case study. The management of workflow for collaborative production of shared specification documents should be ensured with compliance to the management of security. Thus, there are two parallel management paradigms in this case study: workflow and security. The added value of implementing this scenario is to demonstrate 1) one directional relationship on how changing situations orients the security decisions, 2) the flexibility and simplicity of our security management framework when using situations, and 3) the effective of using our conceptual methodology to engineer the scenario’s situations and policy.

The idea of proposing the third scenario is to prove that our framework has *generic nature* in term of its application domain. The previous two scenarios aim at managing security of managed elements with security objectives only. However, through this third scenario we demonstrate that managing the security of management nature scenario involves ensuring management functionalities (i.e., configurations). Unlike the first scenario, management operations are obligations that *must* be fulfilled in addition to the authorization decision. Thus, the added value of this scenario is to demonstrate the usefulness of having both synchronized (outsourcing) and asynchronized (provisioning) management communications in our framework.

For each scenario, we present the scenario specification, picking from it the relevant situations and model them with association to contexts and decisions, representing the SCD policy, technically identifying the situations, and using the SCD architecture to integrate new scenarios’ agents.

# I. 1<sup>st</sup> Scenario: Break The Glass

In Healthcare, “Break The Glass” (BTG) is a good example that treats the dynamic authorization problem through bypassing traditional authorizations (static). In emergency circumstances, unauthorized doctors can break the policy to get rights to access information that they could never have in an ordinary case. Well-known solution for BTG is giving entities administrative rights to break policy and modify rules when it comes to patients’ life. Our scenario is the following:

*Emma is a doctor in a modern hospital. Access to patients’ information by doctors is controlled. The intervention should be legalized by a reason, e.g. treatments. Therefore, only doctors with reasons are allowed to access. Patients in such hospitals are observed through many sensors that send information about: fever, pulse, blood pressure, conscience status and numerous bio-medical signals. Contextual sensors (CSs) are connected to capture physical movements and positions: room occupancy, patient’s position and doctors’ position. CSs are connected to an alarm system. Once an alarm is launched expressing an emergency situation, a notification message is sent to the doctor in charge in such situation for the concerned patient.*

*Assuming that one-day connected sensors to patient Joe show urgent need for a doctor. While his responsible doctor is not available, Emma was ready to involve. Unfortunately, Joe is not one of her patients. Traditionally, she won’t have access. The BTG solution gives Emma rights in emergency situations to break general policy in order to save Joe’s life.*

**Breaking** the general policy is required to change the authorization. To change rules (break the policy), one should have administrative permissions to do so (in the worst case Emma can change the current policy). Emma will break the policy and give herself authorization to access Joe’s files.

It is important to highlight the importance of this example for our thesis work. The objective of this contribution is to avoid Emma from having administrative permissions, which does not belong to her role as a doctor. All what Emma should concern about is Joe’s life.

## I.A. Modeling BTG Situations

Giving the BTG scenario, we choose three situations extracted from the following scenario specifications: *Patient needs a doctor*, *patient health is normal*, and *the medical-record<sup>33</sup> is broken when the BTG permission is granted*. We believe that modeling three situations is enough to ease and present the expression and implementation of the BTG scenario.

Initially, the healthcare authorization system evaluates the access requests to PI based on the stable policy oriented by a normal situation (i.e., when all systems elements are in normal situations). Within these situations, Emma is not permitted to do any of the following actions: accessing Joe’s PI, BTG request to Joe’s PI and ending the BTG request.

When Joe’s health become worse and there is no available responsible doctor nearby to save him is found around, Joe will be in an urgent need for a doctor (i.e., the situation named “Doctor Needed”). As Emma is the only available doctor, she will receive a notification to take in charge the

---

<sup>33</sup> MD is the electronic record that contains the patient’s information

treatment of Joe. Emma cannot access Joe's PI directly, so she will present a BTG request to Joe's PI. Logically, she won't have permission to end the BTG request, as it is not placed yet.

Once the glass is broken "*BTG Situation*" (i.e., the situation of the PI), Emma can access Joe's PI, but she cannot place another request to BTG again (i.e., as it is already broken). Once Joe treatment is done, Emma can end the BTG situation on his PI. As a result, the situation of Joe's PI will get back to normal and the cycle is completed.

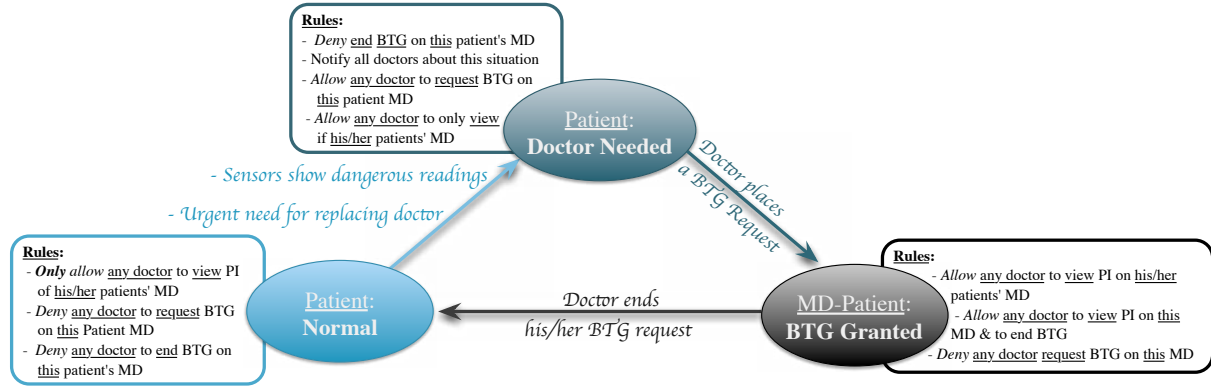


Figure 6.75: State diagram of the BTG management in Healthcare Information Systems

## I.B. Representing SCD Policy

By considering these four-modeled situations, the policy expressing the dynamic authorization can be represented by the **six rules** below. The *first rule* states: if the doctor requesting access to PI is one of the persons responsible of the patient who owns this information. PRPL is the List of Persons Responsible who has access to the Patient's information (with the patient himself). In order for Emma to be able to place a request to break the glass (i.e., BTG Request), the system should be in situation that needs an external doctor (during the urgent need for a doctor: "*Doctor Needed*"). In this case only, she can break the policy using the *second rule*. The *third rule* expresses the configuration of a new message concerning specific situation. That is, at the beginning of the "*Doctor Needed*" situation, an email is sent to doctors to alert them about the urgent situation of the patient, Joe for example. The *fourth rule* is to ensure that only doctors who placed a BTG request can access the Patient Information. The objective of this rule is to avoid other doctors taking advantage of such situation. The *fifth rule* is to let Emma end the BTG process or request on Joe's PI once she finishes the treatment. Finally, to manage conflicts we declare a default sixth rule that denies all other access requests.

Using XACML version 3.0, we represent our SCD policy for the BTG scenario with 6 rules. It is easier with the provided state diagram (associated with contexts and decisions) and the specified SCD rules to write our XACML structure-like policy expressions.

**XACML Policy:**

**Targeting** *Situation (MD-Patient)* = "*Normal*" **For**

**Rule 1:**

**IF** *Type (Resource)* = MD-Patient  $\wedge$  *Role (Subject)* = Doctor  $\wedge$   
*Type (Action)* = Access  $\wedge$  *ID (Subject)*  $\in$  PRPL (*Resource*)  
**Then Permit**

**Targeting** *Situation (Patient)* = "*Doctor Needed*" **For**

**Rule 2:**

**IF** *Role (Subject)* = Doctor  $\wedge$  *Type (Resource)* = MD-Patient  $\wedge$   
*Type (Action)* = BTG Request  
**Then Permit**  
**Take Advice:** Find the Start-BTG Bundle at the following (file path)



*Apply Obligation: Start the BTG Situation of the resource*

**Targeting** *Situation (Patient)* = “Doctor Needed” **For**

**Rule 3:**

*IF SP (Situation)* = “Doctor Needed”

*Then Permit* -- the continuity of this situation

*Take Advice: Find the Patient-Alarm Bundle at the following (file path)*

*Apply Obligation: Notify doctors about this patient urgent need for intervention*

**Targeting** *Situation (MD-Patient)* = “BTG Granted” **For**

**Rule 4:**

*IF Role (Subject)* = Doctor  $\wedge$  *Type (Resource)* = MD-Patient  $\wedge$

*ID (Subject)* = BTG Requester (MD-Patient)  $\wedge$  *Type (Action)* = Access

*Then Permit*

**Targeting** *Situation (MD-Patient)* = “BTG Granted” **For**

**Rule 5:**

*IF Role (Subject)* = Doctor  $\wedge$  *Type (Resource)* = MD-Patient  $\wedge$

*ID (Subject)* = BTG Requester (MD-Patient)  $\wedge$  *Type (Action)* = End BTG

*Then Permit*

*Take Advice: Find the End-BTG Bundle at the following (file path)*

*Apply Obligation: End the BTG Situation of the resource*

**Targeting** *Any Situation* **For**

**Rule 6:**

*IF Any Conditions*

*Then Deny*

## I.C. Identifying BTG situations

Each event type should have at least values describing one element of the system (Subject, Resource, Action and Environment). We have defined a set of ten types of events with ESPER. Events Types are:

Table 6.11: Supported healthcare event types that the context manager can identifies.

Event Type	Event Description
<b>FevEvt</b>	Patient’s Fever,
<b>StatEvt</b>	Patient’s Status,
<b>PlsEvt</b>	Patient’s Pulse,
<b>PPosEvt</b>	Patient’s Position,
<b>OccEvt</b>	Room Occupancy,
<b>DrPosEvt</b>	Doctor’s Position,
<b>BtgRqEvt</b>	BTG Request,
<b>AlrmEvt</b>	Alarm Detection,
<b>AccRqEvt</b>	Access Request,
<b>SitEvt</b>	Situation Detection

The instances of these event types are generated and structured in event streams using events simulator (sensors agents simulating physical sensors). The agents generate events from the mentioned types (Table 6.11) and keep trace inside text-like files. We exchange all events with the structure that we defined in Chapter 5, for example we store the event value of the following message in a file:

senderID FeverSensor destID ContextManger EventType FevEvt PatientName Joe  
FevDegree 40 SensroID PCM2340 TimeStamp 130915221345



There are two examples of Complex Event types or patterns that aggregate events from the ten event-types mentioned earlier. In order for the complex event to be triggered, it should meet the conditions of a query that represents either a pattern, as the sliding window (Mozafari, Zeng, & Zaniolo, 2012), or a simple SQL-Like query:

- ⌘ **CE1 – Patient In Danger:** the Fever should be high, the status of the Patient is claiming and there is no one responsible near her/his.

```
select *           -- it is the Patient in Danger SP, but also the EP of the Normal situation
from              EventPatientFeverAlert.std:lastevent() as FevEvt,
                   EventPatientStatusAlerts.std:lastevent() as StatEvt,
                   EventPatientPlaceChange.std:lastevent() as PPosEvt,
                   EventRoomOccupation.std:lastevent() as OccEvt

where             FevEvt.patientID = PPosEvt.patientID and
                   PPosEvt.patientID = StatEvt.patientID and
                   PPosEvt.newRoomID = OccEvt.roomId and
                   FevEvt.alertMessage = 'Dangerous' and
                   OccEvt.presonsInside = '0' and
                   StatEvt.newStatus = 'Claiming'
```

- ⌘ **CE2 – Urgent need for a Doctor:** The Context Manager should check the position of the doctor if not in the hospital. Thus, a threshold is assigned to the number of persons inside the room, e.g. zero. This threshold is triggered only if the patient is claiming and one of his sensors' readings (fever or pulse) is set to high. Finally, this complex event is called whenever the patient is unconscious.

Whenever one of the following (CE2) or the previous (CE1) compositions is detected, the Context Manager sends SP of the situation “*Doctor Needed*” (i.e., this ends the situation “*Normal*” of the patient automatically by the Situation Manager).

```
select *           -- it is the Doctro-Needed SP, but also the EP of the Normal situation
from              EventPatientFeverAlert.win:time(1 sec) as FevEvt
                   EventPatientStatusAlerts.win:time(1 sec) as StatEvt,
                   EventPatientPulseAlert.win:time(1 sec) as PlsEvt,
                   EventPatientPlaceChange.win:time(1 sec) as PPosEvt,
                   EventRoomOccupation.win:time(1 sec) as OccEvt,

where             PPosEvt.patientID = StatEvt.patientID and
                   StatEvt.patientID = PlsEvt.patientID and
                   PlsEvt.patientID = FevEvt.patientID and
                   PPosEvt.newRoomID = OccEvt.roomId and
                   ((PlsEvt.alertMessage = 'High' or FevEvt.alertMessage = 'High' and
                     OccEvt.presonsInside = '0' and StatEvt.newStatus = 'Claiming')
                     or StatEvt.newStatus = 'Unconscious')
```

Whenever a BTG request is accepted by the PDP during the situation “*Doctor Needed*”, the BTG status of the patient medical record should be changed to the “*BTG Granted*” situation. If a doctor got accepted on his/her BTG request, the SCD policy obligates the BTG Manager to start the situation “*BTG granted*”. Thus, the BTG Manager fulfills the obligation by sending a *simple event* to the Context Manager (ordering the “*BTG Granted*” situation to start). The Context Manager generates the SP of “*BTG Granted*” and sends it to the Situation Manager (i.e., this ends the situation “*Normal*” of the patient’s MD). Likewise, whenever the doctor who requested the BTG finishes the treatment, (s)he can end the “*BTG Granted*” situation. The BTG Manager fulfills the SCD policy obligation by sending a *simple event* to the Context Manager (ordering the “*BTG Granted*” situation to end). The Context Manager generates the EP of “*BTG Granted*” and sends it to the Situation Manager (i.e., this starts the situation “*Normal*” of the patient’s MD again).

## I.D. SCD Architecture

Based on our unified and adaptive architecture presented in the previous chapter, we implement the BTG SCD policy using this architecture and by implementing the following three types of PEP agent (see Figure 6.76).

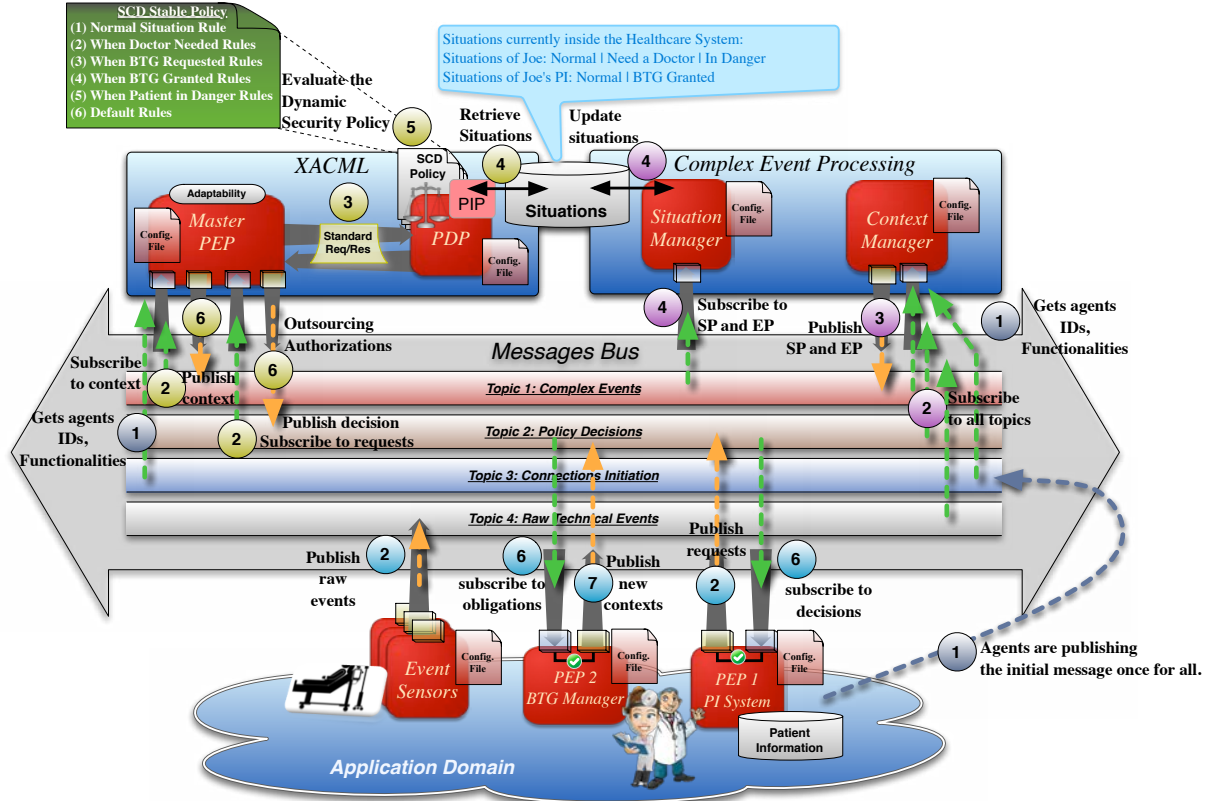


Figure 6.76: Technical implementation of the BTG prototype based on our framework

We remind the global process we presented earlier, where phases are happening in a sequential manner ordered by numbers. The colors distinguish the type of processing. However, having the same number with different colors means that the processing is happening in parallel (for example, the step two in blue, yellow, and violet).

First, all PEP agents publish their initiation message described earlier with their ID and functionality. The Master PEP and the Context Manager collect this information and build two lists holding all the management agents (i.e., the BTG Manager (PEP 2), the PI system (PEP 1) and all the events sensors: fever, pulse, etc.).

### I.D.1. Event Sensors

Event sensors publish their readings of the defined events type to the technical events topic. We configured the sensors to generate events through an automated simulation process. PEP sensors create six streams to be generated with around 20 events in each. Approximately 120 events are from the following six types: Fever, Pulse, Status, Patient Position, Doctor Position and Room Occupancy.

During the detection of events, the Context Manager composes events and listens to both access and BTG requests. Once CE1 (patient in danger) or CE2 (urgent need for a doctor) is detected, the Context Manager sends SP to the Situation Manager that a doctor is needed for the patient (Joe).

Through the management of situation process, the prototype identifies three scenarios based on the exchanged events: there is no situation concerning Joe, Joe is urgently needs a doctor and BTG granted on Joe's PI. This is the final step concerning the situation management loop. The situation manager stores and manages all three situations in the database (i.e., "Normal", "Doctor Needed" and "BTG Granted").

Supposing that software sensors has published readings in step 2 (the blue phase) concerning Joe's health situation. Thus, through the violet phases, the context manager identifies the readings, identifies CE2, and forwards it as SP to the situation manager. The latter gets the SP and stores the corresponding situation: "Doctor Needed" to be used by the PDP.

## I.D.2. BTG Manager

This PEP agent has the ID PEP 2. Its role is to receive BTG requests from doctors and forward them to the Master PEP. Upon any detection of an access or a BTG request, the Master PEP sends a standard XACML request to the PDP to check the SCD policy decision.

Assuming that the sensors detected what are necessary to match CE1 or CE2, then Joe is now in *Doctor Needed* situation. Emma was available to treat Joe and she knows the policy rules. Joe is not her patient, so she knows her need for a specific authorization to engage with his treatment. Emma places a BTG request on PI of Joe to PEP 2.

Upon the acceptance of Emma's BTG request, the Master PEP sends an obligation to PEP 2 to start the *BTG Granted* situation on the MD of Joe. Thus, PEP 2 publishes a new context (stating the start of the BTG period) to the context manager (step 7, blue phase). The context manager forwards the new context to the situation manager about Joe's MD. The situation manager changes the current MD situation to become "BTG Granted" (violet phase, step 4).

Any ongoing BTG request ends through two ways: either the doctor ends the treatment of the patient or the doctor places a BTG end request. If the BTG end request was accepted or a relevant context detected (referring that Emma ended the treatment), the Master PEP obligates PEP 2 to end the BTG. Likewise, PEP 2 sends new context to the context manger (step 7, blue phase). The context manger forwards the new context that states the ending of BTG. The situation manager removes the BTG granted situation from Joe's MD.

## I.D.3. PI System

The PI system is responsible of publishing regular access requests, if any, to the Master PEP. For instance, any doctor can place an access request on his/her patients' PI to PEP 1.

Otherwise, and after accepting the BTG request of an available doctor (Emma) on the PI of Joe, the current situation of Joe's MD should be *BTG Granted*. Emma has now BTG permission and can access PI of Joe. At the blue phase, step 2, Emma publishes a request to access through PEP 1. The yellow phase treats this request as explained from step 2 to 6. Then, the response gets back to PEP 1 (to Emma). Then, PEP receives the decision (blue phase, step 6) and allows therefore Emma to view Joe's PI.

As for the identification of *in danger* situation by the violet phase, it is very similar to the "*doctor needed*" afterward. However, the differences is that Emma can, but not obligated, to request BTG on Joe's MD to PEP 2. Thus, if Emma places in step 2 (blue phase) a view request to PEP 1, the yellow phase will return a *permit* decision from the Master PEP to PEP 1 (yellow phase, step 6). Emma can view directly Joe's PI and at the same time, the BTG manager (PEP 2) receives obligation from the Master PEP (step 6, blue phase). This obligation is caused by Emma access request to PEP 1.

PEP 2 then sends SP of *BTG Granted* to the situation manager (step 7, blue phase). The situation manager changes Joe's MD situation to BTG granted (step 4, violet phase). Likewise, the end of the *BTG Granted* situation is done either manually (Emma places an end request) or automatically (the treatment is finished).

## 1<sup>st</sup> Scenario's Conclusion

---

*Therefore, we proved that our approach enforces dynamic authorization by implementing "Break Glass" scenario. Moreover, we proved that our approach creates the small PREDYKOT loop by automating the management of the situations. We used the policy obligations to adaptably configure the situation manager and control situations, with human intervention (i.e., Emma requesting BTG on Joe's MD during "Doctor Needed" situation) or without any human intervention (i.e., automatically change Joe's MD situation to "BTG granted" during "patient in danger" situation).*

---



## II. 2<sup>nd</sup> Scenario: Workflow Management

The scenario aims at ensuring that the security management of workflows respects the IAM business requirements inside virtual organizations (VOs). VOs are concerned about network and tool evolution that promotes collaborative work. Multipartite projects need to federate experts on different areas and information systems toward completing the common objective. Communications between distributed parties brings many security issues. Under certain circumstances, administrators have to unlock some of the security doors of parties' information systems. During a past European project called VIVACE (Laborde, Kamel, Barrere, & Benzekri, 2007), the work had proposed to add an attribute describing the shared-documents status in order manage the dynamic authorization. Using *situations*, we generalize the proposed solution, which was dedicated to solve the specific use case.

As shown in Figure 6.77, the workflow management scenario deals with security management in VOs and needs dynamic authorizations and configurations. We summarize it as follows:

*People from different aeronautical companies wish to collaborate in producing a technical specification document. Each organization has people with specific skills and software to complete specific tasks. The technical document has to be created by an analyzer, then designed and finally validated. This sequence of tasks is structured by a workflow, controlled by a conductor. The collaboration is formalized by a contract that states which company provides what kind of employees. Each company is responsible for managing its people and no constraints on who is doing the job. The contract also specifies access control policies on the shared documents. During design task, only people with designer role are permitted to access (read/write) the shared documents. The same for the analyzers and validator roles, they access only when their tasks starts. However, any VO members can read the results when all tasks are finished. A workflow engine (WFE) acts as a conductor. During the design task, the engine enables the set of rules that concerns the current document status and when it is completed, workflow engine disables them and enables another. Finally, when the process terminates, the engine informs participants and grant the "read permission" to all roles on the shared documents.*

*Nevertheless, errors may produce during the production of the document. Therefore, the WFE should react to such exceptions. As during the design task, designers can signal error of analyzing. The WFE should return the document to the analysis phase. Likewise, during the validation task, any validator can signal analysis or design errors. Based on the error, the WFE should return the document to the responsible company, and consequently to the relevant phase.*

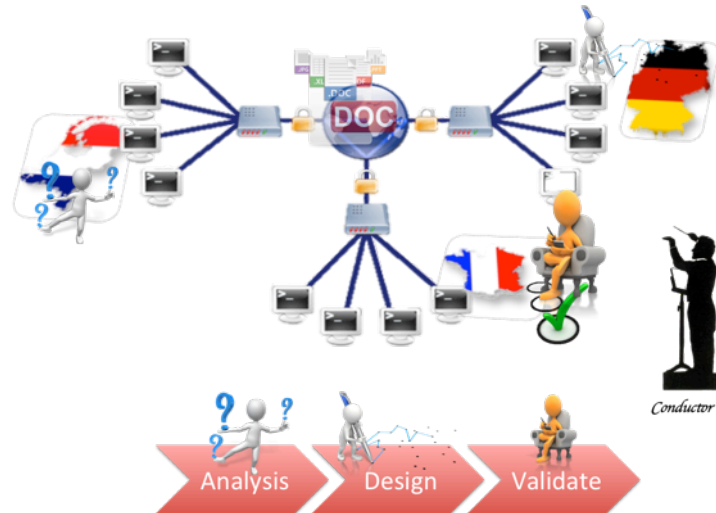


Figure 6.77: Workflow management in the VO scenario for the VIVACE project

## II.A. Modeling Workflow Management Situations

We will demonstrate the state diagram of the VO Workflow Management scenario. The state diagram with association to the workflow management policy is demonstrated in the Figure 6.78. During situation “*await-analysis*” there are two permissions for the employee of the Dutch company with an analyzer role: 1) view and edit the shared document, and 2) finalizing the analyzing task. During the “*await-design*” situation, the permissions for any designer working at the German company are: 1) view and edit the shared documents 2) Notify analyzing errors in the shared documents and email the WFE about them, and 3) Finalizing the design task. During the “*await-validation*” situation, any validator at the French company can: 1) view and edit the shared documents, 2) Notify analyzing and/or designing errors in the shared documents and email the WFE about them, 3) Finalizing the validation task. Situation “*Finish*” permits all employees from the three companies to view the shard documents only. Finally, the function of finalizing the task could be any contact form (e.g., email) with the WFE in order to declare the end of the editing process.

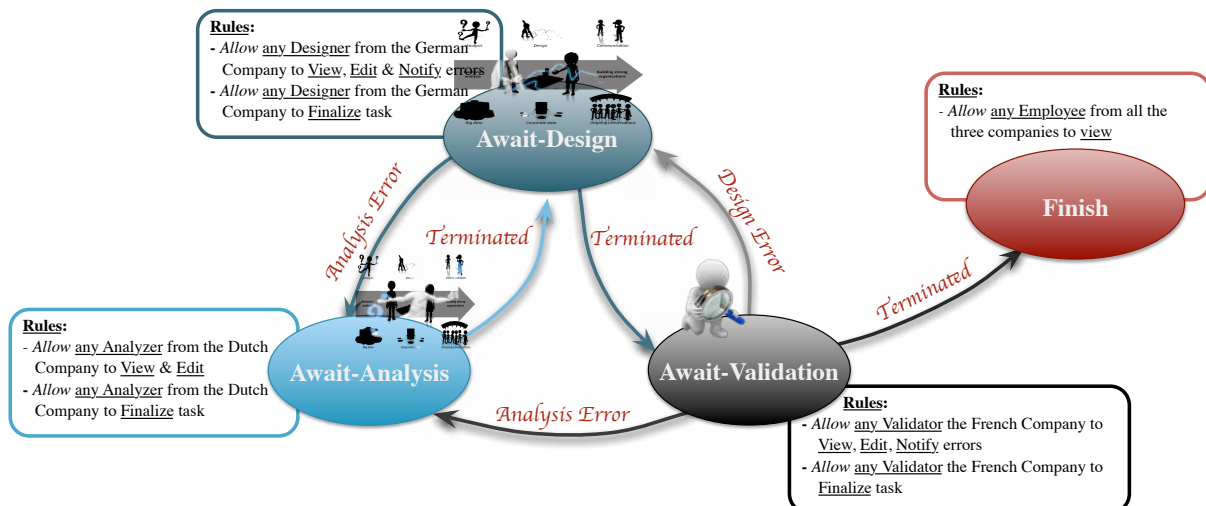


Figure 6.78: State diagram of the Workflow Management



## II.B. Representing SCD Policy

By considering these four situations, the policy expressing the dynamic authorization can be represented by the **five rules** below. *Rules 1, 2 and 3* ensure that a user can access the shared document with read or write permission if his or her role matches the current situation of the shared document represented by its situation (i.e., if he or she is a designer during the await-design situation, if he or she is an analyzer during the await-analyzer situation, and if he or she is a validator during the await-validation situation). The *fourth rule* means that a user can access the shared document with read-only permission if he or she has the role *any* (designer, analyzer, or validator). That is, the rule is selected when the document's situation is identified as *finish*. Finally, to manage conflicts, we declare a default *sixth rule* that denies all other access requests not relevant to the other seven previous rules.

Based on the provided design of situations and their association with contexts and decisions, it is easier to express the XACML policy. However, the expression will follow the presented orientation structure with respect to the Figure 6.78:

**Policy Set:**

**Policy:**

**Targeting** *Situation (Resource)* = "Await-Design" **For**  
**Rule 1:**  
**IF** *Role (Subject)* = *Designer* & *Company (Subject)* = *Germany* &  
*Name (Action)* = *Read / Edit / Finalize*  
**Then** *Permit*

**Targeting** *Situation (Resource)* = "Await-Analysis" **For**  
**Rule 2:**  
**IF** *Role (Subject)* = *Analyzer* & *Company (Subject)* = *Netherlands* &  
*Name (Action)* = *Read / Edit / Finalize*  
**Then** *Permit*

**Targeting** *Situation (Resource)* = "Await-Validation" **For**  
**Rule 3:**  
**IF** *Role (Subject)* = *Validator* & *Company (Subject)* = *France* &  
*Name (Action)* = *Read / Edit / Finalize*  
**Then** *Permit*

**Targeting** *Situation (Resource)* = "Finish" **For**  
**Rule 4:**  
**IF** *Role (Subject)* = *Any* & *Name (Action)* = *Read*  
*Company (Subject)* = *Germany/Netherlands/France* &  
**Then** *Permit*

**Targeting** *Any Situation* **For**  
**Rule 5:**  
**IF** *Any Conditions*  
**Then** *Deny*



## II.C. Identifying Workflow Management Situations

We have implemented our Workflow Management to prove the concept of dynamic management of workflow using our situation-orientation approach. The ESPER engine will detect situations and monitor employees' activities on the shared documents. The engine will react on situations by updating the values in database. We have defined a set of six types of events with ESPER. Events types are:

**Table 6.12: Supported VO event types that the context manager can identifies.**

Event Type	Event Description
<b>DocStatEvt</b>	Document Status,
<b>AnlystActEvt</b>	Analyzer Activities,
<b>DsinrActEvt</b>	Designer Activities,
<b>VldtrActEvt</b>	Validator Activities,
<b>AccRqEvt</b>	Access Request,
<b>SitEvt</b>	Situation Detection

There are two complex event types that aggregate events from the mentioned event types. In order for the complex event to be triggered, it should meet the conditions of specific Esper EPL.

- ⌘ **CE1 – Analysis Errors:** the participant event types to this complex event are analyzer, designer and validator activities. The CE1 is identified whenever the designer submits an analysis error (or more) on a shared document and that an analyzer receives back this shared-document. Moreover, the CE1 is identified also whenever the validator submits an analysis error (or more) on a shared document and that an analyzer receives back this shared-document.

```

select *           -- it is the await-analysis SP, but also the EP of the current situation
from               EventAnalyzerActivities.std:lastevent() as AnlystActEvt,
                    EventDesignerActivities.std:lastevent() as DsinrActEvt,
                    EventValidatorActivities.std:lastevent() as VldtrActEvt,

where              (DsinrActEvt.activity = 'SubmitAnalysisError' and
                    AnlystActEvt.activity = 'RecivedDocumentWithErrors' and
                    DsinrActEvt.docID = AnlystActEvt.docID) or
                    (VldtrActEvt.activity = 'SubmitAnalysisError' and
                    AnlystActEvt.activity = 'RecivedDocumentWithErrors' and
                    VldtrActEvt.docID = AnlystActEvt.docID)

```

- ⌘ **CE2 – Design Errors:** the participant event types to this complex event are designer and validator activities. The CE2 is identified whenever the validator submits a design error (or more) on a shared document and that a designer receives back this shared-document.

```

select *           -- it is the await-design SP, but also the EP of the current situation
from               EventDesignerActivities.std:lastevent() as DsinrActEvt,
                    EventValidatorActivities.std:lastevent() as VldtrActEvt,

where              VldtrActEvt.activity = 'SubmitDesignError' and
                    DsinrActEvt.activity = 'RecivedDocumentWithErrors' and
                    VldtrActEvt.docID = DsinrActEvt.docID

```

Giving the events in Table 6.12 and the previous two complex events, we can define the SPs and EPs for each situation as following:

- SP for the situation “Await-Analysis”:

- Whenever an analyzer is allowed to create a shared-document, the Master PEP should start this situation. (AccRqEvt)
- Whenever the context manager identifies a CE1.
- EP for situation “*Await-Analysis*” and SP for the situation “*Await-Design*”:
  - Whenever an analyzer requests finishing the analysis task. (AccRqEvt)
  - Whenever an analyzer labels the shared-document as finished (AnlystActEvt).
  - Whenever the context manager identifies a CE2.
- EP for the situation “*Await-Design*” and SP for the situation “*Await-Validation*”:
  - Whenever a designer requests finishing the analysis task manually. (AccRqEvt)
  - Whenever a designer labels the shared-document as finished (DsinrActEvt).
- EP for the situation “*Await-Validation*” and SP for the situation “*Finish*”:
  - Whenever a validator requests finishing the analysis task manually. (AccRqEvt)
  - Whenever a validator labels the shared-document as finished (VldtrActEvt).

## II.D. SCD Architecture

Using our unified and adaptive architecture, the workflow management implementation passes by the same six parallel steps and during three phases: yellow, violet and blue (in addition to the initial phase and its steps). Likewise, we repeat the first initiation phase, as it is the same for all scenarios. However, this scenario has same essential agents that self-identify themselves to new agents: VO PEP in each country that controls the access to shared documents: Germany, France and Netherlands).

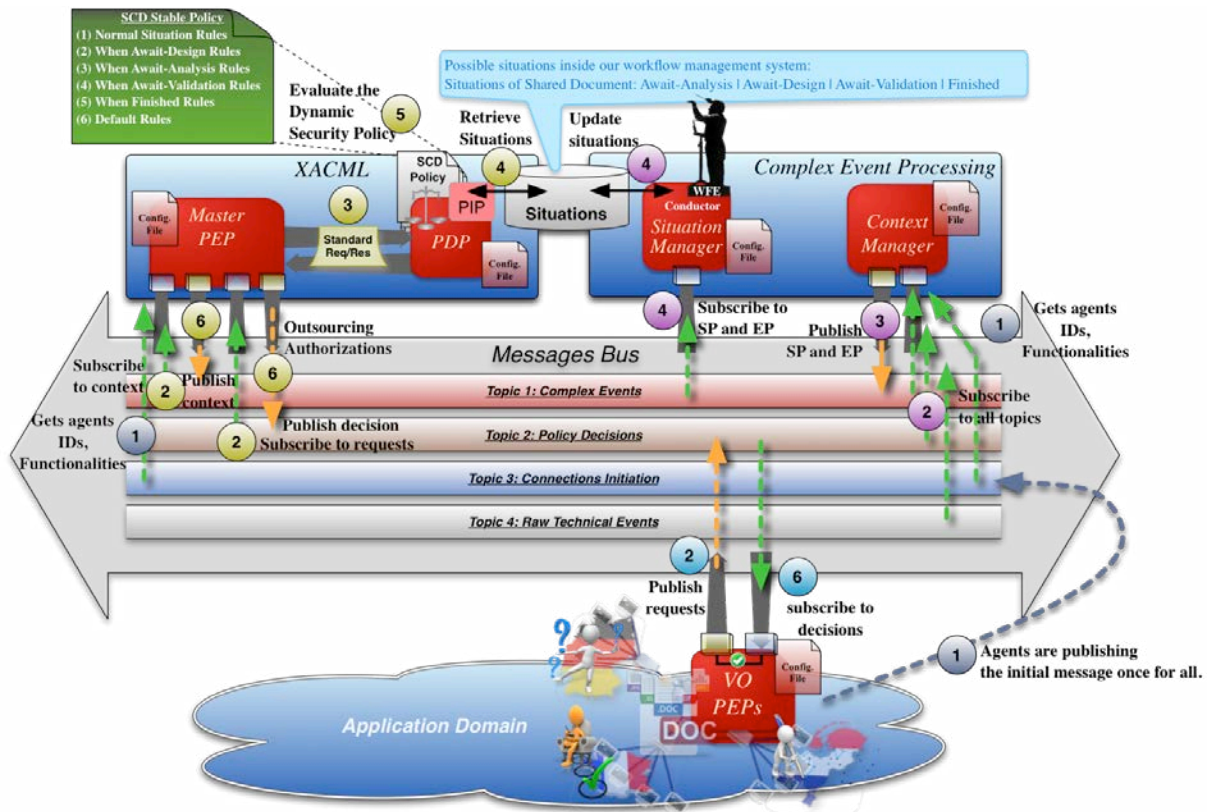


Figure 6.79: Technical implementation of the Workflow Management prototype based on our framework

VO PEP agents send, receive and enforce the access requests on the shared documents. Every VO PEP agent publishes an initiation message described earlier with an ID and functionality(ies). The

VO PEPs start publishing access requests, if any. For instance, a designer places an access request on a shared document to the VO PEP.

The Context Manager composes events and listens to access requests going to the VO PEPs. Once the CE1 detected by the Context Manager, it forwards an SP to the Situation Manager stating that an analysis error is discovered on one of the shared documents. On the other hand, the detection of CE2 by the Context Manager results in forwarding an SP to the Situation Manager stating that a design error is discovered on one of the shared document.

Analyzers, designers and validators can present an access request to the responsible VO PEP in their company in order to process shared documents. At the same time, the situation manager stores and manages all four situations in the database (i.e., “*await-analysis*” “*await-design*”, “*await-validation*” and “*Finish*”). Through the management of situation process, the prototype identifies several possibilities:

- Any created shared document is currently either in analysis, design or validation task. Thus, this shared-document’s situation is *await-analysis*, *await-design*, or *await-validation*. However, the transition from one to another is detected automatically by the Situation Manager (WFE) once the Context Manager identify that one of the roles finalize a task. For example, the analyzer requests finalizing the analysis on a shared document. Once the analyzer is permitted, the Context Manager detects the analyzer activity and sends SP for *await-design*, which is also EP for the *await-analysis*.
- Any document in the *await-design* situation may be returned to the analysis task (i.e., after receiving CE1 from the Context Manager) and therefore becomes in *await-analysis* situation.
- Any document in the *await-validation* situation may be returned to the analysis or the design tasks (i.e., after receiving the CE2) and therefore becomes in *await-analysis* or *await-design* situations.
- Any document in the *await-validation* situation may be finalized by the validator and therefore becomes in the situation *finish*.

Based on their subscription rules, the VO PEPs retrieve messages concerning authorizations. These smart nodes enforce them on the accesses requestor (to be an analyzer, designer or validator) by allowing or denying them to process shared-documents based on their situations.



## 2<sup>nd</sup> Scenario's Conclusion

---

*This scenario illustrates yet another added value of our framework. Beside the features of the first scenario in managing dynamic authorization and configuring the situation manager, we presented a simple implementation for such complex scenario. We were able to easily and externally manage complex transition between situations without any complexity on the security policy. Furthermore, we keep the security policy dynamic in term of giving decisions oriented by these situations. Moreover, externalizing the complexity of situations design prevents having effects on the policy management and simple expression. That is, keeping the complexity of expressing the semantics inside the policy effects the policy management through modifying the policy rules. In addition, it removes the expression simplicity through expanding conditions to reach the expected semantics (see Chapter 4).*

*In summary, we managed the workflow of tasks in parallel with the management of security. We use the outcome of tasks changing by orienting the security policy toward expected authorization decisions.*

---

## III.3<sup>rd</sup> Scenario: Job Submission

The management of energy consumption in order to reduce CO2 emission, costs, etc. is a management best-practices to optimize energy usage inside information systems. Managing security, while applying these practices, entails not only security requirements, but also obligations on energy usage. We consider applying our framework to such type of scenarios is a challenge. Succeeding this challenge is an added value and additional evidence of the generic nature of the thesis framework.

We choose to manage security of the job submission process on grid-computing infrastructure as part of a cooperative project<sup>34</sup>. Indeed, we obligate the respect of energy management best practices during managing security. As a result of our choices, we need to manage authorizations through synchronized communications (outsourcing) and to manage configurations through asynchronized communications (provisioning). Therefore, we think this scenario is a good choice also to prove *unification* feature of our architecture.

To prove the unity of our contribution, we defined this use case where we apply the proposed language and architecture. The scenario represents a sort of heterogeneity in requirements: security and energy. Expressing the SCD policy for both requirements reflect its flexibility and generic nature. Moreover, the scenario consists of highly dynamic environments where *situations* are complex and a matter of frequent changes. Handling these changes highlights the consideration of *dynamicity* in our architecture. On the other hand, applying different configurations (with considering authorizations) dynamically highlights its adaptive and unification natures. The use case is regarding access and energy management in grid computing, where only authorized users can submit jobs to the submission service to be launched on authorized servers during authorized time-slots and with respect to energy saving recommendations. Our scenario is the following:

*Authorized employees launch their jobs on authorized servers every day and night. Updating, migrating, and rebuilding the index of databases, the data warehouse statistics operations, and other heavy jobs are launched on several types of servers, during: daylight or nightshift periods.*

*The job scheduler is an external service that receives, organizes and manages the submission and the execution process. In certain contexts, jobs are not authorized to start on specific servers. The security management system should be dynamic enough to identify and manage such contexts. For instance, in term of reducing energy consumption, a long-duration job XYZ with more than six hours of execution time will consume the servers during the daylight. This may affect the execution of other more important jobs. Hence, the job scheduler should check the possibility (e.g., priority) of postponing the launching of this job to the nightshift period (i.e., if the submitter accepts, if the job priority is low, etc.).*

*Launching jobs at night consume less energy in most European countries. Therefore, the job scheduler verifies each time the possibility of postponing any job launching to the nightshift period.*

---

<sup>34</sup> This project is initially proposed as part of a cooperative work between IRIT, France and DERI, Ireland. It is partially funded by EU ITEA2 under the project 10104 PREDYKOT and the Lion II project supported by Science Foundation Ireland under grant number SFI/08/CE1/I1380 Lion-2.

### III.A. Modeling the Job Submission Situations

Any keyword, i.e., has a significant appearance or importance, in the documents is recognized as a *situation* if it is possible to define conditions and circumstances (aggregation of phenomena) about a specific subject. This should declare the start time-slot of it. It has definitely an end of its duration by also conditions and circumstances (aggregation of phenomena). Therefore, we identify two situations from the scenario where the policy decision should be dynamic: “*cheap*” and “*expensive*” periods. The “*cheap period*” has significant importance and bounded with contexts that declare its start (SP) and its end (EP) (i.e., nightshift context starts at 8 pm and ends before 7 am). The “*expensive period*” is therefore the inverse context (i.e., dayshift starts at 7 am and ends before 8 pm).

Based on these two situations, we will demonstrate the state diagram of the job submission management scenario. The state diagram with association to the job submission policy is demonstrated in the Figure 6.80. We can always identify one-and-one policy (i.e. expressed in XACML as a rule, a policy, a policy set) associated for each situation. Hence, during the situation “*expensive period*” all clients with the role employee are allowed to submit jobs. However, the submitted jobs during the day context are only launched if they meet the following configurations: 1) jobs must have high or normal priority 2) in term of performance, jobs must be with modern or light size of tasks and processes, and 3) jobs must have short or modern duration for their estimated execution time. Otherwise, if none of these configurations are met, the job is submitted also but the execution is postponed to the night (i.e., night-launching-possibility context).

Now, the situation “*cheap period*” permits clients with role employee to submit jobs without any constraints on their configurations. Thus, any submitted job is allowed for launching directly.

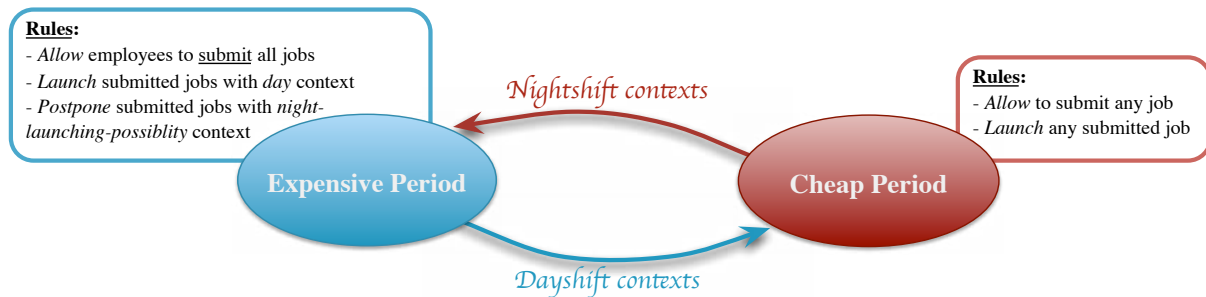


Figure 6.80: State diagram of the Job Submission in grid-computing environment



## III.B. Representing SCD Policy

By considering these two situations, the policy expressing the dynamic authorization and management of the submission process can be represented by the **four rules** below. The *rules 1, 2 and 3* ensure that clients (authorized employees) can submit jobs on servers during day and night. However, the *first rule* entails that clients' jobs are launch during daylight if their jobs respect certain contexts (i.e., job has normal/high priority, light/moderate size and short/moderate execution duration). Otherwise, the *second rule* postpones this job to be launched at night (i.e., at the beginning of the nightshift duration, for instance, at 8 pm). The *third rule* entails that clients' jobs are launched during nightshift within any context. Finally, to manage conflicts, we declare a default *fourth rule* that denies all other submission requests not relevant to the previous three rules.

Based on the provided design of situations and their association with contexts and decisions, it is easier to express the XACML policy. However, the expression will follow the presented orientation structure with respect to the Figure 6.80:

**Policy Set:**

**Policy:**

**Targeting** *Situation (Environment)* = "Expensive" **For**

**Rule 1:**

**IF** *Role (Subject)* = *Employee* & *Type (Resource)* = *Job* & *Type (Action)* = *Submit*  
*Size (Job)* = *Light/Moderate* & *Priority (Job)* = *Normal / High* &  
*Duration (Job)* = *Short / Moderate*  
**Then** *Permit*

**Take Advice:** *Find the Job-Launcher Bundle at the following (file path)*

**Apply** *Obligation:* *Launch the job at (the specified time in the submission request)*

**Targeting** *Situation (Environment)* = "Expensive" **For**

**Rule 2:**

**IF** *Role (Subject)* = *Employee* & *Type (Resource)* = *Job* & *Type (Action)* = *Submit*  
*Size (Job)* = *Heavy* & *Priority (Job)* = *Low* & *Duration (Job)* = *Long*  
**Then** *Permit*

**Take Advice:** *Find the Job-Launcher Bundle at the following (file path)*

**Apply** *Obligation:* *Launch the job at the nightshift SP*

**Targeting** *Situation (Environement)* = "Cheap" **For**

**Rule 3:**

**IF** *Role (Subject)* = *Employee* & *Type (Resource)* = *Job* & *Type (Action)* = *Submit*  
*Size (Job)* = *Any* & *Priority (Job)* = *Any* & *Duration (Job)* = *Any*  
**Then** *Permit*

**Take Advice:** *Find the Job-Launcher Bundle at the following (file path)*

**Apply** *Obligation:* *Launch the job at (the specified time in the submission request)*

**Targeting** *Any Situation* **For**

**Rule 4:**

**IF** *Any Conditions*

**Then** *Deny*



### III.C. Identifying the Job Submission Situations

We have implemented our job submission to use our framework in managing the security of a pure management-oriented (configuration-oriented) scenario. The ESPER engine should detect situations and monitor clients' submissions to the Master PEP (i.e., its role is also a job scheduler as it analyzes jobs and plans new timing for jobs). The engine reacts on situations by updating the values in database. We have defined a set of four types of events with ESPER (Table 6.13).

**Table 6.13:** *Supported grid-computing event types that the context manager can identifies.*

Event Type	Event Description
<b>TDEvt</b>	Time & Date Readings,
<b>SubRqEvt</b>	Submission Requests,
<b>LaunchActEvt</b>	Launching Activities,
<b>SitEvt</b>	Situation Detection

We want to stress in this final scenario that the situation gives semantics. The context to be at day or at night is not what gives the meaning of paying less or more. It is 1) choosing the right expression from the requirements “*cheap*” and “*expensive*”, 2) defining what causes the identification of *cheap* and *expensive* energy usage costs in each country (finding the surrounding contexts of EPs and SPs), 3) correctly represent and use the meaning of *cheap* and *expensive* inside the SCD policy to group rules, and finally efficiently apply this meaningful-contained SCD policy using PEP agents. Therefore, at each of the three scenarios, we used four sections match these four steps.

There are two complex event types that aggregate events from the mentioned event types. In order for the complex event to be triggered, it should meet the conditions of specific Esper EPL.

**CE1 – Beginning of the “*Expensive Period*”:** this complex event should contain all contexts that cause the cheap period to start. For simplicity, we have chosen only the dayshift context. The dayshift context is defined by the first working hour (i.e., as it depends the country, we consider in this scenario 7:00 am is the first hour of the day). However, one can imagine other contexts to identify the cheap period. Thus, the CE1 and the next CE2 are flexible in term of defining their queries.

```

select *      -- it is the expensive SP, but also the cheap EP as well
from          TimeDate_Events.std:lastevent() as TDEvt,
where         TDEvt.roundFloor('hour') == '7' and TDEvt.roundFloor('min') == '00'

```

**CE2 – Beginning of the “*Cheap Period*”:** this complex event should contain all contexts that cause the cheap period to start. For simplicity, we have chosen only the nightshift context. The nightshift context is defined by the first non-working hour (i.e., as it depends the country, we consider in this scenario 8:00 pm is the first hour of the night).

```

select *      -- it is the cheap SP, but also the expensive EP as well
from          TimeDate_Events.std:lastevent() as TDEvt,
where         TDEvt.roundFloor('hour') == '20' and TDEvt.roundFloor('min') == '00'

```

### III.D. SCD Architecture

Using our unified and adaptive architecture, we are able to implement the management of the job submission through passing by seven parallel steps and during three phases: yellow, violet and blue (in addition to the initial phase and its steps).

In Figure 6.81, there are several agents that we need to implement as PEPs and to integrate in our presented unified and adaptive architecture. The first agent is the *executer*. It executes jobs on available servers of the grid-computing infrastructure. These configurations express obligations to be respected while launching in term of time, allocated resources, etc. The second type of agent is the client PEP agent. Agent of this type sends submission request with their jobs details in order to get their jobs executed. The third type of PEP agent is sensor agent. The sensors collect information, for instance, about the current time and date, if the job execution is finished so results will be send to the corresponding client who submitted the job<sup>35</sup>.

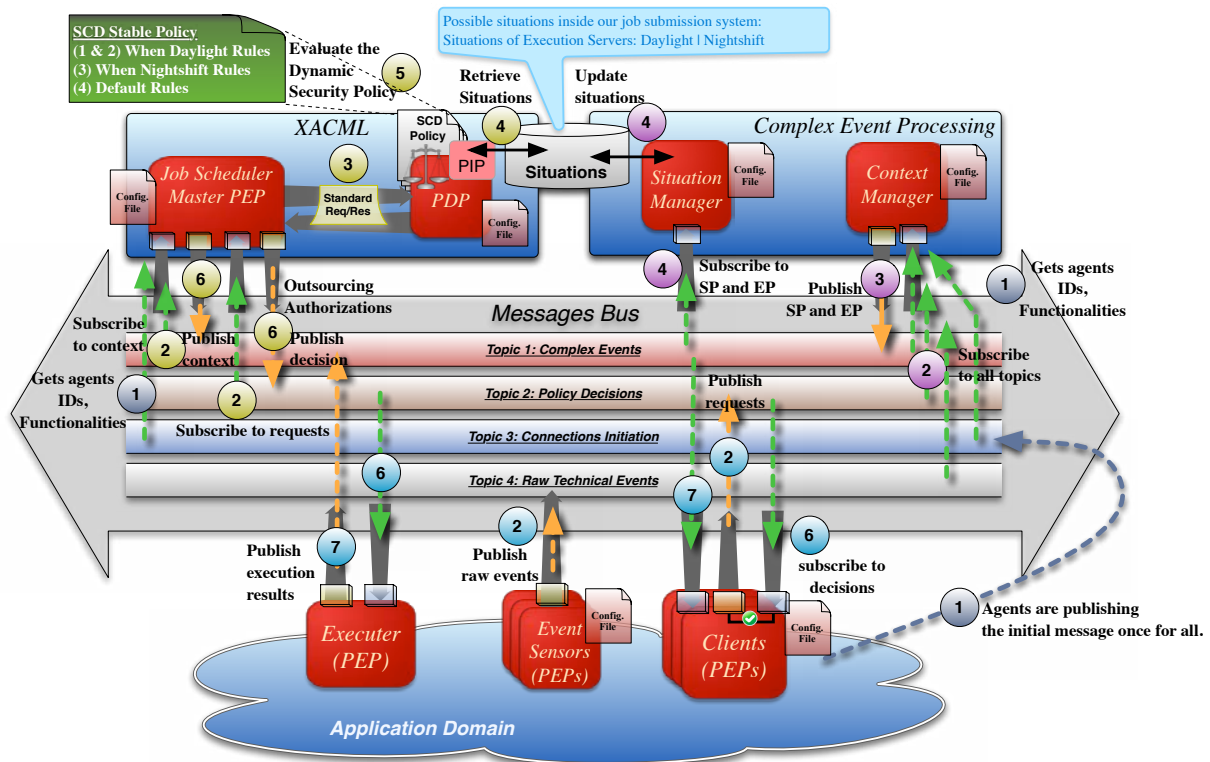


Figure 6.81: Technical implementation of the job submission prototype based on our framework

All the three types of PEP agent publish their initiation message described earlier with their ID and functionality(ies). The Master PEP and the Context Manager collect this information and build two lists holding all the management agents.

<sup>35</sup> We also use sensors for other objectives as well such as collecting information about the grid-computing servers status (overloaded, etc.). However, for the simplicity of the scenario, we were limited on stating only what proves the concept of the architecture generic nature.

### III.D.1. Event Sensors

After the event sensors start publishing their readings, the Context Manager identifies complex events by composing events and listening to submission requests. Once the CE1 or CE2 detected by the Context Manager, it forwards an SP to the Situation Manager stating that the *expensive* or the *cheaper* period has started. The situation manager stores and manages all two situations in the database.

Moreover, event sensors monitor the execution process of each job. They play the bridge between the Context Manager and the client once an authorization decision of launching the job is given.

### III.D.2. Job Submission Clients

Users submit their job through PEP Clients, which generate submission requests and publish them to the Master PEP. For instance, a user wants to submit a job for execution during the dayshift context with certain settings (e.g., high priority). Thus, if the current situation is *expensive period*, the Master PEP allows the submission. However, if the job meets the *night-launching-possibility* context, the Master PEP obligates the executor to launch the job during the *nightshift* context. The clients only receive only the authorization decision of approving or denying the submission.

### III.D.3. Job Executor

The executor PEP receives jobs from the Master PEP. It executes these jobs if there is no *postpone* obligation. Otherwise, it creates a timer for postponed jobs. Once the timer informs the starting of the nightshift context, the executor launches the postponed jobs.

After the executor finishes executing the job, it sends the results to the specified destination inside the job settings. The client who submitted the job is notified by email.

## 3<sup>rd</sup> scenario's Conclusion:

---

*We assume that implementing this scenario provides a good contribution to the energy management in terms of security management and power consumption saving. In term of expression, our contribution does not address rules management (i.e., add, modify and remove, when situations change). As we saw, a good system analysis avoids rewrite or modify the policy, and therefore keep it stable. This stable policy is enforced in a dynamic architecture. Based on this architecture, evaluating the rules provide dynamic management decisions that ensure security and save energy. By implementing this architecture, we demonstrated how it is possible and simple to provide dynamic authorization and configure energy saving obligations with very simple policy.*

*Briefly, we presented a security and energy management solution, particularly considering grid computing. We observed and evaluated how the impact of the system's application comes out with more energy saving and high performance. The scenario implementation does control the submission process and the usage of energy resources. As a result, we used our framework to solve a pure management-oriented scenario and that requires essentially configurations to ensure the management of security.*

---



## IV. Framework Evaluation

This thesis was part of a European project, where the main objective was to propose a proof of concept regarding a dynamic authorization architecture supporting the move from one configuration to another when events occur. Therefore, we focus on the use case studied in Chapter 6 that intend to demonstrate the methodology of expressing and implementing our framework. Through the first scenario, we successfully managed to test the performance of our framework regarding its essential propositions by calculating the average, minimum and maximum time to: 1) situation identification and storage, 2) providing an authorization decision (deny or permit), and 3) applying an obligation from the PDP by the Master PEP that requires additional bundles.

To realize these tests, we used a MacBook Pro laptop configured with 2.4 GHz, core i5 processor and 8 Gb of RAM. We ran the performance testing on top of a virtual machine using Ubuntu Linux operating system (version 10). This VM is configured with 3 Gb of RAM and using 80+ of the execution capacity of the actual machine. We evaluated this test using the running sample data included inside an Excel file (see the Appendix). As a result of this evaluation, we calculated the average processing time of each one of the three calculation points, the minimum processing time registered and the maximum one. The total average time to process each point is approximately 427 milliseconds.

We explain our test scenarios for each evaluation point based on the following policy:

**XACML Policy:**

**Targeting** *Situation (MD-Patient)* = “Normal” **For**

**Rule 1:**

**IF** *Type (Resource)* = MD-Patient  $\wedge$  *Role (Subject)* = Doctor  $\wedge$   
*Type (Action)* = Access  $\wedge$  *ID (Subject)*  $\in$  PRPL (*Resource*)  
**Then** Permit

**Targeting** *Situation (Patient)* = “Doctor Needed” **For**

**Rule 2:**

**IF** *Role (Subject)* = Doctor  $\wedge$  *Type (Resource)* = MD-Patient  $\wedge$   
*Type (Action)* = BTG Request  
**Then** Permit  
**Take Advice:** Find the Start-BTG Bundle at the following (file path)  
**Apply Obligation:** Start the BTG Situation of the resource

**Targeting** *Situation (MD-Patient)* = “BTG Granted” **For**

**Rule 3:**

**IF** *Role (Subject)* = Doctor  $\wedge$  *Type (Resource)* = MD-Patient  $\wedge$   
*ID (Subject)* = BTG Requester (MD-Patient)  $\wedge$  *Type (Action)* = view  
**Then** Permit  
**Take Advice:** Find the Logging Access Bundle at the following (file path)  
**Apply Obligation:** Start logging activities of the relevant doctor

**Targeting** *Situation (MD-Patient)* = “BTG Granted” **For**

**Rule 4:**

**IF** *Role (Subject)* = Doctor  $\wedge$  *Type (Resource)* = MD-Patient  $\wedge$   
*ID (Subject)* = BTG Requester (MD-Patient)  $\wedge$  *Type (Action)* = End BTG  
**Then** Permit  
**Take Advice:** Find the End-BTG Bundle at the following (file path)  
**Apply Obligation:** End the BTG Situation of the resource

**Targeting** *Any Situation* **For**

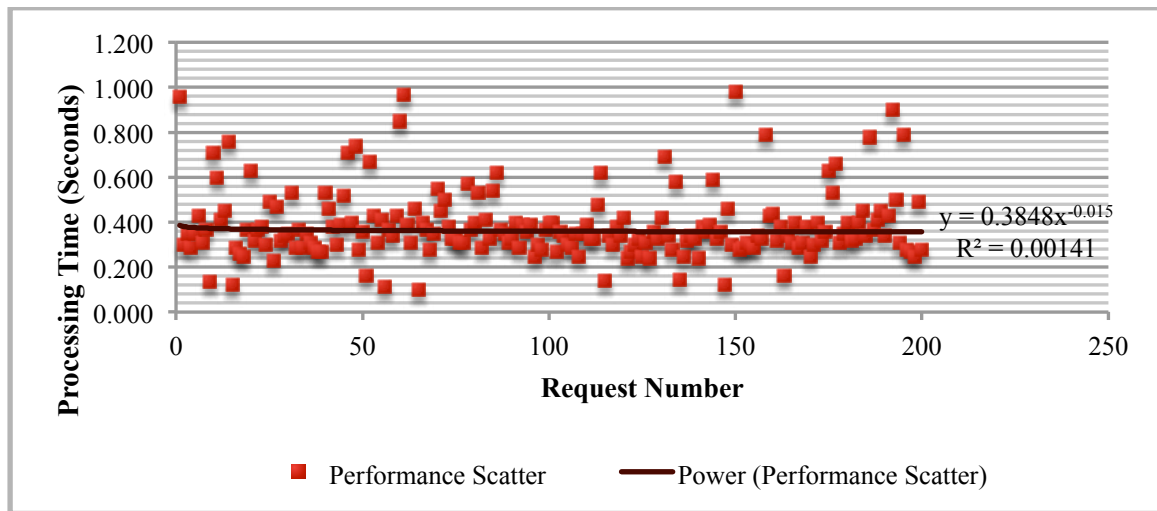
**Rule 5:**

**IF** Any Conditions  
**Then** Deny

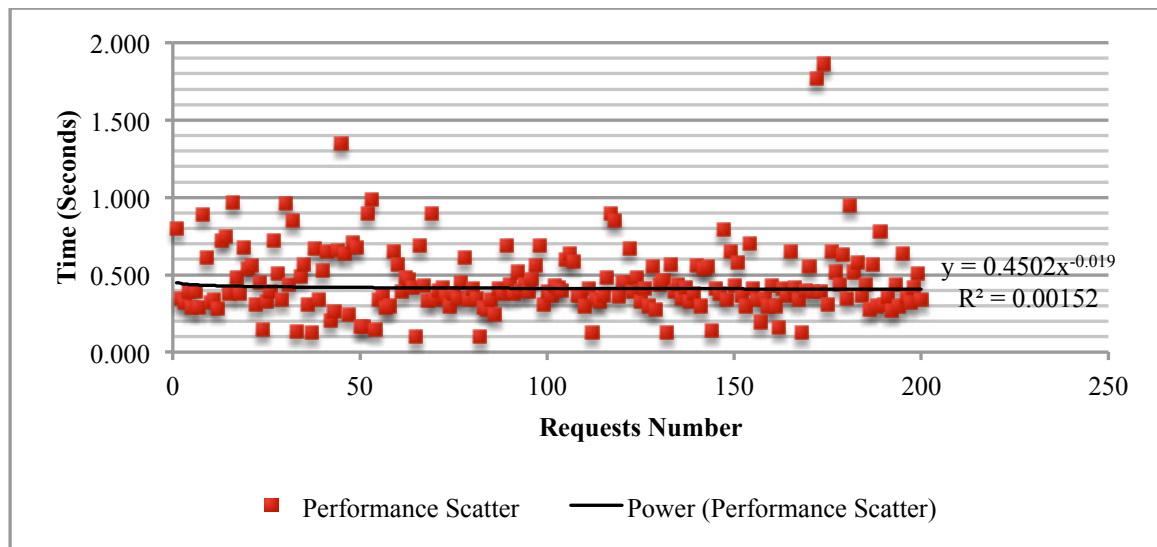
At the first point, we want to test the performance of our architecture through handling requests concerning the provided policy. Therefore, we choose to test the response time for 200 requests by evaluating two rules: 1 and 5. Matching rule 1 conditions gives the “Permit” permission. However, we intentionally reach the rule 5 matching by providing contradict values with rule 1 to get the “Deny” permission. Thus, we evaluate the response of our architecture for both types.

**Table 6.14: Performance testing of the authorizations queries (200 Sequence)**

Authorization Type	Average	Minimum	Maximum
Rule 1: Permit Permission	0.462	0.101	1.866
Rule 5: Deny Permission	0.387	0.100	0.980



**Figure 6.82: Rule 1: Testing 200 requests with Permit permission**



**Figure 6.83: Rule 5: Testing 200 requests with Deny permission**

At the second point, we want to test the performance of our configuration mechanism. As a recall, our configuration process inside the Master PEP passes by four steps: 1) Retrieving the PDP decision, 2) extracting the obligation part from the decision, 3) verifying if the requested service is supported by the Master PEP, 4) finally locating the corresponding bundle to use. The final step is the most interesting step to test in order to calculate the necessary time to apply configurations. The step four has two cases: either the required bundle exists already, then the Master PEP should only install

the bundle and apply it, otherwise the required bundle does not exist locally and the Master PEP should download the bundle to the local repository, then install and apply the bundle. We tested this operation to evaluate the performance with sequence of 50 times. We have for the BTG scenario three bundles. So, we applied this test on each one (BTG-Start, BTG-End, and Logging Access). We explained earlier in the thesis the functionality of each one.

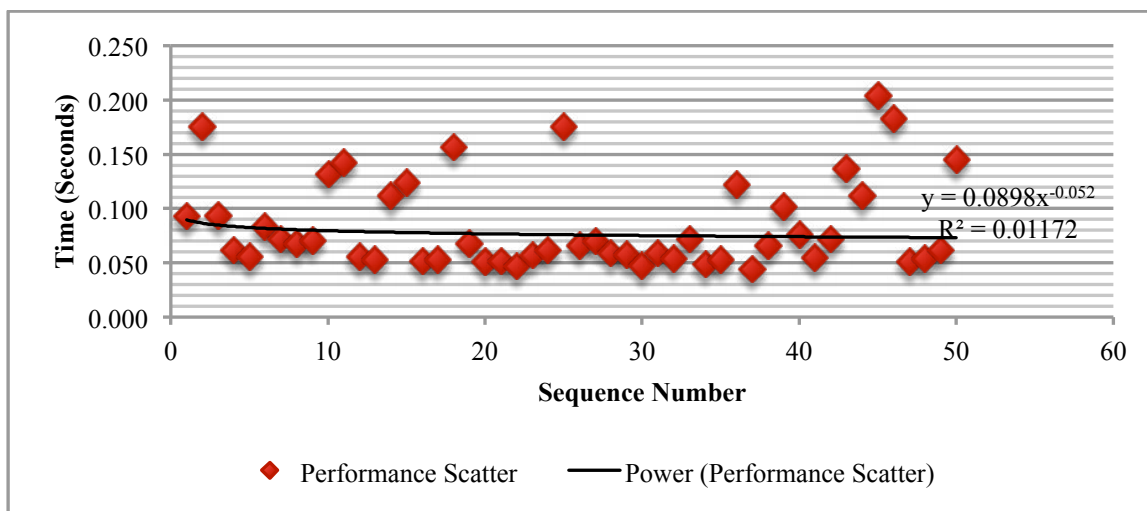


Figure 6.84: BTG-End Bundle: Ready to apply

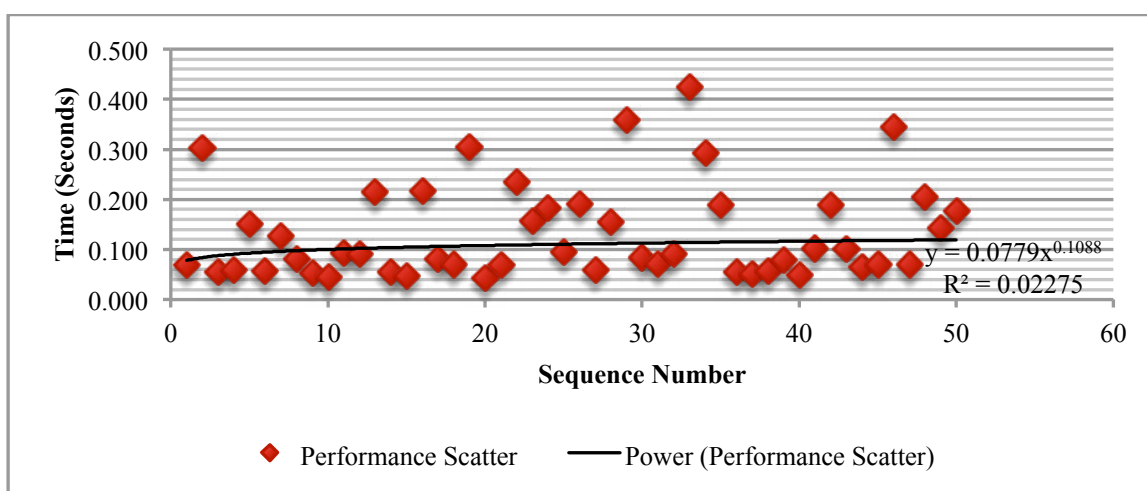


Figure 6.85: BTG-Start Bundle: Ready to apply



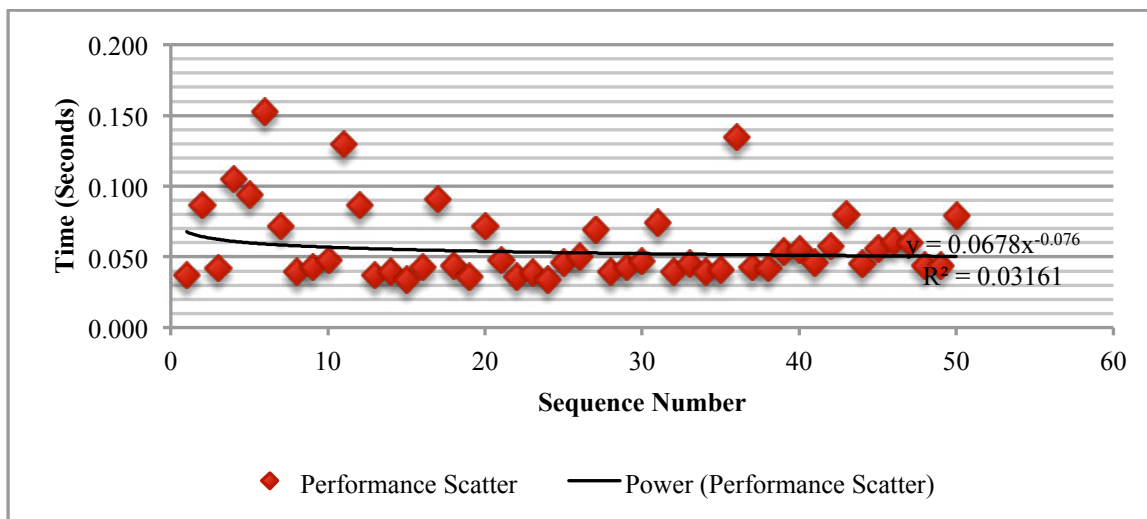


Figure 6.86: *Logging Bundle: Ready to apply*

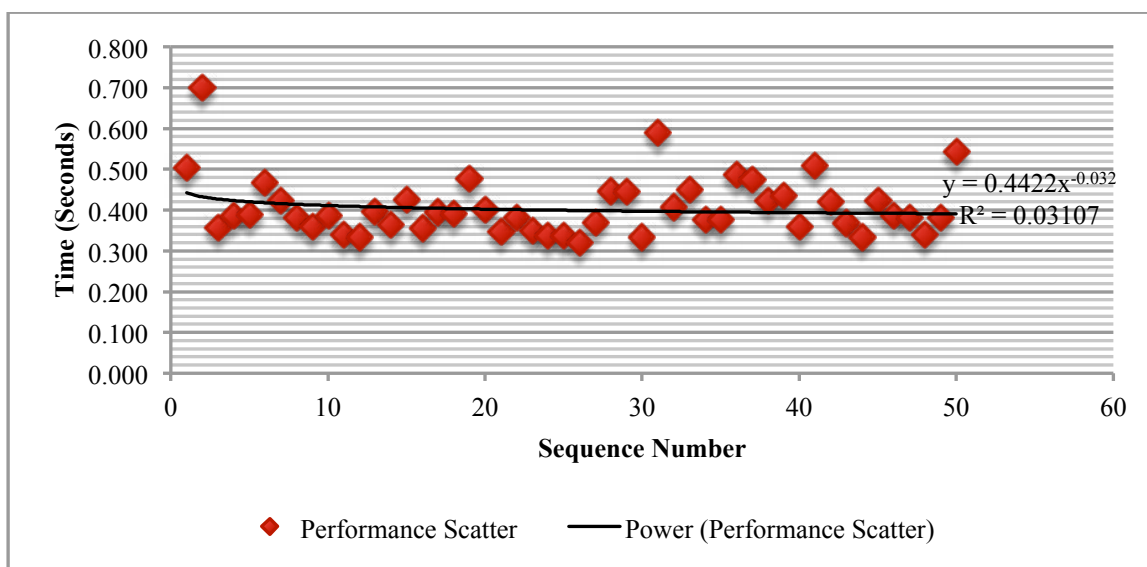


Figure 6.87: *Logging Bundle: Download & Apply*

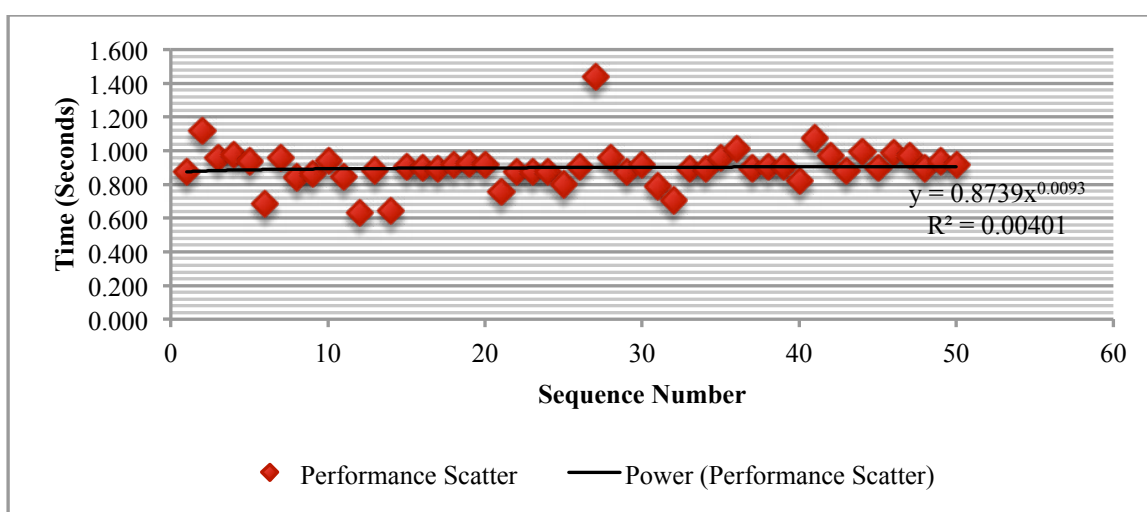


Figure 6.88: *BTG-End Bundle: Download & Apply*

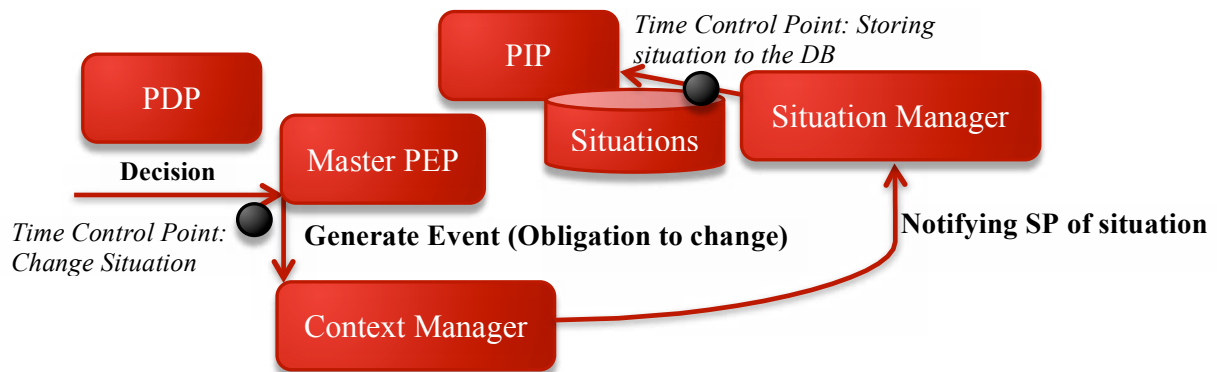


By evaluating the presented figures, we can calculate the average, minimum and maximum processing time (in milliseconds) required to use each bundle (i.e., through both cases).

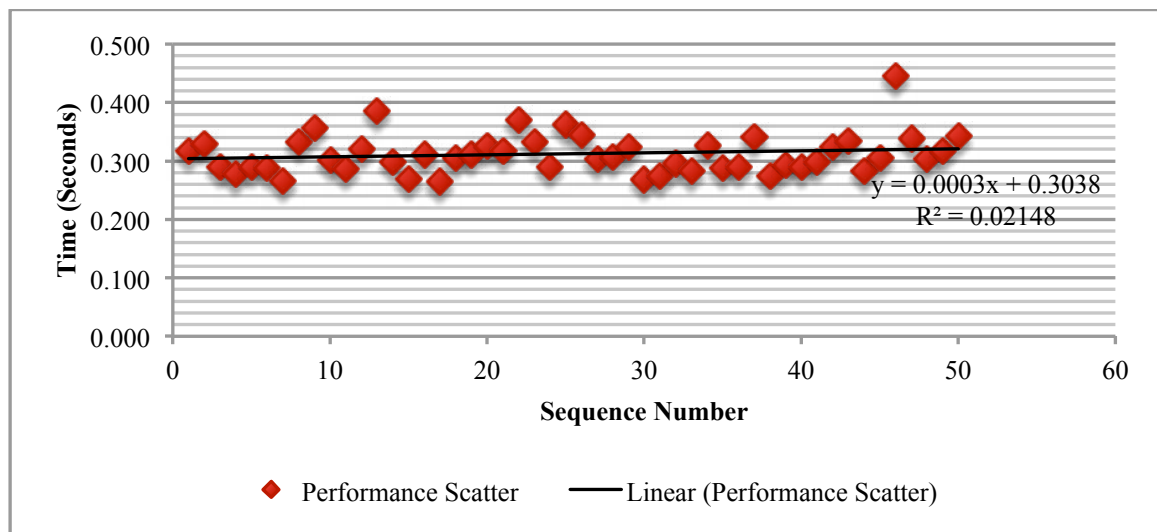
**Table 6.15: Performance testing of the configuration mechanism (50 Sequence)**

Configuration		Average	Minimum	Maximum
Name	Application Method			
Logging Access	Install	0.059	0.034	0.153
	Download & Install	0.408	0.321	0.702
BTG Start	Install	0.133	0.044	0.426
	Download & Install	0.878	0.506	1.202
BTG End	Install	0.085	0.044	0.204
	Download & Install	0.906	0.630	1.441

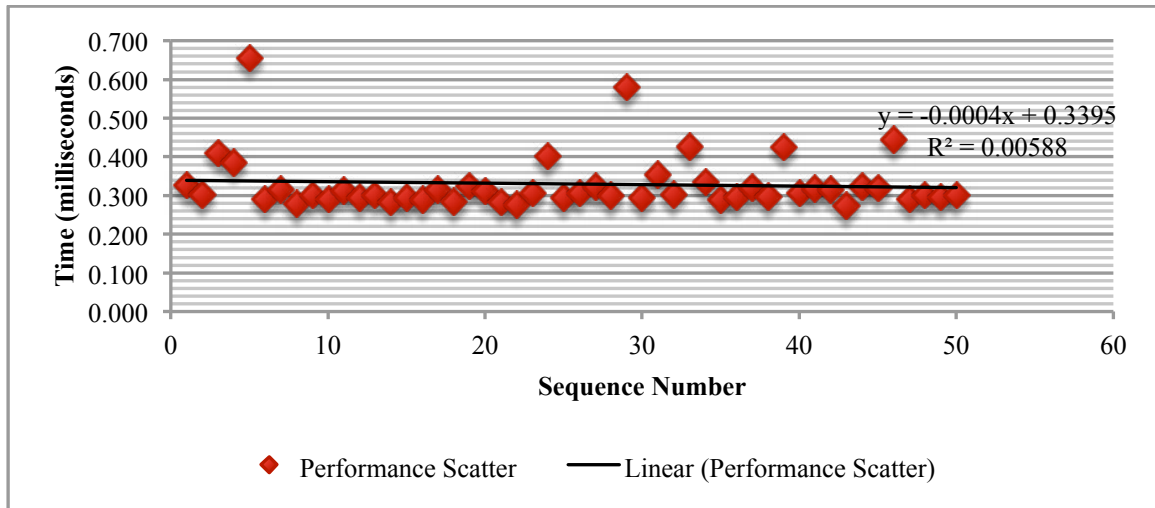
At the third and final point, we want to calculate the beginning of situations and the necessary time needed to process their outbreak (storing the situation in order to be used by the PDP decisions). Thus, we calculate the main event participating in creating the necessary context to start a situation (BTG-Granted, Normal, and Doctor-Needed). In our scenario, we have two points to calculate these contexts.



**Figure 6.89: Performance control points to measure the processing time for obligations going to the SM**



**Figure 6.90: Situation: BTG Granted**

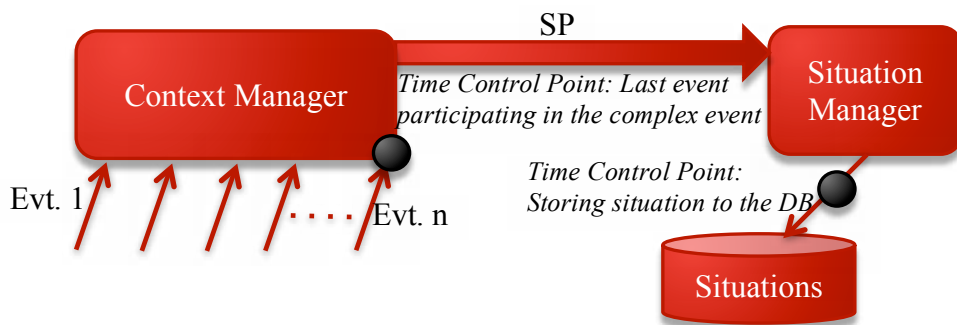


**Figure 6.91: Situation: Normal**

The previous two situations are associated with bundles. The PDP obligates the change of the current situation through the bundles BTG Start and End. However, the last situation is identified using our Esper simulating sensors. Thus, we find the last raw event from sensors participating in creating a complex event (context) that causes the starting of the doctor-needed situation.

**Table 6.16: Performance testing of the situation identification (50 Sequence)**

Situation Name	Average	Minimum	Maximum
Normal	0.330	0.275	0.656
Doctor Needed	0.024	0.016	0.100
BTG-Granted	0.313	0.266	0.446



**Figure 6.92: Performance control points to measure the identification time of situations**

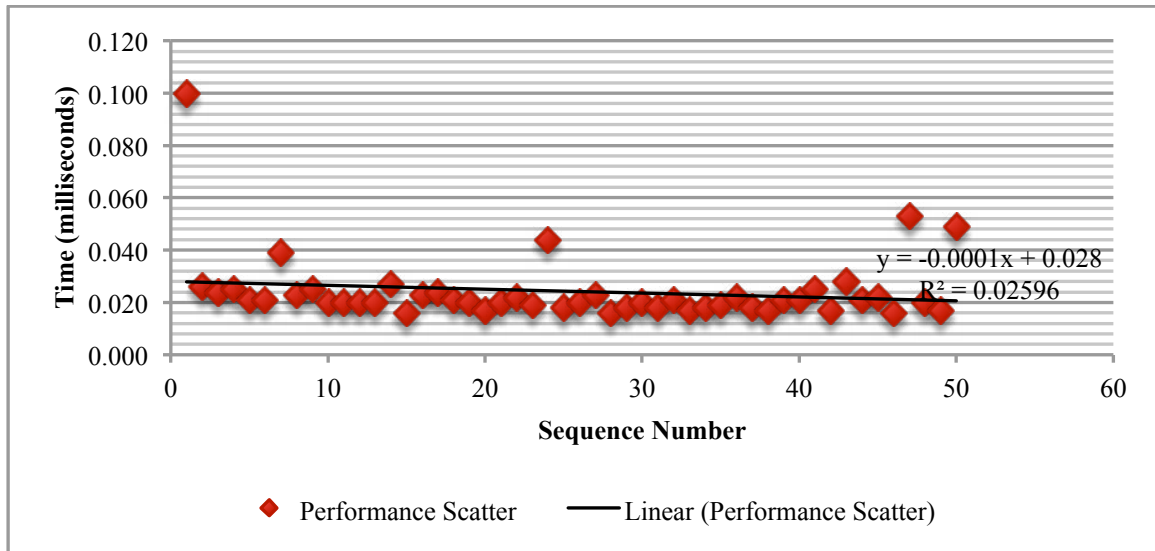


Figure 6.93: Situation: Doctor Needed

We can provide an extract of the test to demonstrate the sequential relationship of the previous figure. The following sequence diagram illustrates the interaction between different agents in order to identify the situation and calculate the time needed to be ready for the consumption by the decision-making point.

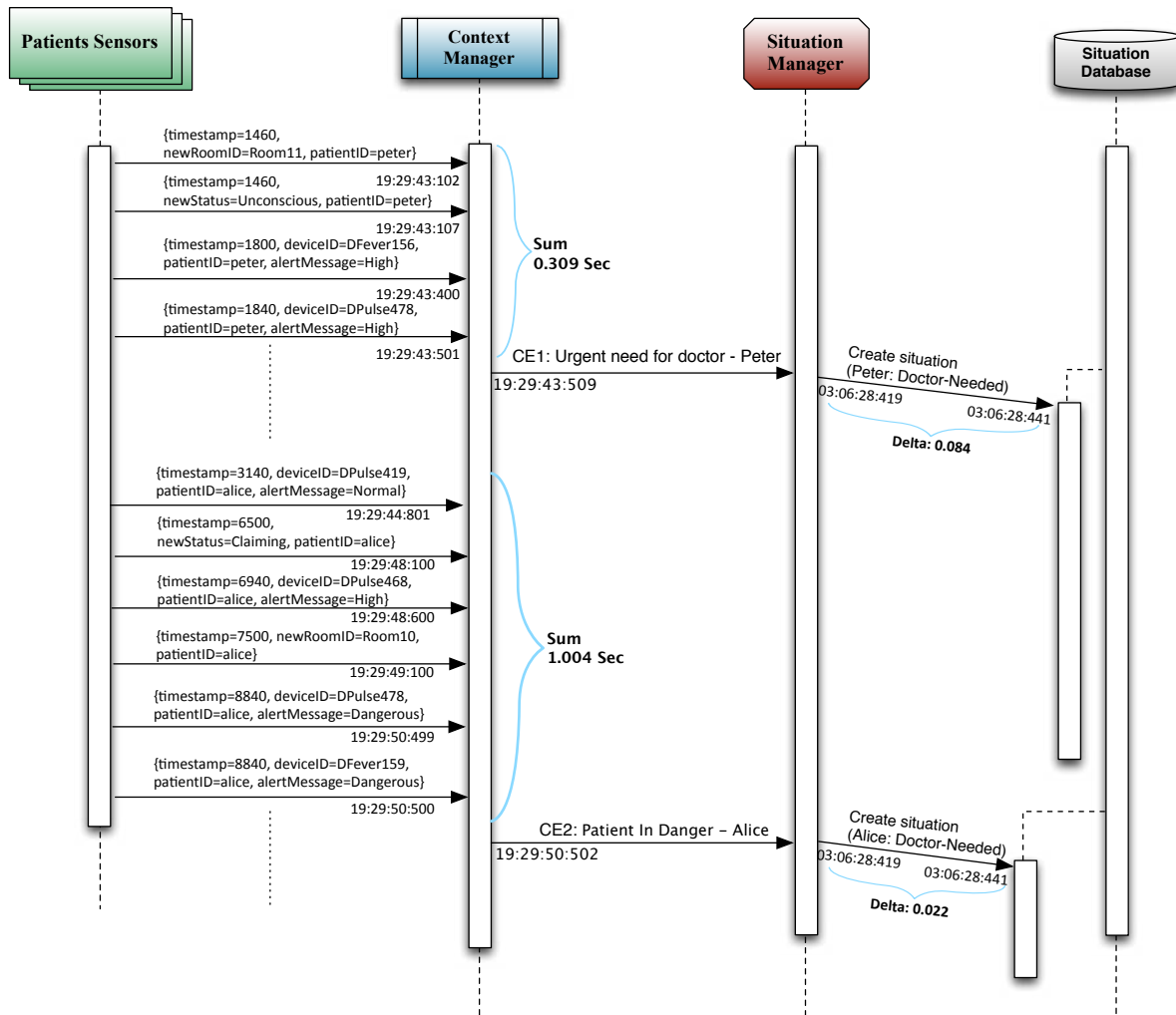


Figure 6.94: Sequence diagram showing the measurement of complex events detection and the start of new situations

In terms of the volumetry, we will mention two work efforts demonstrating the capacity and the scalability of the CEP environment we are using (Esper) for processing and handling events. During the Lion II Project in DERI-DGSIT, we tested event processing and semantics association of the third scenario during 2012/2013 in Ireland. We used a MacBook pro Core i7 2.2 Ghz and 8 Gb RAM. We installed ActiveMQ on an external server and the platform was able to manage more than 23 476 events stored inside the different topics. The events were generated from more than 20 sensors distributed and 5 controllers (raw events gathering points from these sensors).

Nevertheless, this work was unfortunately not published yet. Therefore, we make reference to published work implemented to the same project for measuring events processing time. Our listener clients (publisher/subscriber) were able to process 70 to 90 events per minute (Mehdi et al., 2013). Through the same platform that we used to perform the third scenario for exchanging complex events, the work of (Hasan & Curry, 2015) presented the ability of handling 48,000 different subscriptions.

In an internal confidential document made Nextel from Spain (one of the PREDYKOT partners), Nextel made performance testing of Esper capability on CPU: Intel Core 2 Duo T6670 2.20 Ghz, Memory: 4GB (3GB usable), Operating system Windows 7 Professional (32 bit), JVM Version: 1.6.20 with the following configuration: -Xms512m -Xmx1536m, and the Esper version is 3.4. Tests conducted launch up to 2 million events of the same type on one thread and creating two sliding windows in parallel. The treatment took 3.6 seconds. We took their permission to mention their registering of the ability of Esper to manage more than 500,000 events per second (FASYS<sup>36</sup>, "Performance Testing", Chapter 8.10, 2010).

Now, our main objective of performing the performance testing was to demonstrate the feasibility of our framework. For standalone PC with mentioned configurations, a doctor would have a semi-instantaneous<sup>37</sup> response (around half of second) for his authorization or configuration request. Feasibly, getting this result for all elements running on one machine is considered sufficiently fast (i.e., the VM configuration is shared with all the following elements: Situation Manager, Context Manager, PDP, PIP, PEPs, Master PEP, Sensors, and the ActiveMQ Bus).

We noticed a test design similar to ours that performs performance testing on the generic AAA architecture (Messoud et al., 2004). It is then possible to compare their results with ours as a way to evaluate our testing approach and outcome. At this work, there have been several test scenarios for measuring the response time. The closest one was by launching 200 requests with time results in the range between 300 milliseconds and 700 milliseconds.

Moreover, in a more complex and realistic scenario, MasterCard declared processing more than 160 million transactions per hour. Performance testing returned an average network response time of 130 milliseconds. An authorization request is measured from its processing start-time until it gets to the merchant. The authorization decision is provided in seconds (average more than one second).<sup>38</sup>

Therefore, we believe that our results are promising and interesting. However, we still need to evaluate whether these results are interesting particularly in other domain of healthcare where this scenario should be deployed. So, we first calculated the Standard Deviation (i.e., the mean of the

---

<sup>36</sup> The FASYS Spanish project is an abbreviation for FABRICA ABSOLUTAMENTE SEGURA Y SALUDABLE (ABSOLUTELY HEALTHY AND SAFE MANUFACTURING)

<sup>37</sup> "The basic advice regarding response times has been about the same for thirty years (Miller 1968; Card et al. 1991) that **0.1 second** is about the limit for having the user feel that the system is reacting instantaneously, meaning that no special feedback is necessary except to display the result."

<sup>38</sup> <http://blog.unibulmerchantservices.com/tag/transaction-authorization/>

mean) in order to study the feasibility of these calculations and the Standard Error to measure its degree of correctness.

From Table 6.17, we can interpret and conclude the performance testing results through these two concepts: standard deviation (STD) and error (STE). One can see that the more our measurements are spread apart, the more the STD and the STE are relatively large. Ideally, the more the value is small, the more the test is accurate, and the value 0 of both concepts reflects having the exact response time.

**Table 6.17: Clarifying the standard deviation and error of the measurements**

<i>Performance testing of the situation identification (50 Sequence)</i>			
Situation Name		STD Deviation	STD Error
Normal		<b>0.0723</b>	<b>0.0102</b>
Doctor Needed		0.0134	0.0019
BTG-Granted		0.0340	0.0048
<i>Performance testing of the authorizations queries (200 Sequence)</i>			
Authorization Type		STD Deviation	STD Error
Rule A: Permit Permission		<b>0.2362</b>	<b>0.0167</b>
Rule A: Deny Permission		<b>0.1532</b>	<b>0.0108</b>
<i>Performance testing of the configuration mechanism (50 Sequence)</i>			
Configuration		STD Deviation	STD Error
Name	Application Method		
Logging Access	Install	0.0273	0.0039
	Download & Install	0.0721	0.0102
BTG Start	Install	<b>0.0949</b>	<b>0.0134</b>
	Download & Install	<b>0.1365</b>	<b>0.0193</b>
BTG End	Install	0.0421	0.0060
	Download & Install	<b>0.1209</b>	<b>0.0171</b>

There are several international and national laws that have been proposed for protecting the patients information such as: Helsinki declaration (Helsinki, 1964), the act of privacy (Privacy Act, 1974), the act of HIPAA (Health Insurance Portability and Accountability) (HIPAA, 1996) and the act of patient rights and healthcare systems quality in France (Loi 2003-303, 2003). HIPAA is one of the most repudiated organizations for healthcare systems. Thus, we went searching for similar researches respecting HIPAA in order to compare the response time of their requests.

The work on optimizing the authorization policy framework (Mohan & Blough, 2010) measured the response time for authorization requests. It registered 0.5 second for 8 rules of an XACML policy handling one of the healthcare motivating scenarios. Therefore, our performance testing results is promising and not far from reality. Hence, this encourages the use of our framework to explore more scenarios with realistic data within industrial environments.

As a conclusion, the assessment of our SCD architecture examines the use case held under the PREDYKOT project. In this use case, we considered the need to provide access to health data on a patient in a critical situation. (For example, an emergency physician who by nature is not the responsible doctor needs to perform anesthesia and therefore needs to know any allergies that a patient may develop). We called such situation "Break The Glass" (BTG), where an access to data decision must be made usually by overstepping the rules.

The performance testing results presented earlier are summarized as the following:

- 1) We measured the mean time, the minimum time and maximum time elapsed for a response (i.e. positive or negative decision after a request for access to a resource). To do this we have launched 200 times the performance measurement of the same query toward an authorization server.
- 2) In this use case we also measured the time taken between an application of PEP and the implementation of a decision including an obligation. This allows us to measure the time taken by a PEP extension to download, install and use an implemented service in a bundle.
- 3) Calculating the time that elapses between an event correlation (defining the beginning of a situation) and whenever the situation is created and stored in the basic situations.

We demonstrated that these results are encouraging. Now, we are aware that these results were not obtained in the context of a real system and thus a commit in a more complex context is necessary. However, the objective of the project was to provide PREDYKOT a proof of concept without going further in the TRL levels.





# Chapter's Conclusion

We have presented in this chapter three distinctive scenarios. For each scenario, we use after the scenario specification the same methodology:

1. Modeling relevant situations
2. Representing the SCD policy
3. Identifying relevant situations
4. Applying the SCD architecture

As a result, we can argument that our framework and its methodology are with generic nature. Through this thesis manuscript, we intentionally used a trivial example to concentrate on explaining the research, the technologies, the findings, and the results of our thesis work. However, we see this chapter as a proof of concept for our “*generic nature*” claiming.

The first scenario is usually includes extreme situations that defines healthcare emergencies in a form of exceptions. Such critical situations change rapidly and interfere with the patients’ life, are difficult to manage and cause often some policy modifications. We overcome the modification of the policy with our SCD policy. Moreover, we overcome the highly dynamic of situations changing with our engineering methodology and its steps of analyzing, designing, and expressing these situations. As a result, we put the effort on the engineering step, and we obtain simpler use of our solution afterwards. The second scenario usually requires implementing a management mechanism to calculate the status of the documents and manages each update on this status. Our new implementation in this thesis reduced the complexity and integrated the need of implementing the WFE engine (i.e., the situation manager fulfill the role). At the end, we delivered the same expected results of providing dynamic authorization for collaborative work inside the virtual organizations. The third scenario is an obligation-oriented scenario. This scenario is usually implemented using energy management solution that uses obligation policy. Thus, we were able to simply implement a job submission solution that manages energy and security using obligations.

Finally, we find with all our demonstrations that the relationship between the policy, situations, contexts, and decisions is clear (see Figure 6.95). Through the three scenarios, we noticed how each time the context identifies the happening (beginning and ending) of situations, the situations give meaning to the detected context, the situations give meaning to the use of rules and group several rules, using situations orients the policy to give the corresponding decisions to the detected context, and finally the decisions change the context to stay always with conformity to the security management requirements (i.e., changes that may implies changing the situations as we saw in the BTG scenario). We evaluated this scenario at the end of this chapter with encouraging test results.

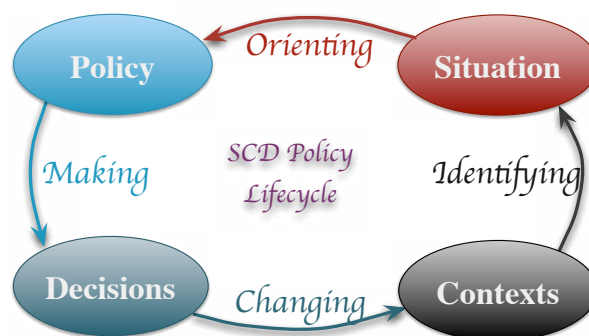


Figure 6.95: SCD Policy Lifecycle

# Chapter 7

## General Conclusion

*"I think and think for months and years. Ninety-nine times, the conclusion is false. The hundredth time I am right."*

Albert Einstein

Quote number 40559

Retrieved June 19, 2015, from [Quotes.net](http://Quotes.net)

Security management aims at ensuring that business requirements are always respected inside each element of the managed environment. Security management is conceptualized using two sides: downward (security governance task) and upward (security supervision task). We demonstrated at the beginning of this thesis the crucial need for connecting both sides together using intelligence (i.e., analyzing monitored data). Our thesis work proposed an implementation of this automatic connection through the previous chapters (Figure 7.96), in which ours is one of several implementations proposed as part of PREDYKOT project.

We have connected the upward and the downward together using the notion of *situations*. Hence, through the two earlier chapters, our objective was to demonstrate how to identify these situations, how to consider situations inside the policy (i.e., our Situation-Context-Decision policy), and how to apply and implement this situation-oriented policy.

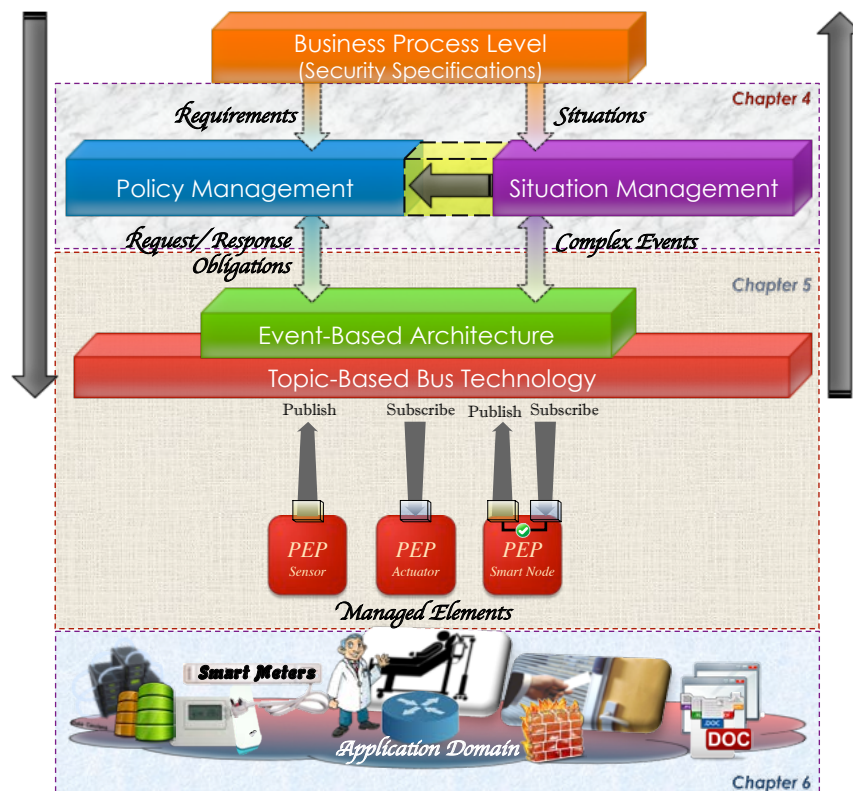


Figure 7.96: Linking our contribution chapters with our security management framework

The expression of our SCD policy requires two edition processes. The first edition process interacts with the policy-based management paradigm in aim of writing the high-level policy (the box in blue). The second edition process aims at extracting the relevant situations to model them based on

the security specifications (the box in mauve). Thus, we use the recognized situations with the created model in order to express SCD policy rules (the yellow arrow).

We used the event-driven architecture and implemented three types of agents: sensors, actuators and smart nodes. Using the three of them together, we implemented a new agent that we called Master PEP. We used the bus in order to converge different message exchanges and different agents together with respect to the interest of each agent (i.e., through using topic-based approach). Finally, we proved the generic nature of our architecture by presenting three different scenarios in the following domains: healthcare, virtual organizations, and energy management.

## I. Research Questions

At the beginning of this dissertation, we anticipated three interesting points to question about (i.e., concerning accepting the *situation* term as a feedback). As we did not answered them yet, we state at the following each question and its answer:

*RQ 4: “First point to question comes at the connection between the behavior feedback and the specified requirements. Studies hold variety of policy expression languages and models. The question is therefore how to unify all existing policy expressions in one solution that represents the behavior feedback and the heterogeneous requirements?”*

We argued in chapter 2 & 3 the existing of several languages to express the security management policy: obligation policy (through ECA rules structure) and authorization policy (through CA rules structure). We demonstrated in the same chapters that the management community is interested on monitoring the behavior changes through events in order to provide obligations (as a high-level description) and translate them to technical configurations. However, the security community is interested in observing only access requests (among the behavior changes) in order to respond by applying authorizations.

In Chapter 4, we confirmed through a simple example the complexity of using ECA or CA to express the policy that considers the behavior changes. Moreover, we confirmed that decisions should not be obligation-oriented or authorization-oriented only for the security management policy. Therefore, we proposed the Situation-Context-Decision structure to express the behavior feedback and the heterogeneous (management and security) requirements:

<b>When</b>	<b>Situation</b>
<b>And</b>	<b>Context</b>
<b>Then</b>	<b>Decision</b>

*RQ 5: “The reasoning box is the innovation point that represents the observing of behavior changes and sends them as a meaningful feedback. Therefore, the question is how to express such a meaningful feedback and how to manage the lifecycle of the policy? Does the feedback representation help bringing closer the expression of the high-level requirements with the low-level policy rules? Why reacting on raw or simple events is not enough anymore as a feedback?”*

We proposed in Chapter 4 to express the behavior feedback using the term *situation*. We presented how this term brings two features that the *event* notion fails to fulfill: semantics and abstraction. We demonstrated that the semantics side of situation is to bring meaningful feedback about the behavior and not only technical readings. The definition of these situations is extracted directly from the security requirements specification. Therefore, it is usage created a link between the SCD policy and the security requirements.

As a result, we defined our management of the policy lifecycle through a new engineering methodology. We start by analyzing the security requirements specifications to extract situations. We model these situations with their permissions and configurations. We translate this model into a policy

expressed using the SCD structure. This process should be done once at the beginning of designing the security management system. There is no need to change the policy unless there is a change request on the security requirements.

*RQ 6: “Current infrastructures are heterogeneous and incompatible with each other. Therefore, the enforcement of policy decisions once taken should be adaptably enforced to overcome such heterogeneity. So, the third point is what enforcement architecture could apply both authorization and configuration decisions into any technical environment?”*

To meet the security management definition, we needed to express using our SCD policy both the obligations with management-oriented objectives and the authorizations with security-oriented objectives. However, there is no implementation for our SCD policy that accepts both decisions possibilities: authorizations and configurations. Therefore, we implemented a unified architecture that combines both policy control models (outsourcing and provisioning) in order to permit the application of authorization and configuration decisions. Moreover, we upgraded our architecture with adaptability features in order to accept new changes faces the infrastructure technological advances.

## II. Future Work

We provide future directions for which this thesis could be considered as the foundation stone on unified and dynamic authorization based environments. We stress on our presented thesis as a work that should be continued. At the following, we present some of the future works that should imperatively be pursued on two points of view: specification and architectural.

### II.A. Security Management Specification

Obviously, *situation* is the heart of our contribution on the security management. In chapter 2 & 4, we presented the efforts carried on the notion of *context-aware* in different domains such as ubiquity and mobility. In addition to our thesis work, these efforts were concentrating on recognizing, identifying, calculating, and other complex operations in order to find the contexts and situations. What we did not sufficiently discussed in this thesis work is the trust.

We believe our identification process of contexts and situations is pertinent, but how do we put confidence that the identified events, contexts and situations are trustworthiness. Thus, there should be a future work that adds sort of metadata to our specification process of the situation-oriented policy. For instance, we recognize these metadata and then use them as the following:

- 1) In SIEM tools, OSSIM (Alien Vault) has a risk measurement process in order to calculate an indicator that helps evaluating the collected information. Thus, the risk measures help prioritizing and classifying security events and incidents using a scoring process. The risk measures are calculated using the following formula:  
$$\text{Risk} = (\text{asset} * \text{reliability} * \text{priority} / 25)$$
 (AlienVault Users Manual V 1.0, 2011)
- 2) In XACML profiles and language structure, there is a main step to verify the information we insert in side each attribute through defining an authorized source of attribute. Thus, at each collecting of contextual information, the policy recommends the specification of the issuer of attributes. If the issuer is an authorized source, then we trust the information provided. (XACML V 3.0, 22 Jan 2013)
- 3) Through a French ANR project called INCOME (Arcangeli et al., 2012), there were several efforts on defining and measuring the quality of context (QoC). To measure the *quality*, INCOME project refers to the several indications (meta-data) to provide the calculation such as up-to-dateness, accuracy, completeness, associated to context information. An interesting ongoing work to consider has generalized the QoC concept (Marie et al., 2015).

We seek in a future work to integrate the work of these three axes to identify the context that marks the SPs and the EPs of situations. Thus, we want to verify the level of trustworthiness we put in assuring the veracity of situations.

Moreover, advanced technologies are seeking to reach *prediction* in all security applications. There are many research domains interested in such feature such as multi-agent systems, machine learning, expert systems, and essentially pervasive computing (or ubiquitous computing). A future effort should be on finding whether predictions introduce an added value to our framework, keeping or logging history of previous situations is necessary, and the evolution of degraded situations.

Finally, we proposed through this thesis work *one* methodology in which it contains one step of modeling *situations* of the security specification. However, we believe that the research effort spent on extracting *situations* from the specifications is not sufficient. We need a mechanism of analyzing the specifications, recognizing relevant abstraction and semantics, and defining situations to be modeled. Thus, we need one step of specification analysis for the necessary orienting situations. The specification analysis must resolve any situations conflict. Moreover, we expect the analysis mechanism to recognize the situations hierarchies (similar to roles hierarchies). For example, the situation X has the permissions {p1, p2}. An exception (of the situation X) belongs to the same hierarchy would be therefore the delta (the difference of permissions) {p3, p4, p5}. As a result, we expect of this future work to simplify the specification of the situation-oriented policy.

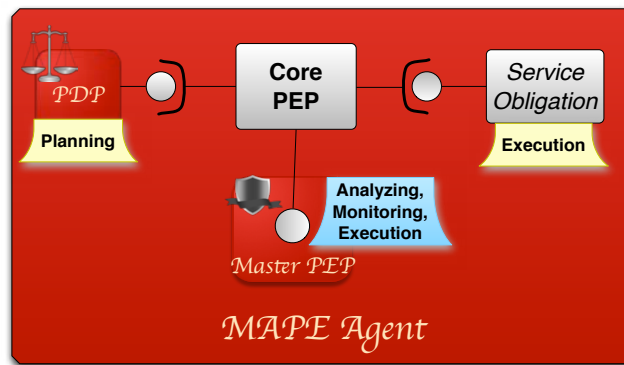
## II.B. Security Management Architecture

In our thesis work, our framework implementation is based on a management architecture solution that follows the policy-based management paradigm. We have chosen the *one-to-many* relationship (presented in Chapter 3) because it respects general policy-based management through its classical centralized architecture. We mentioned in Chapter 3 the existing of two other types: many-to-one and many-to-many. Many-to-one is when the management architecture aims at distributing policies as configurations to one PDP from several PDP sources. The many-to-many is when the management architecture aims at having communications between PDPs. However, both relationships need the implementation of many PDPs.

Therefore, our future work on the architecture side should consider implementing an environment that permits a more intelligent way of communication between not only the PDPs, but also all our SCD architecture agents. We did use an event-based architecture that is famous with the decoupling feature of its terminals. Using the Bus, we can ensure the multiple communications between agents. However, we need more intelligence of our agent in order to build discussions between each other (i.e., socialize). The functionalities that our agents are providing to ensure the security management are mainly: PDP, PIP, PEP, and Configurations.

By approximating these functionalities from the MAPE control loop, we can match each generic management-functionality with the corresponding security-specific-functionality. The MAPE functions are the following: Monitoring, Analyzing, Planning, and Execution. So, we implemented and upgraded the Master PEP with the adaptability feature (i.e., using the core PEP). Thus, our sensor functionality meets the monitoring, our actuating functionality meets the execution, and finally using our sensor or smart node functionalities we meet the analyzing.

Today, we are able to implement our agents as MAPE agents. Actually, the access requests are analyzed by the Master PEP in order to extract and construct the XACML corresponding request and then the authorization decision (also obligations if any) is analyzed in order to whom the response should be delivered. So, the only missing piece from the loop is the *planning* functionality. We interrogate the PDP through standard XACML request/response. However, through our core PEP implementation, we are able to integrate locally or remotely the PDP inside the Master PEP. Therefore, we meet the MAPE *planning* function by upgrading the Master PEP with the making-decisions feature. As a result, we obtain an agent-based architecture implemented using component-oriented services (framework presented Chapter 5). This will allow exchanges between independent MAPE agents.



**Figure 7.97: Our conception for a MAPE Agent**

Nevertheless, accepting such new types of agents would require a social environment to dialogue between each other. Thus, we are seeking to build social exchanges between agents, for example, to get specific updates (situations) or new planning (configurations). We found some studies exploring this problem within the domain of Multi-Agent Systems (MAS) (Huebscher et al., 2008; Truszkowski et al., 2009). Therefore, we believe it is interesting to bring our work closer to these studies, namely in what concerns the dialogue between agents. For example, the Master PDP can discuss with a remote PDP (through requesting specific service) in order to request configurations for updating its local SCD policy.





# Table of References

## CHAPTER 1

- [1] **Barker, S., Boella, G., Gabbay, D. M., & Genovese, V. (2009).**  
*A Meta-model of Access Control in a Fibred Security Language.* *Studia Logica*, 92(3), 437–477. <http://doi.org/10.1007/s11225-009-9203-4>  
 The issue of representing access control requirements continues to demand significant attention. The focus of researchers has traditionally been on developing particular access control models and policy specification languages for particular applications. However, this approach has resulted in an unnecessary surfeit of models and languages...
- [2] **McCollum, C. J., Messing, J. R., & Notargiacomo, L. (1990, May).**  
*Beyond the pale of MAC and DAC-defining new forms of access control.* In *Research in Security and Privacy, 1990. Proceedings., 1990 IEEE Computer Society Symposium on* (pp. 190-200). IEEE.  
 Designers of trusted systems have thus far concentrated almost exclusively on providing access controls according to the mandatory and discretionary access control paradigm, with mandatory access control based on a lattice of sensitivity labels and discretionary access...
- [3] **Kabbani, B., Laborde, R., Barrere, F., & Benzekri, A. (2013).**  
*Managing Break-The-Glass using Situation-oriented authorizations.* *Iccc 2013*, 1–13.  
 The patient's life is a redline in Healthcare environments. Whenever it comes to danger, such environments reject static authorizations. A common problem "Break The Glass" is known as the act of breaking the static authorization in order to reach the required permission. Healthcare environment is full of different contexts and situations that require the authorizations to be dynamic. Dynamic Authorization is a

concept of giving the choice to E-Health authorization system to choose the most suitable permission by considering one's situation...

- [4] **Adi, A., & Etzion, O. (2003).**  
*Amit - the situation manager.* *The VLDB Journal the International Journal on Very Large Data Bases*, 13(2), 177–203. <http://doi.org/10.1007/s00778-003-0108-y>  
 This paper presents the "situation manager", a tool that includes both a language and an efficient runtime execution mechanism aimed at reducing the complexity of active applications.
- [5] **Kabbani, B., Laborde, R., Barrere, F., & Benzekri, A. (2014).**  
*Specification and Enforcement of Dynamic Authorization Policies Oriented by Situations* (pp. 1–6). Presented at the *New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on IS - SN - VO*. <http://doi.org/10.1109/NTMS.2014.6814050>  
 Nowadays, accessing communication networks and systems faces multitude applications with large-scale requirements dimensions. Mobility –roaming services in particular– during urgent situations exacerbate the access control issues. Dynamic authorization then is required. However, traditional access control fails to ensure policies to be dynamic...
- [6] **Laborde, R., Kabbani, B., Barrere, F., & Benzekri, A. (2014).**  
*An adaptive XACMLv3 Policy Enforcement Point* (pp. 1–6). Presented at the *The 38th Annual International Computers, Software & Applications Conference*.  
 Policies are rules that govern the choices in behavior of a system. Policy based management aims at supporting dynamic adaptability of behavior by changing policy without recoding or stopping the system...

## CHAPTER 2

- [7] **Abrams, M., Albert, R., Anderson, J. P., & Arbaugh, W. (1993).**  
*Proceeding: Information System Security: User Choices* (p. 542). Presented at the *Proceedings of the 16th National Computer Security Conference (NCSC)*, Baltimore, Maryland.
- [8] **Thomas, R. K., & Sandhu, R. S. (1993).**  
*Task-based authorization: A paradigm for flexible and adaptable access control in distributed applications.* Presented at the *Proceedings of the 16th National Computer Security Conference (NCSC)*.  
 Historically, the access control problem has been couched within the framework of subjects, objects, and rights access types. An access control request thus essentially seeks an answer to a question posed typically as: Is subject *s* allowed access *a* or possess the right *a* to ...
- [9] **Harrison, M. A., Ruzzo, W. L., & Ullman, J. D. (1976).**  
*Protection in operating systems.* *Communications of the ACM*, 19(8), 461-471.  
 A model of protection mechanisms in computing systems is presented and its appropriateness is argued. The "safety" problem for protection systems under this model is to determine in a given situation whether a subject can acquire a particular right to an ...
- [10] **Bell, D. E. (2005, December).**  
*Looking Back at the Bell-La Padula Model.* *Acsac*, 337–351. <http://doi.org/10.1109/CSAC.2005.37>  
 The Bell-La Padula security model produced conceptual tools for the analysis and design of secure computer systems. Together with its sibling engineering initiatives, it identified and elucidated security principles that endure today. This paper reviews those ...
- [11] **Buss, D., & Harrison, R. (2006).**

Administrator-defined mandatory compliance expression. U.S. Patent Application 11/490,225.

- ... difficult for old-style Mandatory Access Control (MAC) and Discretionary Access Control (DAC) systems to ... The result has been the development of a new security models, such as rules-based ... Therefore, the permission contours for each actor in a system can be unique without ...
- [12] **Hu, V. C., Ferraiolo, D., & Kuhn, D. R. (2006).**  
*Assessment of access control systems.* US Department of Commerce, National Institute of Standards and Technology.  
 Abstract Adequate security of information and information systems is a fundamental management responsibility. Nearly all applications that deal with financial, privacy, safety, or defense include some form of access control. Access control is concerned with ...
- [13] **Park, J., & Sandhu, R. (2002).**  
*Originator control in usage control.* In *Policies for Distributed Systems and Networks, 2002. Proceedings. Third International Workshop on* (pp. 60-66). IEEE.  
 ... ContentGuard™ has defined it as "a language in XML for describing specifications of rights, fees and conditions for using digital contents, together with message integrity and entity authentication within these specifications ... [11] Park, Jaehong., Ravi Sandhu., and James ...
- [14] **Damianou, N., Bandara, A., Sloman, M., & Lupu, E. (2002).**  
*A survey of policy specification approaches.* Department of Computing, Imperial College of Science Technology and Medicine, London, 4, 1-37.  
 ... Non-discretionary access control (NDAC) identifies the situations in which authority is vested in some users, but there ... RDL has an ill-defined notion of delegation, whereby roles can be delegated instead of ... The notion of election is introduced to enable a client to delegate a role ...



- [15] **Cullough, D. M. (1990).**  
A hookup theorem for multilevel security. *Software Engineering, IEEE Transactions on*, 16(6), 563-568.  
In this paper, the author describes a security property for trusted multilevel systems, restrictiveness, which restricts the inferences a user can make about sensitive information. This property is a hookup property, or composable, meaning that a collection of secure ...
- [16] **Clark, D. D., & Wilson, D. R. (1987, April).**  
A comparison of commercial and military computer security policies. In *Security and Privacy, 1987 IEEE Symposium on* (pp. 184-184). IEEE.  
A Comparison of Commercial and Military Computer Security Policies David D. Clark\* - David H. Wilson\* ... identification and authorization of users, generation of audit information, and association of access control labels with all information objects are well understood. ...
- [17] **Brewer, D. F., & Nash, M. J. (1989, May).**  
The chinese wall security policy. In *Security and Privacy, 1989. Proceedings., 1989 IEEE Symposium on* (pp. 206-214). IEEE.  
The authors explore a commercial security policy (the Chinese Wall) which represents the behavior required of those persons who perform corporate analysis for financial institutions. It can be distinguished from Bell-LaPadula-like policies by the way that a user's permitted accesses are constrained by the history of his previous accesses.
- [18] **Hu, V. C., Ferraiolo, D., & Kuhn, D. R. (2006).**  
Assessment of access control systems. US Department of Commerce, National Institute of Standards and Technology.  
Access control systems come with a wide variety of features and administrative capabilities, and the operational impact can be significant. In particular, this impact can pertain to administrative and user productivity, as well as to the organization's ability to perform its mission. Therefore, it is reasonable to use a quality metric to verify the administrative capabilities, administrative cost, policy coverage, extensibility, and performance qualities of access control systems.
- [19] **Gasser, M. (1988).**  
Building a secure computer system (p. 85). New York, NY: Van Nostrand Reinhold Company.  
levels of trust identified by the Criteria range from systems that have minimal protection features to those that provide the highest level of security modern technology can produce (table 1-1) ... The worst problem with call-back modems, however, is that they may cause you to ...
- [20] **Gallagher Jr, P. R. (1988).**  
National Computer Security Center 9800 Savage Road Fort George G. Meade, MD 20755-6000 Attention: Chief, Criteria and Technical Guidelines Division.  
... A well-defined procedure or sequence of rules or steps used to produce a key stream or cipher ... This includes any action that causes unauthorized destruction, modification, or delay of service. ... the identity and need-to-know of the user, process and/or groups to which they belong ...
- [21] **Ferraiolo, D., Kuhn, D. R., & Chandramouli, R. (2003).**  
Role-based access control. Artech House.  
Role-based access control (RBAC) is a security mechanism that can greatly lower the cost and complexity of security administration for large networked applications. RBAC simplifies security administration by using roles, hierarchies, and constraints to organize privileges. ...
- [22] **Boutaba, R., & Aib, I. (2007).**  
Policy-based management: A historical perspective. *Journal of Network and Systems Management*, 15(4), 447-480.  
This paper traces the history of policy-based management and how it evolved from the first security models dating back to the late 1960's until today's more elaborate frameworks, languages, and policy-based management tools. The focus will be on ...
- [23] **Goyne, E. J., Weil, T. R., & Kuhn, R. (2011).**  
Role engineering: Methods and standards. *IT Professional*, (6), 54-57.  
Most of today's large firms use some form of role-based access control (RBAC) to support thousands of users and permission controls. Recognizing the need for some commonality among the various RBAC models, the National Institute of Standards ...
- [24] **INCITS 359-2004, Standard, R. B. A. C. (2004).**  
ANSI INCITS 359-2004. International Committee for Information Technology Standards.  
The RBAC Reference Model defines sets of basic RBAC elements (i.e., users, roles, permissions, operations and objects) and relations as types and functions that are included in this standard. The RBAC reference model serves two purposes ...
- [25] **Chadwick, D. W., Otenko, A., & Ball, E. (2003).**  
Role-based access control with X. 509 attribute certificates. *Internet Computing, IEEE*, 7(2), 62-69.  
A public key infrastructure (PKI) aims to authenticate communicating parties, but authentication alone is not enough. In addition to knowing a remote party's identity, we also need to know what actions they may perform. Thus, we need an authorization mechanism. ...
- [26] **Gavrila, S. I., & Barkley, J. F. (1998, October).**  
Formal specification for role based access control user/role and role/role relationship management. In *Proceedings of the third ACM workshop on Role-based access control* (pp. 81-90). ACM.  
Role Based Access Control (RBAC), an access control mechanism, reduces the cost of administering access control policies as well as making the process less error-prone. The Admin Tool developed for the NIST RBAC Model manages user/role and role/role
- [27] **Gouglidis, A., & Mavridis, I. (2010).**  
On the definition of access control requirements for grid and cloud computing systems. In *Networks for Grid Applications* (pp. 19-26). Springer Berlin Heidelberg.  
The emergence of grid and cloud computing systems has introduced new security concepts, so it requires new access control approaches. Traditional systems engineering processes can be enriched with helper approaches that can facilitate the definition of ...
- [28] **O'Connor, A. C., & Loomis, R. J. (2010).**  
2010 Economic Analysis of Role-Based Access Control. NIST, Gaithersburg, MD, 20899.  
This study is a retrospective economic impact analysis of role-based access control (RBAC), one of the principal approaches for managing users' access to information technology resources.
- [29] **Ni, Q., Bertino, E., Lobo, J., Brodie, C., Karat, C. M., Karat, J., & Trombeta, A. (2010).**  
Privacy-aware role-based access control. *ACM Transactions on Information and System Security (TISSEC)*, 13(3), 24.  
In this article, we introduce a comprehensive framework supporting a privacy-aware access control mechanism, that is, a mechanism tailored to enforce access control to data containing personally identifiable information and, as such, privacy sensitive. The key ...
- [30] **Ferreira, A., Chadwick, D., Farinha, P., Correia, R., Zao, G., Chiro, R., & Antunes, L. (2009, December).**  
How to securely break into RBAC: the BTG-RBAC model. In *Computer Security Applications Conference, 2009. ACSAC'09. Annual* (pp. 23-31). IEEE.  
Access control models describe frameworks that dictate how subjects (eg users) access resources. In the Role-Based Access Control (RBAC) model access to resources is based on the role the user holds within the organization. RBAC is a rigid model where ...
- [31] **Joshi, J. B., Bertino, E., Latif, U., & Ghafoor, A. (2001).**  
Trbac: A temporal role-based access control model. In *ACM Transactions on Information and System Security (TISSEC)*.  
A temporal RBAC (TRBAC) model has recently been proposed that addresses the temporal aspects of roles and trigger-based role enabling. However, it is limited to constraints on enabling of roles only. We propose a Generalized Temporal Role Based Access Control model (GTRBAC) that is capable of expressing a wider range of temporal constraints.
- [32] **Gunti, N., Sun, W., & Niamat, M. (2011, July).**

- Irbac: Isolation enabled role-based access control. In Privacy, Security and Trust (PST), 2011 Ninth Annual International Conference on (pp. 79-86). IEEE.*  
Access control is a means by which the ability to access the system is explicitly enabled or restricted in some way. Access control system enables an authority to control access to areas and resources in a given physical facility or computer based information ...
- [33] **Bouchahda, A., Le Thanh, N., Bouhoula, A., & Labbene, F. (2010, July).**  
*RBAC+: Dynamic Access Control for RBAC-administered web-based Databases. In Emerging Security Information Systems and Technologies (SECURWARE), 2010 Fourth International Conference on (pp. 135-140). IEEE.*  
Web database applications, access control issues related to data stored in the back-end databases have largely been neglected. Current approaches to access control on databases do not fit web databases because they are mostly based on individual user ...
- [34] **Masoumzadeh, A., & Joshi, J. B. (2008).**  
*PuRBAC: Purpose-aware role-based access control. In On the Move to Meaningful Internet Systems: OTM 2008 (pp. 1104-1121). Springer Berlin Heidelberg.*  
Several researches in recent years have pointed out that for the proper enforcement of privacy policies within enterprise data handling practices the privacy requirements should be captured in access control systems. In this paper, we extend the role-based access ...
- [35] **Jung, Y., & Joshi, J. B. (2012).**  
*CRIBAC: Community-centric role interaction based access control model. computers & security, 31(4), 497-523.*  
As one of the most efficient solutions to complex and large-scale problems, multi-agent cooperation has been in the limelight for the past few decades. Recently, many research projects have focused on context-aware cooperation to dynamically provide complex ...
- [36] **Zhang, Z., Zhang, X., & Sandhu, R. (2006, November).**  
*ROBAC: Scalable role and organization based access control models. In Collaborative Computing: Networking, Applications and Worksharing, 2006. CollaborateCom 2006. International Conference on (pp. 1-9). IEEE.*  
In RBAC, roles are typically created based on job functions inside an organization. Traditional RBAC does not scale up well for modeling security policies spanning multiple organizations. To solve this problem, a family of extended RBAC ...
- [37] **Thomas, R. K., & Sandhu, R. S. (1997).**  
*Task-based authorization controls (TBAC): A family of models for active and enterprise-oriented authorization management. DBSec, 113, 166-181.*  
In this paper, we develop a new paradigm for access control and authorization management, called task-based authorization controls (TBAC). TBAC models access controls from a task-oriented perspective than the traditional subject-object one. Access ...
- [38] **Laborde, R., Kamel, M., Barrère, F., & Benzekri, A. (2007).**  
*Implementation of a formal security policy refinement process in WBEM architecture. Journal of Network and Systems Management, 15(2), 241-266.*  
Security mechanisms enforcement consists in configuring devices with the aim that they cooperate and guarantee the defined security goals. In the network context, this task is complex due to the number, the nature, and the interdependencies of the devices to ...
- [39] **Khaled, A., Husain, M. F., Khan, L., Hamlen, K. W., & Thuraisingham, B. (2010, November).**  
*A token-based access control system for RDF data in the clouds. In Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on (pp. 104-111). IEEE.*  
The Semantic Web is gaining immense popularity—and with it, the Resource Description Framework (RDF) broadly used to model Semantic Web content. However, access control on RDF stores used for single machines has been seldom discussed in the ...
- [40] **Smirnov, A., Kashevnik, A., Shilov, N., & Teslya, N. (2013, June).**  
*Context-based access control model for smart space. In Cyber Conflict (CyCon), 2013 5th International Conference on (pp. 1-15). IEEE.*  
The smart space is an aggregation of devices, which can share their resources (information and services) and operate in coalitions. This nature of smart space enables of appearance of cyber conflicts between different smart space devices (or participants) ...
- [41] **Beimel, D., & Peleg, M. (2010).**  
*The context and the SitBAC models for privacy preservation—an experimental comparison of model comprehension and synthesis. Knowledge and Data Engineering, IEEE Transactions on, 22(10), 1475-1488.*  
Situation-Based Access Control (SitBAC) is a conceptual model for representing access control policies of healthcare organizations by characterizing situations of access to patient data. The SitBAC model enables formal representation of access situations as an ...
- [42] **Beimel, D., & Peleg, M. (2011).**  
*Using OWL and SWRL to represent and reason with situation-based access control policies. Data & Knowledge Engineering, 70(6), 596-615.*  
Access control is a central problem in confidentiality management, in particular in the healthcare domain, where many stakeholders require access to patients' health records. Situation-Based Access Control (SitBAC) is a conceptual model that allows for modeling ...
- [43] **dos Santos, A. L., Celestino, J., Scarlata, V., Lima, A. C., Alves, I. C., & di C Sampaio, D. (2013, December).**  
*SACM: Stateful Access Control Model a more detailed approach. In Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference on (pp. 317-324). IEEE.*  
Access control mechanisms are a fundamental building block in the construction of secure computing environments; however, most of the research in this area has been spent on traditional access control needs. These models were sufficient in classical ...
- [44] **Margaritis, G., Hatzieleftheriou, A., & Anastasiadis, S. V. (2013).**  
*Nephele: Scalable Access Control for Federated File Services. Journal of grid computing, 11(1), 83-102.*  
The integration of storage resources across different administrative domains can serve as building block for the development of efficient collaboration environments. In order to improve application portability across such environments, we target data sharing ...
- [45] **Diep, N. N., Lee, S., Lee, Y. K., & Lee, H. (2007).**  
*Contextual Risk-Based Access Control. Security and Management, 2007, 406-412.*  
Context-based access control is an emerging approach for modeling adaptive solution, making access control management more flexible and powerful. However, these strategies are inadequate for the increased flexibility and performance that ubiquitous ...
- [46] **McGraw, R. W. (2009, September).**  
*Risk-adaptable access control (radac). In Privilege (Access) Management Workshop. NIST-National Institute of Standards and Technology-Information Technology Laboratory.*  
How does the risk of maintaining the confidentiality of a given piece of information change as each properly cleared person is given access to it?—How likely is it that the confidentiality of information will be preserved after 1000 properly cleared people have been given ...
- [47] **Kalam, A. A. E., Baida, R. E., Balbiani, P., Benferhat, S., Cuppens, F., Deswarte, Y., ... & Trouessin, G. (2003, June).**  
*Organization based access control. In Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on (pp. 120-131). IEEE.*  
None of the classical access control models such as DAC, MAC, RBAC, TBAC or TMAC is fully satisfactory to model security policies that are not restricted to static permissions

- but also include contextual rules related to permissions, prohibitions, ...
- [48] **Park, J., & Sandhu, R. (2002, June).**  
Towards usage control models: beyond traditional access control. In *Proceedings of the seventh ACM symposium on Access control models and technologies* (pp. 57-64). ACM.  
In this paper we develop the concept of Usage Control (UCON) that encompasses traditional access control, trust management, and digital rights management and goes beyond them in its definition and scope. While usage control concepts have been ...
- [49] **Feltus, C. (2008, April).**  
Preliminary Literature Review of Policy Engineering Methods; Toward Responsibility Concept. In *Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on* (pp. 1-6). IEEE.  
Based on that overview's results, we are able to orient our researches more deeply by proposing an innovative approach that focuses in one hand on a policy model designed to take into account the responsibility of stakeholders and in the other hand on policy ...
- [50] **Danwei, C., Xiuli, H., & Xunyi, R. (2009).**  
Access control of cloud service based on ucon. In *Cloud computing* (pp. 559-564). Springer Berlin Heidelberg.  
Cloud computing is an emerging computing paradigm, and cloud service is also becoming increasingly relevant. Most research communities have recently embarked in the area, and research challenges in every aspect. This paper mainly discusses cloud service ...
- [51] **Lazowski, A., Martinelli, F., & Mori, P. (2010).**  
Usage control in computer security: A survey. *Computer Science Review*, 4(2), 81-99.  
Protecting access to digital resources is one of the fundamental problems recognized in computer security. As yet it remains a challenging problem to work out, starting from the design of a system until its implementation. Access control is defined as the ability to ...
- [52] **Zhang, X., Parisi-Presicce, F., Sandhu, R., & Park, J. (2005).**  
Formal model and policy specification of usage control. *ACM Transactions on Information and System Security (TISSEC)*, 8(4), 351-387.  
The recent usage control model (UCON) is a foundation for next-generation access control models with distinguishing properties of decision continuity and attribute mutability. A usage control decision is determined by combining authorizations, obligations, and ...
- [53] **Barker, S. (2009, June).**  
The next 700 access control models or a unifying meta-model?. In *Proceedings of the 14th ACM symposium on Access control models and technologies* (pp. 187-196). ACM.  
We address some fundamental questions, which were raised by Atluri and Ferraiolo at SACMAT'08, on the prospects for and benefits of a meta-model of access control. We demonstrate that a meta-model for access control can be defined and that multiple access ...
- [54] **Kuhn, D. R., Coyne, E. J., & Weil, T. R. (2010).**  
Adding attributes to role-based access control. *Computer*, (6), 79-81.  
RBAC has frequently been criticized for the difficulty of setting up an initial role structure and for inflexibility in rapidly changing domains. A pure RBAC solution may provide inadequate support for dynamic attributes such as time of day, which might need to be considered when ...
- [55] **Yang, K., & Jia, X. (2014).**  
ABAC: Attribute-based access control. In *Security for cloud storage systems* (pp. 39-58). Springer New York.  
Cloud storage service allows data owner to outsource their data to the cloud and through which provide the data access to the users. Because the cloud server and the data owner are not in the same trust domain, the semi-trusted cloud server cannot be relied to ...
- [56] **Hur, J., & Noh, D. K. (2011).**  
Attribute-based access control with efficient revocation in data outsourcing systems. *Parallel and Distributed Systems, IEEE Transactions on*, 22(7), 1214-1221.
- Some of the most challenging issues in data outsourcing scenario are the enforcement of authorization policies and the support of policy updates. Ciphertext-policy attribute-based encryption is a promising cryptographic solution to these issues for ...
- [57] **Lang, B., Foster, I., Siebenlist, F., Ananthakrishnan, R., & Freeman, T. (2009).**  
A flexible attribute based access control method for grid computing. *Journal of Grid Computing*, 7(2), 169-180.  
Grid systems have huge and changeable user groups, and different autonomous domains always have different security policies. The attribute based access control (ABAC) model, which is flexible and scalable, is more suitable for Grid systems. This paper ...
- [58] **Li, M. M., Gao, S., & Wang, J. S. (2013, September).**  
A Flexible Attribute-Based Access Control for the Dynamic Cloud. In *Applied Mechanics and Materials* (Vol. 340, pp. 671-675).  
The purpose of the cluster in order to achieve more economic expansion, the expanded use cloud computing cluster point of departure, proposed the basic framework of a cluster expansion and describe local priority strategy and cloud environments priority ...
- [59] **Golantonio, A., Di Pietro, R., & Ocello, A. (2012).**  
Role Mining in Business: Taming Role-Based Access Control Administration. *World Scientific*.  
With continuous growth in the number of information objects and the users that can access these objects, ensuring that access is compliant with company policies has become a big challenge. Role-based Access Control (RBAC) a policy-neutral access control model that ...
- [60] **Tonti, G., Bradshaw, J. M., Jeffers, R., Montanari, R., Suri, N., & Uszok, A. (2003).**  
Semantic Web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder. In *The Semantic Web-ISWC 2003* (pp. 419-437). Springer Berlin Heidelberg.  
Policies are being increasingly used for automated system management and controlling the behavior of complex systems. The use of policies allows administrators to modify system behavior without changing source code or requiring the consent or ...
- [61] **Jajodia, S., Samarati, P., Subrahmanian, V. S., & Bertino, E. (1997, June).**  
A unified framework for enforcing multiple access control policies. In *ACM Sigmod Record* (Vol. 26, No. 2, pp. 474-485). ACM.  
Although several access control policies can be devised for controlling access to information, all existing authorization models, and the corresponding enforcement...
- [62] **Hitchens, M., & Varadharajan, V. (2001).**  
Tower: A language for role based access control. In *Policies for Distributed Systems and Networks* (pp. 88-106). Springer Berlin Heidelberg.  
A language for specifying role-based access control (RBAC) policies is presented. The language is designed to support the range of access control policies of commercial object systems. The basic structures of RBAC, such as role, users and permission, are ...
- [63] **Han, W., & Lei, C. (2012).**  
A survey on policy languages in network and security management. *Computer Networks*, 56(1), 477-489.  
Policy-driven management is gaining popularity today, mainly due to the ever-growing scale and complexity of large networked systems. In these systems, policies are usually used to simplify the tasks of the system management, thus making way for further system...
- [64] **Chadwick, D. W., & Otenko, A. (2002).**  
RBAC policies in XML for X. 509 based privilege management. In *Security in the Information Society* (pp. 39-53). Springer US.  
This paper describes a role based access control policy template for use by privilege management infrastructures where the roles are stored as X. 509 Attribute Certificates in an LDAP directory. There is a brief description of the X. 509 privilege ...
- [65] **Zhou, W., & Meinel, C. (2007, September).**



- Team and task based RBAC access control model. In *Network Operations and Management Symposium, 2007. LANOMS 2007. Latin American* (pp. 84-94). IEEE.
- In this paper, we introduce a new paradigm for to model contextual information associated with organizational access control and authorization management, called team and roles and responsibilities and collaborative activity. task based RBAC (TT-RBAC). TT-...
- [66] López, G., Cánovas, Ó., Gómez-Skarmeta, A. F., Otenko, S., & Chadwick, D. W. (2005). *A heterogeneous network access service based on PERMIS and SAML. In Public Key Infrastructure* (pp. 55-72). Springer Berlin Heidelberg.
- The expansion of inter-organizational scenarios based on different authorization schemes involves the development of integration solutions allowing different authorization domains to share, in some way, protected resources. This paper analyzes different ...
- [67] Cánovas, Ó., López, G., & Gómez-Skarmeta, A. F. (2004). *A credential conversion service for SAML-based scenarios. In Public key infrastructure* (pp. 297-305). Springer Berlin Heidelberg.
- Coordination of different administrative domains involves several security concerns, especially from an authorization point of view. SAML is getting a lot of popularity as a language that can be used to bridge several isolated authorization systems in order to ...
- [68] Bajaj, S., Box, D., Chappell, D., Curbera, F., Daniels, G., Hallam-Baker, P., ... & Orchard, D. (2006). *Web Services policy 1.2-framework (WS-policy). W3C Member Submission, 25, 12.*
- This WS-Policy Specification is a public draft release and is provided for review and evaluation only. The Authors hope to solicit your contributions and suggestions in the near future.
- [69] Della-Libera, G., Gudgin, M., Hallam-Baker, P., Hondo, M., Granqvist, H., Kaler, G., ... & Zolfonoon, R. (2005). *Web services security policy language. IBM and Microsoft and RSA Security and VeriSign, 25-32.*
- [70] Simon, B., Goldschmidt, B., Budai, P., Hartung, I., Kondorosi, K., Laszlo, Z., & Risztics, P. (2011). *A metamodel of the ws-policy standard family. In The Fifth International Conference on Digital Society, ICDS* (pp. 57-62).
- The basic building blocks of SOA systems are web services. The domain specific language SOAL developed by the authors has a Java and C#-like syntax for describing web service interfaces. Beside the syntax a metamodel (SoaMM) is also defined. The paper ...
- [71] Speiser, S. (2010, July). *Semantic annotations for ws-policy. In Web Services (ICWS), 2010 IEEE International Conference on* (pp. 449-456). IEEE.
- WS-Policy is a standard to express requirements and capabilities in Web service systems. Policies are based on domain-specific assertions. In this paper we present a lightweight approach to semantic annotations of policy assertions. The approach allows ...
- [72] Moui, A. (2013). *Un cadre de spécification et de déploiement de politiques d'autorisation (Doctoral dissertation, Université de Toulouse, Université Toulouse III-Paul Sabatier).*
- L'objectif des travaux présentés dans ce mémoire est de proposer une solution logicielle (cadriciel) apte à supporter l'automatisation de l'adaptation d'une activité de surveillance. Une telle adaptation résulte en une modification de la manière de surveiller. Ainsi, il est ...
- [73] Akue, L., Lavinal, E., Desprats, T., & Sibilla, M. (2013, October). *Integrating an online configuration checker with existing management systems: Application to CIM/WBEM environments. In Network and Service Management (CNSM), 2013 9th International Conference on* (pp. 339-344). IEEE.
- Runtime configuration validation is a critical requirement if we are to build reliable self-adaptive management systems. This paper presents a generic framework that includes a runtime configuration checker built upon a high-level language dedicated to the ...
- [74] Desertot, M., Escoffier, G., & Donsez, D. (2007). *Towards an autonomic approach for edge computing. Concurrency and Computation: Practice and Experience, 19(14), 1901-1916.*
- Nowadays, one of the biggest challenges for companies is to cope with the high cost of their information technologies infrastructure. Edge computing is a new computing paradigm designed to allocate on-demand computing and storage resources. Those ...
- [75] Cheaito, M. (2012). *Un cadre de spécification et de déploiement de politiques d'autorisation (Doctoral dissertation, Université de Toulouse, Université Toulouse III-Paul Sabatier).*
- Notre travail propose une méthodologie de conception et de développement d'un système d'autorisation adaptable aux différentes facettes que peut recouvrir le contrôle d'accès dans les organisations telles que l'hétérogénéité des pratiques organisationnelles, des ...
- [76] RFC 1155: Rose, M., & McGlohrrie, K. (1990). *RFC 1155-Structure and Identification of Management Information for TCP. IP-based Internets.*
- [77] Case, J., McGlohrrie, K., Rose, M., & Waldbusser, S. (1999). *RFC 2578 SMIV2 April 1999. Structure.*
- [78] RFC 3216: Jason, J., Schoenwaelder, J., Strauss, F., & Weiss, W. (2001). *RFC 3216 SMIng Objectives December 2001.*
- [79] Shafer, P. (2011). *An Architecture for Network Management Using NETCONF and YANG.*
- The Network Configuration Protocol (NETCONF) gives access to native capabilities of the devices within a network, defining methods for manipulating configuration databases, retrieving operational data, and invoking specific operations. YANG provides the means to ...
- [80] Schönwälder, J., Björklund, M., & Shafer, P. (2010). *Network configuration management using NETCONF and YANG. IEEE communications magazine, 48(9), 166-173.*
- The Internet Engineering Task Force has standardized a new network configuration management protocol called NETCONF, which provides mechanisms to install, manipulate, and delete the configuration of network devices. This article describes ...
- [81] Knertser, D., & Tsarinenko, V. (2013). *Network Device Discovery.*
- Modern heterogeneous networks present a great challenge for network operators and engineers from a management and configuration perspective. The Tail-f Systems' Network Control System (NCS) is a network management framework that addresses these ...
- [82] Boutaba, R., & Aib, I. (2007). *Policy-based management: A historical perspective. Journal of Network and Systems Management, 15(4), 447-480.*
- This paper traces the history of policy-based management and how it evolved from the first security models dating back to the late 1960's until today's more elaborate frameworks, languages, and policy-based management tools. The focus will be on ...
- [83] DSP0004, 3.0.0 (2012). *DMTF Standard - Common Information Model (CIM) Metamodel.*
- [84] Festor, O., Festor, P., Youssef, N. B., & Andrey, L. (1999). *Integration of WBEM-based Management Agents in the OSI Framework. In Integrated Network Management, 1999. Distributed Management for the Networked Millennium. Proceedings of the Sixth IFIP/IEEE International Symposium on* (pp. 49-64). IEEE.
- In this paper, we propose a set of mappings and an implementation of an integration agent allowing WBEM-based agents implementing a CIM information model to be

- managed by OSI-based management platforms and applications. Extending existing ...
- [85] **Nataf, E., Festor, O., & Doyen, G. (2003).**  
*An SMIng-centric proxy agent for integrated monitoring and provisioning. In Integrated Network Management VIII (pp. 491-503). Springer US.*  
 The combined use of SNMP and policy based frameworks is growing very fast. From both an information model and programming interface point of view, an integrated view is highly desirable. Several approaches have been proposed so far in the ...
- [86] **de Vergara, J. E. L., Villagrà, V. A., & Berrocal, J. (2005).**  
*On the formalization of the common information model metaschema. In Ambient Networks (pp. 1-11). Springer Berlin Heidelberg.*  
 Integrated network management frameworks include a common definition of the managed resources, known as an information model, which is a key factor to describe the domain to be managed. In this scope, it is important to understand the semantics each ...
- [87] **Agrawal, D., Lee, K. W., & Lobo, J. (2005).**  
*Policy-based management of networked computing systems. Communications Magazine, IEEE, 43(10), 69-75.*  
 This article provides an overview of the Policy Management for Autonomic Computing (PMAC) platform, and shows how it can be used for the management of networked systems. We present the policy information model adopted by PMAC ...
- [88] **Agrawal, D., Calo, S., Lee, K. W., & Lobo, J. (2007, May).**  
*Issues in designing a policy language for distributed management of it infrastructures. In Integrated Network Management, 2007. IM'07. 10th IFIP/IEEE International Symposium on (pp. 30-39). IEEE.*  
 The objectives of this paper are twofold. First, we introduce a novel policy language, called CIM-SPL (Simple Policy Language for CIM) that complies with the CIM (Common Information Model) Policy Model and fully incorporates CIM constructs. ...
- [89] **RFC 3060: Moore, B., Ellesson, E., Strassner, J., & Westerinen, A. (2001).**  
*Policy Core Information Model-Version 1 Specification (No. RFC 3060).*  
 This document presents the object-oriented information model for representing policy information developed jointly in the IETF Policy Framework WG and as extensions to the Common Information Model (CIM) activity in the Distributed Management Task Force ...
- [90] **Lymberopoulos, L. A. (2004).**  
*An adaptive policy based framework for network management (Doctoral dissertation, Imperial College London London).*  
 Policy-based management has emerged as a promising solution for the management of large-scale and heterogeneous networks. This approach has been adopted in several network management areas, such as in the areas of Quality of Service (QoS) ...
- [91] **DMTF Standard DSP107 (2003).**  
*CIM event model white paper, Preliminary CIM V 2.7*
- [92] **DMTF Standard DSP0231 (2009).**  
*CIM Simplified Policy Language (CIM-SPL)*
- [93] **RFC 3198: Westerinen, A., Schnizlein, J., Strassner, J., Scherling, M., Quinn, B., Herzog, S., ... & Waldbusser, S. (2001).**  
*Terminology for policy-based management (No. RFC 3198).*  
 This document is a glossary of policy-related terms. It provides abbreviations, explanations, and recommendations for use of these terms. The intent is to improve the comprehensibility and consistency of writing that deals with network policy, particularly ...
- [94] **Chan, K., Durham, D., Gai, S., Herzog, S., McCloghrie, K., Reichmeyer, F., ... & Yavatkar, R. (2001).**  
*COPS usage for policy provisioning. IETF Draft draft-ietf-rapp-02.txt, March 2000.*
- [95] **Lobo, J., Bhatia, R., & Naqvi, S. (1999).**  
*A policy description language. In Proceedings of AAAI (pp. 291-298).*  
 A policy describes principles or strategies for a plan of action designed to achieve a particular set of goals. We define a policy as a function that maps a series of events into a set of actions. In this paper we introduce P~ D $\bar{E}$ , a simple but expressive language to specify ...
- [96] **Strassner, J. (2003).**  
*Policy-based network management: solutions for the next generation. Morgan Kaufmann.*  
 Policy-Based Network Management (PBNM) systems enable business rules and procedures to be translated into policies that configure and control the network and its services. Those who manage network systems are aware that this approach can benefit both network ...
- [97] **Kohli, M., & Lobo, J. (1999).**  
*Policy based management of telecommunication networks. In Policy Workshop 1999.*  
 Network management can be split into five probably interrelated branches: fault management, configuration management, performance management, security and accounting. Many network management activities are carried out by enforcing policies ...
- [98] **Chomicki, J., Lobo, J., & Naqvi, S. (2000).**  
*A Logic Programming Approach to Conflict Resolution in Policy Management. Syntax.*  
 The simple event-condition-action (ECA) rule paradigm of active databases has proved very useful in many AI and database applications. However, its applicability goes beyond data management. ECA rules can be used in network management and ...
- [99] **Nicklisch, J. (1999).**  
*A rule language for network policies. Policy.*  
 This position paper outlines the design and implementation of a multi-purpose policy definition language, referred to as PFDL. The language is used in a prototypical policy server to implement the centralized view of a network administrator with regard to signaled ...
- [100] **Damianou, N., Dulay, N., Lupu, E., & Sloman, M. (2001).**  
*The ponder policy specification language. In Policies for Distributed Systems and Networks (pp. 18-38). Springer Berlin Heidelberg.*  
 The Ponder language provides a common means of specifying security policies that map onto various access control implementation mechanisms for firewalls, operating systems, databases and Java. It supports obligation policies that are event triggered ...
- [101] **Polyrakis, A., & Boutaba, R. (2002).**  
*The meta-policy information base. Network, IEEE, 16(2), 40-48.*  
 The recent considerable growth of computer networks has revealed significant scalability and efficiency limitations to the traditional management techniques. Policy-based networking (PBN) has emerged as a promising paradigm for configuration management ...
- [102] **Jin, X., Krishnan, R., & Sandhu, R. S. (2012).**  
*A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC. DBSec, 12, 41-55.*  
 Recently, there has been considerable interest in attribute based access control (ABAC) to overcome the limitations of the dominant access control models (ie, discretionary-DAC, mandatory-MAC and role based-RBAC) while unifying their advantages. Although ...
- [103] **Damianou, N., Dulay, N., Lupu, E., & Sloman, M. (2001).**  
*The ponder policy specification language. In Policies for Distributed Systems and Networks (pp. 18-38). Springer Berlin Heidelberg.*  
 The Ponder language provides a common means of specifying security policies that map onto various access control implementation mechanisms for firewalls, operating systems, databases and Java. It supports obligation policies that are event triggered ...
- [104] **Zhou, G., Wang, D. W., Li, F., Zhang, L., Li, N., Wu, Z. S., ... & Cheng, H. M. (2010).**

- Graphene-wrapped Fe<sub>3</sub>O<sub>4</sub> anode material with improved reversible capacity and cyclic stability for lithium ion batteries. *Chemistry of Materials*, 22(18), 5306-5313.
- A well-organized flexible interleaved composite of Fe<sub>3</sub>O<sub>4</sub> particles enwrapped by graphene nanosheets (GNSs) was prepared. The GNSs play a "flexible confinement" function for tolerating the volume change of Fe<sub>3</sub>O<sub>4</sub> particles, whereas the Fe<sub>3</sub>O<sub>4</sub> particles inhibit the ...
- [105] **Chadha, R., & Kant, L. (2008).**  
*Policy-driven mobile ad hoc network management (Vol. 102).* John Wiley & Sons.  
 This book should be immensely interesting to those trying to decide what MANET research is worth undertaking and why."-J. Christopher Ramming, Program Manager, Defense Advanced Research Projects Agency (DARPA) Strategic Technology Office A thorough, ...
- [106] **RFC 3512: MacFaden, M., Partain, D., Saperia, J., & Tackabury, W. F. (2003).**  
 Configuring networks and devices with Simple Network Management Protocol (SNMP), RFC 3512. *IETF Request for Comment*, [Online], 1-69.
- [107] **Mauro, D. R., & Schmidt, K. J. (2001).**  
*O'Reilly Essential SNMP.* July.  
 Simple Network Management Protocol (SNMP) provides a "simple" set of operations that allows you to more easily monitor and manage network devices like routers, switches, servers, printers, and more...
- [108] **Zhang, L., Berson, S., Herzog, S., & Jamin, S. (1997).**  
 Resource ReSerVation protocol (RSVP)-version 1 functional specification. *Resource*.
- [109] **Blake, S., Bernet, Y., Binder, J., Carlson, M., Keshav, S., Davies, E., ... & Weiss, W. (1998).**  
 A framework for differentiated services. *draft-ietf-diffserv-framework-01.txt*.
- [110] **Ferrari, T., & Chimento, P. F. (2000).**  
 A measurement-based analysis of expedited forwarding PHB mechanisms. In *Quality of Service, 2000. IWQOS. 2000 Eighth International Workshop on (pp. 127-137).* IEEE.  
 The differentiated services (diffserv) architecture defines a new framework for the support of quality of service (QoS) in IP-based networks. In this paper, we focus on the mechanisms used to implement the Expedited Forwarding (EF) Per-Hop Behavior (PHB)....
- [111] **RFC 2598: Jacobson, V., Nichols, K., & Poduri, K. (1999).**  
 An expedited forwarding PHB (No. RFC 2598).  
 The definition of PHBs (per-hop forwarding behaviors) is a critical part of the work of the DiffServ Working Group. This document describes a PHB called Expedited Forwarding. We show the generality of this PHB by noting that it can be produced by more than one mechanism and...
- [112] **Snir, Y., Strassner, J., Ramberg, Y., Moore, B., & Cohen, R. (2003).**  
 Policy quality of service (QoS) information model. *Policy*.
- [113] **Seitz, L., & Rissanen, E. (2007).**  
 NETCONF access control profile for XACML.  
 The NETCONF remote network configuration protocol currently lacks an access control model. The need for such a model has been recognized within the NETCONF working group. The eXtended Access Control Markup Language (XACML) is an XML-based ...
- [114] **RFC 6241: Enns, R., Björklund, M., Schoenwaelder, J., & Bierman, A. (2011).**  
 Network configuration protocol (NETCONF) (No. RFC 6241).  
 The Network Configuration Protocol (NETCONF) defined in this document provides mechanisms to install, manipulate, and delete the configuration of network devices. It uses an Extensible Markup Language (XML)-based data encoding for the configuration data as...
- [115] **Schönwälder, J., Björklund, M., & Shafer, P. (2010).**  
 Network configuration management using NETCONF and YANG. *IEEE communications magazine*, 48(9), 166-173.  
 The Internet Engineering Task Force has standardized a new network configuration management protocol called NETCONF, which provides mechanisms to install, manipulate, and delete the configuration of network devices. This article describes ...
- [116] **Sehgal, A., Perelman, V., Kuryla, S., & Schönwälder, J. (2012).**  
 Management of resource constrained devices in the internet of things. *Communications Magazine, IEEE*, 50(12), 144-149.  
 The embedded computing devices deployed within the Internet of Things are expected to be resource constrained. This resource constraint not only applies to memory and processing capabilities, but the low-power radio standards utilized further constrain ...
- user SLAs (Service Level Agreements). But, network applications have different characteristics and thus have not the same QoS requirements. Yesterday ...
- [121] **RFC 2753: Yavatkar, R., Pendarakis, D., & Guerin, R. (2000).**  
 IETF RFC 2753: A framework for policy based admission control. *Network Working Group. Request for Comments: 2753 Intel Category: Informational D. Pendarakis IBM*.
- [122] **ISO/IEC 10040. INTERNATIONAL STANDARD / Second edition. (1998)**  
 Information technology — Open Systems Interconnection — Systems management overview
- [123] **ISO/IEC 7498-4 (1989)**  
 Information processing systems - Open Systems Interconnection - Basic Reference Model - Part 4: Management framework
- [124] **ITU-T M3010 (2000).**  
 Principles for a telecommunications management network. *TELECOMMUNICATION STANDARDIZATION SECTOR OF ITU*  
 SERIES M: TMN AND NETWORK MAINTENANCE: INTERNATIONAL TRANSMISSION SYSTEMS, TELEPHONE CIRCUITS, TELEGRAPHY, FACSIMILE AND LEASED CIRCUITS
- [125] **Jude, M. (2001).**  
 Policy-Based Management: beyond the hype. *Business Communication Review*, 52-56.  
 PBNM At A Glance PBNM functions start with bandwidth management, ie, prioritizing the applications or transactions

## CHAPTER 3

- [117] **Clemm, A. (2007).**  
*Network management fundamentals (Vol. 800).* Cisco Press.  
 Network Management Fundamentals A guide to understanding how network management technology really works Alexander Clemm, Ph. D. Network management is an essential factor in successfully operating a network. As a company becomes increasingly...
- [118] **ISO/IEC 10040: 1998. | Recommendation, I. T. U. T. X. 701 (1997)**  
 Information technology—Open Systems Interconnection—Systems management overview.
- [119] **Harrington, D., Wijnen, B., & Presuhn, R. (2002).**  
 An architecture for describing simple network management protocol (SNMP) management frameworks.  
 This document describes an architecture for describing Simple Network Management Protocol (SNMP) Management Frameworks. The architecture is designed to be modular to allow the evolution of the SNMP protocol standards over time. The major...
- [120] **Bennani, F., Boutignon, A., & Simoni, N. (2001, July).**  
 IP QoS issues: efficient cooperation of DiffServ, MPLS, and management. In *ITCom 2001: International Symposium on the Convergence of IT and Communications (pp. 1-11).* International Society for Optics and Photonics.  
 The end of any network architecture is to provide and to guarantee the QoS (Quality Of Service), in order to meet with



- that must share network resources. This can be as simple as the bandwidth of a particular transmission path or as complex as the bandwidth of a server. In ...
- [126] Mohaban, S., Parnafes, I., Ramberg, Y., Snir, Y., & Strassner, J. (2004).  
*Method and apparatus for storing policies for policy-based management of network quality of service*. U.S. Patent No. 6,718,380. Washington, DC: U.S. Patent and Trademark Office.  
*A method and apparatus for storing policies for use in policy-based management of quality of service treatments of network data traffic flows are described. The policies are stored in the form of policy statements. Each policy statement applies to a specific application that ...*
- [127] Strassner, J. (2004, April).  
*Autonomic networking-theory and practice*. In *Network Operations and Management Symposium, 2004. NOMS 2004. IEEE/IFIP (Vol. 1, pp. 927-Vol). IEEE*.  
*A new genre of management applications is required to accommodate current and future uses of network services. The key to solving this problem is to realize that currently, network operation is divorced from how the business operates, and that current approaches don't ...*
- [128] Follows, J., & Straeten, D. (1999).  
*Application driven networking: Concepts and architecture for policy-based systems*. IBM Corporation.  
*This book describes the standards and IBM's implementation of application-driven networking, a term for an architecture and implementation in which network services are driven by rules, global policies and by the applications themselves. This architecture turns the old concept around; applications used to have to conform to the limitations of the network, whereas now networks can be made to conform to the requirements of the applications...*
- [129] Sloman, M., & Lupu, E. (2002).  
*Security and management policy specification*. *Network, IEEE*, 16(2), 10–19. <http://doi.org/10.1109/65.993218>  
*Policies are rules governing the choices in behavior of a system. They are increasingly being used as a means of implementing flexible and adaptive systems for management of Internet services, networks, and security systems. There is also a need for a common specification of security policy for large-scale multi-organizational systems where access control is implemented in a variety of heterogeneous components.*
- [130] RFC 2748: Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R., & Sastry, A. (2000).  
*RFC 2748. The COPS (Common Open Policy Service) Protocol*.
- [131] Barrère, F., Benzekri, A., Grasset, F., Laborde, R., & Raynaud, Y. (2001).  
*Distribution de politiques de sécurité IPsec*. Marrakech, Morocco, December.  
*Le déploiement du réseau Internet et la richesse des services proposés permet de positionner aujourd'hui ce réseau comme infrastructure d'interconnexion complémentaire de celles traditionnellement utilisées. Le standard IPsec de l'IETF présente les ...*
- [132] Rensing C., Karsten, H. M., & Stiller, B. (2001).  
*A Survey on AAA Mechanisms, Protocols, and Architectures and a Policy-based Approach beyond: A\**  
*AAA, the Authentication, Authorization, and Accounting approach for dial-up connectivity of mobile users and devices has reached a status of maturity, however, limited to a dedicated set of minor scenarios. While the commercialization of the Internet has lead ...*
- [133] RFC 2865: Willens, S., Rubens, A. C., Rigney, G., & Simpson, W. A. (2000).  
*Remote authentication dial in user service (RADIUS)*.  
*This document describes a protocol for carrying authentication, authorization, and configuration information between a Network Access Server which desires to authenticate its links and a shared Authentication Server.[STANDARDS-TRACK]*
- [134] RFC 1492: Finseth, C. (1993).  
*An access control protocol, sometimes called TACACS*.  
*This RFC documents the extended TACACS protocol use by the Cisco Systems terminal servers. This same protocol is used by the University of Minnesota's distributed authentication system. This memo provides information for the Internet community. It does ...*
- [135] RFC 3588: Calhoun, P., Loughney, J., Guttman, E., Zorn, G., & Arkko, J. (2003).  
*J. Arkko," Diameter Base Protocol. RFC 3588, September. Network Working Group.*  
*The Diameter base protocol is intended to provide an Authentication, Authorization and Accounting (AAA) framework for applications such as network access or IP mobility Diameter is also intended to work in both local Authentication, Authorization & Accounting ...*
- [136] RFC 2749: Cohen, R., Durham, D., Rajan, R., & Sastry, A. (2000).  
*Network Working Group S. Herzog, Ed. Request for Comments: 2749 IPHighway Category: Standards Track J. Boyle Level3.*  
*The Common Open Policy Service (COPS) protocol is a query response protocol used to exchange policy information between a network policy server and a set of clients [COPS]. COPS is being developed within the RSVP Admission Policy Working Group (RAP WG) of ...*
- [137] RFC 3084: Gai, S., McCloghrie, K., Herzog, S., Reichmeyer, F., Yavatkar, R., & Smith, A. (2001).  
*Network Working Group K. Chan Request for Comments: 3084 J. Seligson Category: Standards Track Nortel Networks D. Durham Intel.*  
*This document describes the use of the Common Open Policy Service (COPS) protocol for support of policy provisioning (COPS-PR). This specification is independent of the type of policy being provisioned (QoS, Security, etc.) but focuses on the mechanisms ...*
- [138] Vollbrecht, J. R., Holdrege, M., de Laat, C. T., Calhoun, P. R., Gommans, L., Farrell, S., ... & Gross, G. M. (2000).  
*AAA authorization framework*.  
*This memo serves as the base requirements for Authorization of Internet Resources and Services (AIRS). It presents an architectural framework for understanding the authorization of Internet resources and services and derives requirements for ...*
- [139] Jacquenet, C., Bourdon, G., & Boucadair, M. (2008).  
*Service automation and dynamic provisioning techniques in IP/MPLS environments (Vol. 16)*. John Wiley & Sons.  
*Chapter 11: Dynamic Enforcement of Security Policies in IP/MPLS Environments... Keywords: IP/MPLS network and AAA chain; web-based access control; user authentication and authorization phase; RADIUS messages; Access-Accept message in AAA server; ...*
- [140] Hassell, J. (2002).  
*RADIUS: securing public access to private resources. "O'Reilly Media, Inc."*  
*The subject of security never strays far from the minds of IT workers, for good reason. If there is a network with even just one connection to another network, it needs to be secured. RADIUS, or Remote Authentication Dial-In User Service, is a widely deployed protocol ...*
- [141] RFC 2906: Farrell, S., de Laat, C. T., Calhoun, P. R., & Gross, G. M. (2000).  
*AAA Authorization Requirements*.  
*This document specifies the requirements that Authentication Authorization Accounting (AAA) protocols must meet in order to support authorization services in the Internet. The requirements have been elicited from a study of a range of applications...*
- [142] RFC 2903: De Laat, C. T. A. M., Gross, G., Gommans, L., Vollbrecht, J., & Spence, D. (2000).  
*Generic AAA architecture (No. RFC 2903)*.
- [143] Godik, S., Moses, T., Anderson, A., Parducci, B., Adams, C., Flinn, D., ... & Moses, T. (2003).  
*extensible access control markup language (xacml) version 1.0*.
- [144] Laborde, R., Gheaito, M., Barrère, F., & Benzekri, A. (2009, November).  
*An extensible XACML authorization web service: Application to dynamic web sites access control. In Signal-Image Technology & Internet-Based Systems*



- (SITIS), 2009 Fifth International Conference on (pp. 499-505). IEEE.
- Attribute Based Access Control can define permissions based on just about any security relevant characteristics of requestors, actions, resources, and environment, known as attributes. XACML is an access control OASIS standard compliant to this approach. ...
- [145] Cheaito, M., Laborde, R., Barrère, F., & Benzekri, A. (2010, October).  
A deployment framework for self-contained policies. In *Network and Service Management (CNSM), 2010 International Conference on* (pp. 88-95). IEEE.  
One of the key motivations of policy-based management is flexibility and adaptability to existing infrastructure and change management. In the context of security, modern policy languages such as XACML are extensible and support natively the ...
- [146] Laborde, R., Kamel, M., Barrère, F., & Benzekri, A. (2008, June).  
Pep= point to enhance particularly. In *Policies for Distributed Systems and Networks, 2008. POLICY 2008. IEEE Workshop on* (pp. 93-96). IEEE.  
Policies are rules that govern the choices in behaviour of a system. Policy based management aims at supporting dynamic adaptability of behaviour by changing policy without recoding or stopping the system. The common accepted architecture of such...
- [147] Bosch, J. (2009, August).  
From software product lines to software ecosystems. In *Proceedings of the 13th International Software Product Line Conference* (pp. 111-119). Carnegie Mellon University.
- Software product line companies increasingly expand their platform outside their organizational boundaries, in effect transitioning to a software ecosystem approach. In this paper, we discuss the emerging trend of software ecosystems and provide a overview of ...
- [148] van den Berk, I., Jansen, S., & Luinenburg, L. (2010, August).  
Software ecosystems: a software ecosystem strategy assessment model. In *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume* (pp. 127-134). ACM.  
Software companies and organizations increasingly open up their business to other software companies and as a consequence they find themselves in an ecosystem of software companies, developers and partners. These actors, and especially the software...
- [149] Samarji, L., Cuppens, F., Cuppens-Boulahia, N., Kanoun, W., & Dubus, S. (2013).  
Situation calculus and graph based defensive modeling of simultaneous attacks. In *Cyberspace Safety and Security* (pp. 132-150). Springer International Publishing.  
Recent attacks are better coordinated, difficult to discover, and inflict severe damages to networks. However, existing response systems handle the case of a single ongoing attack. This limitation is due to the lack of an appropriate model that describes...
- [150] Hu, B., Patkos, T., Chibani, A., & Amirat, Y. (2012).  
Rule-based context assessment in smart cities (pp. 221-224). Springer Berlin Heidelberg.  
This paper presents a rule-based architecture that enables causal and temporal reasoning of events and supports their relevance assessment given the user's situation, in order to provide contextualized services for citizens inhabiting a smart city. Our approach...
- ## Chapter 4
- [151] Tsai, W. T., Liu, X., Chen, Y., & Paul, R. (2005, April).  
Simulation verification and validation by dynamic policy enforcement. In *Simulation Symposium, 2005. Proceedings. 38th Annual* (pp. 91-98). IEEE.  
This paper presents a new verification and validation (V&V) technique for simulation using dynamic policy enforcement. Constraints are formally specified as policies, and they will be used to check whether simulation satisfies these policies at runtime. This...
- [152] Thollot, R. (2012, April 3).  
Dynamic situation monitoring and Context-Aware BI recommendations. (M.-A. Aufaure, Ed.). Ecole Centrale Paris, MAS Laboratory, and SAP Research, Business Intelligence Practice.  
The amount of information generated and maintained by information systems and their users leads to the increasingly important concern of information overload. Personalized systems have thus emerged to help provide more relevant information and services to the user...
- [153] Thollot, R., Aufaure, M., & Gras, J. (2013, March 14).  
SITUATIONAL RECOMMENDATIONS IN HETEROGENOUS SYSTEM ENVIRONMENT. United States Patent Application.  
Situational recommendations in heterogeneous system environment are described herein. An event is received, where the event represents an interaction between an agent and a first resource from a number of resources available at the heterogeneous system ...
- [154] Boytsov, A., & Zaslavsky, A. (2013).  
Formal verification of context and situation models in pervasive computing. *Pervasive and Mobile Computing*, 9(1), 98-117.  
Pervasive computing is a paradigm that focuses on the availability of computer resources anytime anywhere for any application and supports non-intrusive integration of computing services into everyday life. Context awareness is the core feature of pervasive computing ...
- [155] Janiesch, G. (2010, January).  
Situation vs. context: considerations on the level of detail in modelling method adaptation. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on* (pp. 1-10). IEEE.  
Creating a universal conceptual modelling method, which can be used without modification in all situations, is not feasible. Appropriate methods for problem solving must be chosen, adapted or designed depending on the characteristics of the (pre-) supposed ...
- [156] Yau, S. S., Huang, D., Gong, H., & Seth, S. (2004, September).  
Development and runtime support for situation-aware application software in ubiquitous computing environments. In *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International* (pp. 452-457). IEEE.  
Due to the dynamic and ephemeral nature of ubiquitous computing (ubicom) environments, it is necessary that application software in ubicom environments is situationaware (SA) and should be adaptable to both users' situation changes and the ...
- [157] Yau, S. S., Wang, Y., Huang, D., & In, H. P. (2003, May).  
Situation-aware contract specification language for middleware for ubiquitous computing. In *Distributed Computing Systems, 2003. FTDGS 2003. Proceedings. The Ninth IEEE Workshop on Future Trends of* (pp. 93-99). IEEE.  
Ubiomp applications are characterized as situationaware, frequently-and-ephemerally-communicated and QoS-properties-associated. Using middleware to provide multiple QoS support for these ubiomp applications will enhance the development of the ...
- [158] Yau, S. S., Karim, F., Wang, Y., Wang, B., & Gupta, S. K. (2002).  
Reconfigurable context-sensitive middleware for pervasive computing. *IEEE Pervasive Computing*, 1(3), 33-40.  
A principal goal of pervasive computing is to make the actual computing part of it and its enabling technologies essentially transparent. 1-3 This transparency is partially possible because a pervasive computing environment is a collection of embedded, wearable, and ...
- [159] Kim, M., & Kim, M. (2009).

*A Formal Definition of Situation towards Situation-Aware Computing. Visioning and Engineering the Knowledge Society. a Web Science Perspective, 5736(Chapter 56), (pp. 553-563). Springer Berlin Heidelberg. [http://doi.org/10.1007/978-3-642-04754-1\\_56](http://doi.org/10.1007/978-3-642-04754-1_56)*

Recent works are trying to exploit the concept of situation for understanding and representing computing elements and environments in more comprehensive way beyond context. However, in such research, the situation is not defined and differentiated from the context clearly. Accordingly, the systems do not take advantage of the situation. In this paper, we provide a formal definition of the situation and differentiate it from the context clearly.

**[160] Loke, S. W. (2006).**

*On representing situations for context-aware pervasive computing: six ways to tell if you are in a meeting (pp. 5 pp.-39). Presented at the Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on IS - SN - VO -, IEEE. <http://doi.org/10.1109/PERCOMW.2006.102>*

Context-aware pervasive systems are emerging as an important class of applications. Such work attempts to recognize the situations of entities. This position paper notes three points when modelling situations: (1) there can be multiple ways to represent a situation; ...

**[161] Salber, D., Dey, A. K., & Abowd, G. D. (1999).**

*The context toolkit: aiding the development of context-enabled applications, 434-441. <http://doi.org/10.1145/302979.303126>*

Context is an important, yet poorly understood and poorly utilized source of information in interactive computing. It will be of particular importance in the new millennium as users move away from their desktops and into settings where their contexts are changing rapidly. Context is difficult to use because...

**[162] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002).**

*A survey on sensor networks. Communications magazine, IEEE, 40(8), 102-114.*

The availability of low-cost hardware such as CMOS cameras and microphones has fostered the development of Wireless Multimedia Sensor Networks (WMSNs), ie, networks of wirelessly interconnected devices that are able to ubiquitously retrieve multimedia content...

**[163] Dorn, C., Schall, D., & Dustdar, S. (2008, October).**

*Achieving team-awareness in scientific grid environments. In Grid and Cooperative Computing, 2008. GCC'08. Seventh International Conference on (pp. 75-83). IEEE.*

The vision of the Scientific Web is to enable research scientists to collaborate in a ubiquitous manner; sharing resources, data, and insights how to perform experiments. The grid provides the underlying computing infrastructure, thereby enabling workload sharing ...

**[164] Melgani, F., Serpico, S. B., & Vernazza, G. (2001).**

*Fusion of multitemporal contextual information by neural networks for multisensor image classification. In Geoscience and Remote Sensing Symposium, 2001. IGARSS'01. IEEE 2001 International (Vol. 7, pp. 2952-2954). IEEE.*

The analysis of a multitemporal sequence of images of a given site makes possible to exploit temporal information in addition to spectral and spatial information, and therefore represents a way to improve the accuracy with respect to the non-contextual single-time ...

**[165] INCOME: Arcangeli, J. P., Bouzeghoub, A., Camps, V., Canut, M. F., Chabridon, S., Conan, D., ... & Zaraté, P. (2012).**

*INCOME-multi-scale context management for the internet of things. In Ambient Intelligence (pp. 338-347). Springer Berlin Heidelberg.*

Nowadays, context management solutions in ambient networks are well-known. However, with the IoT paradigm, ambient information is not anymore the only source of context. Context management solutions able to address multiple network scales ranging ...

**[166] Zhang, Y., Zhang, Y., & Wang, W. (2005).**

*Policy Engineering for Security Management of Organization Information Systems. In LANOMS (pp. 289-294).*

Security management of information systems is a complex and daunting task in most organizations. Management policies are then introduced to guide and control the management of information systems, but the management of management policies is also ...

**[167] Cochran, E., & Bullemer, P. (1996).**

*Abnormal situation management: Not by new technology alone. In AIChE Plant Safety Symposium, Houston, TX.*

Abnormal Situations have always challenged operations personnel, and they likely always will. Abnormal Situation Management is a particular challenge at this point in history because the aggressive application of increasingly complex processes, sophisticated ...

**[168] Jakobson, G., Lewis, L., Matheus, C. J., Kokar, M. M., & Buford, J. (2005, October).**

*Overview of situation management at SIMA 2005. In Military Communications Conference, 2005. MILCOM 2005. IEEE (pp. 1630-1636). IEEE.*

This paper is an introduction to the Workshop on Situation Management, SIMA 2005. We discuss the scope of the workshop, the big picture of situation management, and a summarization of the papers selected for inclusion in the workshop. Topics include ...

**[169] CogSIMA 2011- 2014: Jakobson, G. (2014, March).**

*On modeling context in Situation Management. In Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2014 IEEE International Inter-Disciplinary Conference on (pp. 160-166). IEEE.*

One of the inherit features of cognitive situation awareness and decision support processes is that they are context dependent. This is hardly disputed by anybody, but as soon one wants to understand what this context dependency means, or even more, what ...

**[170] Jakobson, G., Buford, J., & Lewis, L. (2007).**

*Situation management: Basic concepts and approaches. In Information Fusion and Geographic Information Systems (pp. 18-33). Springer Berlin Heidelberg.*

This paper scopes the issues of Situation Management in dynamic systems, defines the basic concepts of Situation Management, and identifies several key enabling technologies. Particular focus of the paper is given to situation modeling. The paper ...

**[171] Adi, A., & Etzion, O. (2004).**

*Amit-the situation manager. The VLDB Journal-The International Journal on Very Large Data Bases, 13(2), 177-203.*

This paper presents the "situation manager", a tool that includes both a language and an efficient runtime execution mechanism aimed at reducing the complexity of active applications. This tool follows the observation that in many cases there is a gap between...

**[172] Porter, T. (2005).**

*The perils of deep packet inspection. Security Focus.*

*The Perils of Deep Packet Inspection The Perils of Deep Packet Inspection Dr. Thomas Porter 2005-01-11 Introduction This paper looks at the evolution of firewall technology towards Deep Packet Inspection, and then ...*

**[173] Cao, Y., Guan, J., Quan, W., Zhao, J., Xu, C., & Zhang, H. (2013).**

*User Behavior Prediction based Adaptive Policy Pre-fetching Scheme for Efficient Network Management.*

In recent years, network management is commonly regarded as an essential and promising function for managing and improving the security of network infrastructures. However, as networks get faster and network centric applications get more complex, there ...

**[174] NISTIR 7007: Hu, V., Lippmann, R., Haines, J., & Zissman, M. (2003).**

*An overview of issues in testing intrusion detection systems.*

While intrusion detection systems are becoming ubiquitous defenses in today's networks, currently we have no comprehensive and scientifically rigorous methodology to test the effectiveness of these systems. This paper explores the types of performance ...

- [175] **Gabriel, R., Hoppe, T., Pastwa, A., & Sowa, S. (2009, March).**  
*Analyzing malware log data to support security information and event management: Some research results.* In *Advances in Databases, Knowledge, and Data Applications, 2009. DBKDA'09. First International Conference on* (pp. 108-113). IEEE.  
 It is a well known fact in the fields of general business management research and business informatics that the efficient and effective processing of information through the use of adequate information systems constitutes an important driver for the success of an ...
- [176] **Luckham, D. C., & Frasca, B. (1998).**  
*Complex event processing in distributed systems.* Computer Systems Laboratory Technical Report CSL-TR-98-754. Stanford University, Stanford, 28.  
 Complex event processing is a new technology for extracting information from distributed message-based systems. This technology allows users of a system to specify the information that is of interest to them. It can be low level network processing data or high...
- [177] **Vermesan, O., & Friess, P. (Eds.). (2013).**  
*Internet of things: converging technologies for smart environments and integrated ecosystems.* River Publishers.  
 The book aims to provide a broad overview of various topics of the Internet of Things (IoT) from the research and development priorities to enabling technologies, architecture, security, privacy, interoperability and industrial applications. It is intended to be a stand...
- [178] **Costa, F., Silva, V., de Oliveira, D., Ocaña, K., Ogasawara, E., Dias, J., & Mattoso, M. (2013, March).**  
*Capturing and querying workflow runtime provenance with prov: a practical approach.* In *Proceedings of the Joint EDBT/ICDT 2013 Workshops* (pp. 282-289). ACM.  
 Scientific workflows are commonly used to model and execute large-scale scientific experiments. They represent key resources for scientists and are enacted and managed by Scientific Workflow Management Systems (SWfMS). Each SWfMS has its particular...
- [179] **Fülöp, L. J., Tóth, G., Rácz, R., Pánczél, J., Gergely, T., Beszédes, Á., & Farkas, L. (2010, July).**  
*Survey on complex event processing and predictive analytics.* In *Proceedings of the Fifth Balkan Conference in Informatics* (pp. 26-31).  
 Observing failures and other-desired or undesired-behavior patterns in large scale software systems of specific domains (telecommunication systems, information systems, online web applications, etc.) is difficult. Very often, it is only possible by examining the ...
- [180] **Cugola, G., & Margara, A. (2010).**  
*SLIM: Service location and invocation middleware for mobile wireless sensor and actuator networks.* *International Journal of Systems and Service-Oriented Engineering (IJSSOE)*, 1(3), 60-74.  
 One of the main obstacles to the adoption of Wireless Sensor Networks (WSNs) outside the research community is the lack of high-level mechanisms to easily program them. This problem affects distributed applications in general, and it has been replied by...
- [181] **Adi, A., & Etzion, O. (2002).**  
*The situation manager rule language.* In *RuleML* (Vol. 60, pp. 36-57).  
 This paper presents the "situation manager" rule language. The situation manager is a tool that includes both a language and an efficient runtime execution mechanism, aimed at reducing the complexity of active applications. It follows the observation that in many...
- [182] **Adi, A., Botzer, D., & Etzion, O. (2002).**  
*The situation manager component of Amit-active middleware technology.* In *Next Generation Information Technologies and Systems* (pp. 158-168). Springer Berlin Heidelberg.  
 Reactive applications are applications that include a substantial processing that is triggered by events. This paper describes an application development tool that resolves a major problem in this area: the gap that exists between events that are reported by various ...
- [183] **Costa, P. D., Mielke, I. T., Pereira, I., & Almeida, J. P. A. (2012, September).**  
*A model-driven approach to situations: Situation modeling and rule-based situation detection.* In *Enterprise Distributed Object Computing Conference (EDOC), 2012 IEEE 16th International* (pp. 154-163). IEEE.  
 This paper presents a model-driven approach to the specification of situations and situation detection. We offer two main contributions: (i) a Situation Modeling Language (SML), which is a graphical language for situation modeling, and (ii) an approach to ...
- [184] **Heckmann, D. (2006).**  
*Situation modeling and smart context retrieval with semantic web technology and conflict resolution.* In *Modeling and Retrieval of Context* (pp. 34-47). Springer Berlin Heidelberg.  
 We present a service to model situations and retrieve contextual information in mobile and ubiquitous computing environments. We introduce the general user model and context ontology Gumo for the uniform interpretation of distributed situational information ...
- [185] **Barford, P., Dacier, M., Dietterich, T. G., Fredrikson, M., Giffin, J., Jajodia, S., ... & Yen, J. (2010).**  
*Cyber SA: Situational awareness for cyber defense.* In *Cyber Situational Awareness* (pp. 3-13). Springer US.  
 Be aware of the current situation. This aspect can also be called situation perception. Situation perception includes both situation recognition and identification. Situation identification can include identifying the type of attack (recognition is only recognizing that an attack is occurring)...
- [186] **Yau, S. S., & Liu, J. (2006, April).**  
*Hierarchical situation modeling and reasoning for pervasive computing.* In *Software Technologies for Future Embedded and Ubiquitous Systems, 2006 and the 2006 Second International Workshop on Collaborative Computing, Integration, and Assurance. SEUS 2006/WCCIA 2006. The Fourth IEEE Workshop on* (pp. 6-pp). IEEE.  
 Situation awareness is one of the most fundamental features of entities in pervasive computing environments to dynamically adapt their behavior to situation changes to satisfy user requirements, including security and privacy. In order to support situation-aware...
- [187] **Costa, P. D., Guizzardi, G., Almeida, J. P. A., Pires, L. F., & van Sinderen, M. (2006, October).**  
*Situations in Conceptual Modeling of Context.* In *EDOC Workshops* (p. 6).  
 In previous work, we have defined conceptual foundations that can be beneficially used in context modeling. These conceptual foundations include the separation of entity and context, and the characterization of context as either Intrinsic or Relational. This paper...
- [188] **Ye, J., Dobson, S., & McKeever, S. (2012).**  
*Situation identification techniques in pervasive computing: A review.* *Pervasive and mobile computing*, 8(1), 36-66.  
 Pervasive systems must offer an open, extensible, and evolving portfolio of services which integrate sensor data from a diverse range of sources. The core challenge is to provide appropriate and consistent adaptive behaviours for these services in the face of huge...
- [189] **Zhang, Y., Liu, X., & Wang, W. (2005).**  
*Policy lifecycle model for systems management.* *IT professional*, 7(2), 50-54.  
 In an enterprise, policies are the glue that hold network and systems management activities, tools, managed resources, and process participants together. Nevertheless, gaps between management objectives, policy definitions, and policy enforcement are inevitable during ...
- [190] **XACML-V3.0. (2013) OASIS Standard.**  
*Organization for the Advancement of Structured Information Standards, "eXtensible Access Control Markup Language (XACML) Version 3.0", OASIS*



## Chapter 5

- [191] **Sloman, M. (1994).**  
*Policy driven management for distributed systems. Journal of network and Systems Management*, 2(4), 333-360.
- [192] **Zhang, Y., Liu, X., & Wang, W. (2005).**  
*Policy lifecycle model for systems management. IT professional*, 7(2), 50-54.
- [193] **Mitropoulos, S., Patsos, D., & Douligeris, C. (2006).**  
*On Incident Handling and Response: A state-of-the-art approach. Computers & Security*, 25(5), 351-370.
- [194] **Bergstra, J., Bethke, I., & Burgess, M. (2007).**  
*A process algebra based framework for promise theory. arXiv preprint arXiv:0707.0744.*
- [195] **Etzion, O., & Niblett, P. (2011).**  
*Event Processing in Action, Manning Publications.*  
Unlike traditional information systems which work by issuing requests and waiting for responses, event-driven systems are designed to process events as they occur, allowing the system to observe, react dynamically, and issue personalized data depending on the recipient and situation.
- [196] **Ji-chen, J., & Ming, G. (2006, October).**  
*Enterprise service bus and an open source implementation. In Management Science and Engineering, 2006. ICMSE'06. 2006 International Conference on (pp. 926-930). IEEE.*  
In the first part we give a referenced architecture of SOA which is composed of Application Representation, Business Integration, Service Components Architecture, Service Management, Service Modeling and Development, Information Integration, Enterprise ...
- [197] **Mühl, G., Fiege, L., & Pietzuch, P. (2006).**  
*Distributed event-based systems. Springer Science & Business Media.*  
In today's world, services and data are integrated in ever new constellations, requiring the easy, flexible and scalable integration of autonomous, heterogeneous components into complex systems at any time. Event-based architectures inherently decouple system ...
- [198] **Schmidt, M. T., Hutchison, B., Lambros, P., & Phippen, R. (2005).**  
*The enterprise service bus: making service-oriented architecture real. IBM Systems Journal*, 44(4), 781-797.  
The Enterprise Service Bus (ESB) is the infrastructure which underpins a fully integrated and flexible end-to-end service-oriented architecture (SOA). This paper details the essential meta-data and capabilities of the ESB. It presents a summary of the key concepts of the...
- [199] **Subramanian, N., & Chung, L. (2001, September).**  
*Software architecture adaptability: an NFR approach. In Proceedings of the 4th International Workshop on Principles of Software Evolution (pp. 52-61). ACM.*  
Adaptation of software systems is almost an inevitable process, due to the change in customer requirements, needs for faster development of new, or maintenance of existing software systems, etc. No doubt numerous techniques have been developed to deal with...
- [200] **Fox, J., & Clarke, S. (2009, June).**  
*Exploring approaches to dynamic adaptation. In Proceedings of the 3rd International DiscCoTec Workshop on Middleware-Application Interaction (pp. 19-24). ACM.*

## Chapter 6

- [208] **Break-Glass JIRA. (2004).**  
*An Approach to Granting Emergency Access to Healthcare Systems. medicalimaging.org. JOINT NEMA/COCIR/JIRA SECURITY AND PRIVACY COMMITTEE (SPC). Retrieved from <http://www.nema.org/medical/spc>*  
This white paper discusses a simple yet effective emergency-access solution, sometimes called "break-glass". The purpose of break-glass is to allow operators emergency access to the system in cases where the normal authentication cannot be successfully completed or is not working properly...

In this work, we compare current approaches to dynamic adaptation (DA) and identify the need for further research on mechanisms for DA, which should allow for higher compositionality and flexibility. Moreover, after exploring the research landscape in DA we...

- [201] **Cheaito, M. (2012).**  
*Un cadre de spécification et de déploiement de politiques d'autorisation (Doctoral dissertation, Université de Toulouse, Université Toulouse III-Paul Sabatier).*  
Notre travail propose une méthodologie de conception et de développement d'un système d'autorisation adaptable aux différentes facettes que peut recouvrir le contrôle d'accès dans les organisations telles que l'hétérogénéité des pratiques organisationnelles, des ...
- [202] **Papazoglou, M. P., Traverso, P., Dustdar, S., & Leymann, F. (2007).**  
*Service-oriented computing: State of the art and research challenges. Computer*, (11), 38-45.  
Service-oriented computing promotes the idea of assembling application components into a network of services that can be loosely coupled to create flexible, dynamic business processes and agile applications that span organizations and ...
- [203] **Cervantes, H., & Hall, R. S. (2004, May).**  
*Autonomous adaptation to dynamic availability using a service-oriented component model. In Proceedings of the 26th International Conference on Software Engineering (pp. 614-623). IEEE Computer Society.*  
This paper describes a project, called Gravity, that defines a component model where components provide and require services (ie, functionality) and all component interaction occurs via services. This approach introduces service-oriented concepts into a ...
- [204] **Laborde, R., Barrère, F., & Benzekri, A. (2013, April).**  
*Toward authorization as a service: a study of the XACML standard. In Proceedings of the 16th Communications & Networking Symposium (p. 9). Society for Computer Simulation International.*  
Cloud computing has promoted the notion of service as the leading way to deliver and consume computing resources. Today, security is going down that road and the term security as a service is emerging. Authorization that consists in managing permissions is ...
- [205] **Sprott, D., & Wilkes, L. (2004).**  
*Understanding service-oriented architecture. The Architecture Journal*, 1(1), 10-17.  
It seems probable that eventually most software capabilities will be delivered and consumed as services. Of course they may be implemented as tightly coupled systems, but the point of usage—to the portal, to the device, to another endpoint, and so on, will use a ...
- [206] **Hall, R., Pauls, K., McCulloch, S., & Savage, D. (2011).**  
*OSGi in action: Creating modular applications in Java. Manning Publications Co..*  
What is OSGi? Simply put, OSGi is a standardized technology that allows developers to create the highly modular Java applications that are required for enterprise development. OSGi lets you install, start, stop, update, or uninstall components without ...
- [207] **Balana August (2012).**
- [209] **Brucker, A. D., & Petritsch, H. (2009, June).**  
*Extending access control models with break-glass. In Proceedings of the 14th ACM symposium on Access control models and technologies (pp. 197-206). ACM.*  
Access control models are usually static, ie, permissions are granted based on a policy that only changes seldom. Especially for scenarios in health care and disaster management, a more flexible support of access control, ie, the underlying policy, is ...
- [210] **Garminati, B., Ferrari, E., & Guglielmi, M. (2011, October).**

- Secure information sharing on support of emergency management. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on* (pp. 988-995). IEEE.
- One of the most strategic component for an efficient and effective emergency management is the availability of an infrastructure on support of information sharing. Indeed, during emergency situations it is vital that the necessary information timely reaches the ...
- [211] **Ferreira, A., Antunes, L., Chadwick, D., & Correia, R. (2010).**  
Grounding information security in healthcare. *International Journal of Medical Informatics*, 79(4), 268-283.  
The objective of this paper is to show that grounded theory (GT), together with mixed methods, can be used to involve healthcare professionals in the design and enhancement of access control policies to Electronic Medical Record (EMR) systems. ...
- [212] **Marinovic, S., Craven, R., Ma, J., & Dulay, N. (2011, June).**  
Rumpole: a flexible break-glass access control model. In *Proceedings of the 16th ACM symposium on Access control models and technologies* (pp. 73-82). ACM.  
Access control operates under the assumption that it is possible to correctly encode and predict all subjects' needs and rights. However, in human-centric pervasive domains, such as health care, it is hard if not impossible to encode all emergencies and exceptions, ...
- [213] **Mozafari, B., Zeng, K., & Zaniolo, C. (2012, May).**  
High-performance complex event processing over xml streams. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data* (pp. 253-264). ACM.  
Much research attention has been given to delivering high-performance systems that are capable of complex event processing (CEP) in a wide range of applications. However, many current CEP systems focus on processing efficiently data having a simple ...
- [214] **Laborde, R., Kamel, M., Wazan, S., Barrere, F., & Benzekri, A. (2009).**  
A secure collaborative web-based environment for virtual organisations. *International Journal of Web Based Communities*, 5(2), 273-292.  
The concept of the Virtual Organisation (VO) is a natural outcome of network evolution and the growth of collaborative work tools. In the projects Value Improvement thought a
- Virtual Aeronautical Collaborative Enterprise (VIVACE) and Transglobal Secure Collaboration ...
- [215] **Ouyang, J. and Hokao, K. (2009)**  
Energy-saving potential by improving occupants behavior in urban residential sector in hangzhou city, china. *Energy and Buildings*.
- [216] **Holmes, T. G. (2007)**  
Eco-visualization: combining art and technology to reduce energy consumption. *Proceedings of the 6th ACM SIGCHI conference on Creativity & cognition*.
- [217] **Darby, S. (2006)**  
The effectiveness of feedback on energy consumption. A Review for DEFRA of the Literature on Metering, Billing and direct Displays, 486, 2006.
- [218] **Curry, E., Hasan, S., Banduk, M., & O'Riain, S. (2011).**  
Toward Situation Awareness for the Semantic Sensor Web: CEP with Dynamic Linked Data Enrichment.
- [219] **Jha, S., Kaiser, H., Merzky, A. and Weidner, O. (2007).**  
Grid Interoperability at the Application Level Using SAGA. (2007).
- [220] **Curry, E. (2012, July).**  
System of systems information interoperability using a linked dataspace. In *System of Systems Engineering (SoSE), 2012 7th International Conference on* (pp. 101-106). IEEE.
- [221] **Chadwick, D. W., Su, L., & Laborde, R. (2008).**  
Coordinating access control in grid services. *Concurrency and Computation: Practice and Experience*, 20(9), 1071-1094.
- [222] **Curry, E., Hasan, S., & O'Riain, S. (2012, October).**  
Enterprise energy management using a linked dataspace for energy intelligence. In *Sustainable Internet and ICT for Sustainability (SustainIT), 2012* (pp. 1-6). IEEE.
- [223] **Jie, W., Huang, Z., Daw, M., Procter, R., Li, X., Tang, L., & Lu, S. (2007).**  
Secure Access to Grid Information Service Using Shibboleth and PERMIS (pp. 297-304).
- [224] **Curry, E., Hasan, S., White, M., & Melvin, H. (2012).**  
An environmental chargeback for data center and cloud computing consumers. In *Energy Efficient Data Centers* (pp. 117-128). Springer Berlin Heidelberg.
- [225] **AlienVault Users Manual V 1.0, 2011**  
Alienvault OpenSource SIEM,  
<https://www.alienvault.com/products.php?section=OpenSourceSIEM>,
- [226] **Standard, O. A. S. I. S. (2013).**  
eXtensible Access Control Markup Language (XACML) Version 3.0. 22 January 2013.
- [227] **Arcangeli, J. P., Bouzeghoub, A., Camps, V., Canut, M. F., Chabridon, S., Conan, D., ... & Zaraté, P. (2012).**  
INCOME-multi-scale context management for the internet of things. In *Ambient Intelligence* (pp. 338-347). Springer Berlin Heidelberg.  
Nowadays, context management solutions in ambient networks are well-known. However, with the IoT paradigm, ambient information is not anymore the only source of context. Context management solutions able to address multiple network scales ranging ...
- [228] **Marie, P., Desprats, T., Chabridon, S., & Sibilla, M. (2013).**  
QoCIM: a meta-model for quality of context. In *Modeling and Using Context* (pp. 302-315). Springer Berlin Heidelberg.  
In the last decade, several works proposed their own list of quality of context (QoC) criteria. This article relates a comparative study of these successive propositions. The result is that no consensus has been reached about the semantic and the comprehensiveness ...
- [229] **Huebscher, M. C., & McCann, J. A. (2008).**  
A survey of autonomic computing—degrees, models, and applications. *ACM Computing Surveys (CSUR)*, 40(3), 7.  
Autonomic Computing is a concept that brings together many fields of computing with the purpose of creating computing systems that self-manage. In its early days it was criticised as being a “hype topic” or a rebadging of some Multi Agent Systems work. In this ...
- [230] **Truszkowski, W., Hallock, H., Rouff, C., Karlin, J., Rash, J., Hinchey, M., & Sterritt, R. (2009).**  
Autonomous and autonomic systems: with applications to NASA intelligent spacecraft operations and exploration systems. *Springer Science & Business Media*.  
In the early 1990s, NASA Goddard Space Flight Center started researching and developing autonomous and autonomic ground and spacecraft control systems for future NASA missions. This research started by experimenting with and developing expert systems to ...
- [231] **Truszkowski, W., Hallock, L., Rouff, C., Karlin, J., Rash, J., Hinchey, M. G., & Sterritt, R. (2009).**  
Autonomous and Autonomic Systems.  
Over the years, NASA missions have been gradually adding autonomy to flight and ground systems to increase the amount of science data returned from missions, perform new science, and reduce mission costs. In the early 1990s, a group of researchers at NASA ...

# Bibliography

- [1] **Hoffman, L. J. (1970).**  
*THE FORMILJARY MODEL FOR ACCESS CONTROL AND PRIVACY IN COMPUTER SYSTEMS.*  
*This thesis presents a model for engineering the user interface for large database systems in order to maintain flexible access controls over sensitive data. The model is independent of both machine and data base structure, and is sufficiently modular to allow ...*
- [2] **Stenzel, K. (1993).**  
*A Verified Access Control Model. Univ., Fak. für Informatik.*
- [3] **Abrams, M. D. (1995, August).**  
*Renewed understanding of access control policies. In Proceedings of the 16th National Computer Security Conference-Information System Security: User Choices (pp. 87-96).*
- [4] **Kuhn, D. R., Coyne, E. J., & Weil, T. R. (2010).**  
*Adding attributes to role-based access control. Computer, (6), 79-81.*  
*Role-Based Access Control. A US standard defined in ANSI/INCITS 359-2004. Information Technology—Role Based Access Control, RBAC controls all access through roles assigned to users. Each role assigns a collection of permissions to users. ...*
- [5] **Ni, Q., Bertino, E., & Lobo, J. (2008, June).**  
*An obligation model bridging access control policies and privacy policies. In Proceedings of the 13th ACM symposium on Access control models and technologies (pp. 133-142). ACM.*  
*In this paper, we present a novel obligation model for the Core Privacy-aware Role Based Access Control (P-RBAC), and discuss some design issues in detail. Pre-obligations, post-obligations, conditional obligations, and repeating obligations are supported by the ...*
- [6] **Kamel, M., Benzekri, A., Barrère, F., & Laborde, R. (2007).**  
*Evaluating the Virtual Organizations security solutions using the ISO/IEC 17799 standard. ICE2007, Sophia-Antipolis, France.*  
*A Virtual Organization (VO) network, a solution to collaborative environments, allows the participating organizations to share and exchange resources and competencies for economical or educational purposes; thus, they can realize common projects. Within ...*
- [7] **Peleg, M., Beimel, D., Dori, D., & Denekamp, Y. (2008).**  
*Situation-based access control: Privacy management via modeling of patient data access scenarios. Journal of biomedical informatics, 41(6), 1028-1040.*  
*Access control is a central problem in privacy management. A common practice in controlling access to sensitive data, such as electronic health records (EHRs), is Role-Based Access Control (RBAC). RBAC is limited as it does not account for the circumstances ...*
- [8] **Sandhu, R., & Park, J. (2003).**  
*Usage control: A vision for next generation access control. In Computer network security (pp. 17-31). Springer Berlin Heidelberg.*  
*The term usage control (UCON) is a generalization of access control to cover obligations, conditions, continuity (ongoing controls) and mutability. Traditionally, access control has dealt only with authorization decisions on a subject's access to target ...*
- [9] **Park, J., & Sandhu, R. (2004).**  
*The UCON ABC usage control model. ACM Transactions on Information and System Security (TISSEC), 7(1), 128-174.*  
*In this paper, we introduce the family of UCON ABC models for usage control (UCON), which integrate Authorizations (A), oBligations (B), and Conditions (C). We call these core models because they address the essence of UCON, leaving administration, ...*
- [10] **Karp, A. H., Haury, H., & Davis, M. H. (2010, April).**  
*From ABAC to ZBAC: the evolution of access control models. In Proceedings of the 5th International Conference on Information Warfare and Security, ed. EL Armistead (pp. 202-211).*  
*Several attempts at using the Services Oriented Architecture have failed to achieve their goals of scalability, security, and manageability. These systems, which base access decisions on the authentication of the requester, have been found to be inflexible, don't ...*
- [11] **Dini, P., Bochmann, G. V., Das, A., Dssouli, R., Dini, P., Bochmann, G. V., ... & Dssouli, R. (1993).**  
*Critical Review of Existing Configuration Management Methods.*
- [12] **Hegering, H. G., Abeck, S., & Neumair, B. (1999).**  
*Integrated management of networked systems: concepts, architectures, and their operational application. Morgan Kaufmann.*  
*This is a comprehensive book covering architecture, implementation, and operational use of all the current approaches to management-OSI/TMN, SNMP, CORBA, DMTF, and Web-based. It not only covers most of the modern approaches to management but also ...*
- [13] **RFC 4962: Housley, R., & Aboba, B. (2007).**  
*Guidance for authentication, authorization, and accounting (AAA) key management (No. RFC 4962).*  
*This document provides guidance to designers of Authentication, Authorization, and Accounting (AAA) key management protocols. The guidance is also useful to designers of systems and solutions that include AAA key management protocols. Given the complexity...*
- [14] **Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002).**  
*Wireless sensor networks: a survey. Computer networks, 38(4), 393-422.*
- [15] **De Oliveira, T. R., De Oliveira, S., Macedo, D. F., & Nogueira, J. M. (2011, October).**  
*An adaptive security management model for emergency networks. In Network Operations and Management Symposium (LANOMS), 2011 7th Latin American (pp. 1-4). IEEE.*
- [16] **Kott, A. (2014).**  
*Cyber Defense and Situational Awareness. C. Wang, & R. F. Erbacher (Eds.). New York: Springer.*
- [17] **Adi, A., Biger, A., Botzer, D., Etzion, O., & Sommer, Z. (2003, June).**  
*Context awareness in amit. In Autonomic Computing Workshop. 2003. Proceedings of the (pp. 160-166). IEEE.*
- [18] **Etzion, O., & Adi, A. (2003).**  
*U.S. Patent No. 6,604,093. Washington, DC: U.S. Patent and Trademark Office.*  
*A method for situation management includes specifying a composite event as a combination of two or more predefined component events and defining a rule, which causes a reaction to be invoked upon an occurrence of the composite event subject to a given condition. When ...*
- [19] **Jajodia, S., Liu, P., Swarup, V., & Wang, C. (2010).**  
*Cyber situational awareness (Vol. 14). New York, NY: Springer.*
- [20] **Benzekri, A., Barrère, F., & Laborde, R. (2013).**  
*Gestion des habilitations: modèles et architectures. REE. Revue de l'électricité et de l'électronique, (4), 35-41.*  
*Access control is of major importance in nowadays information systems, which are open, multi-domains and multi-suppliers. We address architectural and modeling issues of authorization systems allowing a clear separation of concerns between the...*
- [21] **Ersue, M., & Claise, B. (2012).**  
*An Overview of the IETF Network Management Standards (No. RFC 6632).*
- [22] **Verma, D. C. (2002).**



- Simplifying network administration using policy-based management. Network, IEEE, 16(2), 20-26.*
- [23] **Alzahrani, A. M. G. (2013).**  
*Efficient Enforcement of Security Policies in Distributed Systems.*
- [24] **Bächlin, M., & Tröster, G. (2009, December).**  
*Pervasive computing in swimming: A model describing acceleration data of body worn sensors in crawl swimming. In Pervasive Computing (JGPC), 2009 Joint Conferences on (pp. 293-298). IEEE.*
- [25] **Bandara, A. K., Lupu, E. C., Moffett, J., & Russo, A. (2004).**  
*A goal-based approach to policy refinement (pp. 229-239). Presented at the Policies for Distributed Systems and Networks, 2004. POLICY 2004. Proceedings. Fifth IEEE International Workshop on. doi:10.1109/POLICY.2004.1309175*
- [26] **Bicocchi, N., Cecaj, A., Fontana, D., Mamei, M., Sassi, A., & Zambonelli, F. (2013).**  
*Collective Awareness for Human-ICT Collaboration in Smart Cities (pp. 3-8). Presented at the Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2013 IEEE 22nd International Workshop on IS -, IEEE. doi:10.1109/WETICE.2013.54*
- [27] **Boytsov, A., & Zaslavsky, A. (2013).**  
*Formal verification of context and situation models in pervasive computing. Pervasive and Mobile Computing, 9(1), 98-117. doi:10.1016/j.pmcj.2012.03.001*
- [28] **Chen, Yunan, & Xu, H. (2013).**  
*Privacy management in dynamic groups: understanding information privacy in medical practices. Presented at the CSCW '13: Proceedings of the 2013 conference on Computer supported cooperative work, ACM Request Permissions. doi:10.1145/2441776.2441837*
- [29] **Duflos, S., Diaz, G., Gay, V., & Horlait, E. (2002).**  
*A comparative study of policy specification languages for secure distributed applications. In S. Duflos, G. Diaz, V. Gay, & E. Horlait (Eds.), (Vol. 2506, pp. 157-168). Presented at the Trust Management, Proceedings, Berlin, Heidelberg. doi:10.1007/3-540-36110-3\_16*
- [30] **Fong, P. W. L., & Siahaan, I. (2011a).**  
*Relationship-based access control policies and their policy languages. Proceedings of the 16th ACM Symposium on Access Control Models and Technologies, 51-60.*
- [31] **Frenzel, C., Sanneck, H., & Bauer, B. (2013).**  
*Rational Policy System for Network Management (pp. 776-779). Presented at the Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on IS - SN - VO -, IEEE.*
- [32] **Gama, P., & Ferreira, P. (2005).**  
*Obligation policies: an enforcement platform. Policies for Distributed Systems and Networks, 2005. Sixth IEEE International Workshop on, 203-212. doi:10.1109/POLICY.2005.18*
- [33] **Gates, C. E. (2007). (Ref 77)**  
*Access Control Requirements for Web 2.0 Security and Privacy, 1-3.*
- [34] **Henricksen, K., & Indulska, J. (2006).**  
*Developing context-aware pervasive computing applications: Models and approach. Pervasive and Mobile Computing, 2(1), 37-64. doi:10.1016/j.pmcj.2005.07.003*
- [35] **Horvath, L., & Rudas, I. J. (2013).**  
*Situation based feature definition on product behavior level (pp. 245-250). Presented at the Applied Computational Intelligence and Informatics (SACI), 2013 IEEE 8th International Symposium on IS - SN - VO -, IEEE. doi:10.1109/SACI.2013.6608976*
- [36] **Indulska, J., & Nicklas, D. (2010).**  
*Introduction to the special issue on context modelling, reasoning and management. Pervasive and Mobile Computing, 6(2), 159-160.*
- [37] **Jajodia, S., Samarati, P., & Subrahmanian, V. (1997).**  
*A logical language for expressing authorizations. Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on, 31-42.*
- [38] **Januario, G. C., Costa, C. H. A., Amarai, M. C., Riekstin, A. G., Carvalho, T. C. M. B., & Meirosu, C. (2013).**  
*Evaluation of a policy-based network management system for energy-efficiency (pp. 596-602). Presented at the Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on IS - SN - VO -, IEEE.*
- [39] **Kephart, J., & Walsh, W. (2004).**  
*An artificial intelligence perspective on autonomic computing policies. Audio, Transactions of the IRE Professional Group on, 3-12. doi:10.1109/POLICY.2004.1309145*
- [40] **Kirsch-Pinheiro, M., & Rychkova, I. (2013).**  
*Dynamic Context Modeling for Agile Case Management. In Y. Demey & H. Panetto (Eds.), Lecture Notes in Computer Science (Vol. 8186, pp. 144-154-154). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-642-41033-8\_21*
- [41] **Ko, E.-N., & Bang, K. C. (2006).**  
*An Intelligent Error Detection Model for Reliable QoS Constraints Running on Pervasive Computing (pp. 518-521). Presented at the Artificial Reality and Telexistence-Workshops, 2006. ICAT '06. 16th International Conference on IS - SN - VO -, IEEE. doi:10.1109/ICAT.2006.41*
- [42] **Laborde, R., & Desprats, T. (n.d.).**  
*An Extension of XACML to Improve the Performance of Decision Making Processes When Dealing with Stable Conditions. In L. Boursas, M. Carlson, W. Hommel, M. Sibilla, & K. Wold (Eds.), Communications in Computer and Information Science (Vol. 18, pp. 13-24). Springer Berlin Heidelberg.*
- [43] **Lee, K. X. K., Jiang, Z., Kim, S., & Kim, S. (2005).**  
*Security policy management for healthcare system network (pp. 289-292). Presented at the Enterprise networking and Computing in Healthcare Industry, 2005. HEALTHCOM 2005. Proceedings of 7th International Workshop on IS - SN - VO -, IEEE. doi:10.1109/HEALTH.2005.1500463*
- [44] **Lin, X., Li, S., Yang, Z., & Shi, W. (2005).**  
*Application-oriented Context Modeling and Reasoning in Pervasive Computing (pp. 495-501). Presented at the Computer and Information Technology, 2005. CIT 2005. The Fifth International Conference on IS - SN - VO -, IEEE. doi:10.1109/CIT.2005.77*
- [45] **Macfarlane, R., & Buchanan, W. (2011).**  
*Review of security policy implementations. ... & Security. doi:10.1016/j.cose.2011.10.003*
- [46] **Michel, J., Julien, C., Payton, J., & Roman, G. (2012).**  
*A spatiotemporal model for ephemeral data in pervasive computing networks (pp. 179-184). Presented at the Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on IS - SN - VO -, IEEE. doi:10.1109/PerComW.2012.6197474*
- [47] **Ohno-Machado, L., Silveira, P. S. P., & Vinterbo, S. (2004).**  
*Protecting patient privacy by quantifiable control of disclosures in disseminated databases. Realizing Security Into the Electronic Health Record, 73(7-8), 599-606.*
- [48] **Ranganathan, A., Chetan, S., Al-Muhtadi, J., Campbell, R. H., & Mickunas, M. D. (2005).**  
*Olympus: A High-Level Programming Model for Pervasive Computing Environments (pp. 7-16). Presented at the*

- Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on IS - SN - VO -, IEEE. doi:10.1109/PERCOM.2005.26*
- [49] **Roy, N., Gu, T., & Das, S. K. (2010).**  
*Supporting pervasive computing applications with active context fusion and semantic context delivery. Pervasive and Mobile Computing, 6(1), 21-42. doi:10.1016/j.pmcj.2009.10.002*
- [50] **Sayenko, V., & Panchenko, A. (2001).**  
*Design of policy based network traffic management system (pp. 227-229). Presented at the CAD Systems in Microelectronics, 2001. CADSM 2001. Proceedings of the 6th International Conference. The Experience of Designing and Application of IS - SN - VO -, Lviv Polytechnic Nat. Univ. doi:10.1109/CADSM.2001.975819*
- [51] **Turkmen, F., & Crispo, B. (2008).**  
*Performance evaluation of XACML PDP implementations. Presented at the SWS '08: Proceedings of the 2008 ACM workshop on Secure web services, ACM Request Permissions. doi:10.1145/1456492.1456499*
- [52] **Dey, Anind K. (2001).**  
*Understanding and Using Context. » Personal Ubiquitous Comput. 5 (1): 4-7. doi:10.1007/s007790170019.*  
*Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves*
- [53] **Turner, K. J., Reiff-marganiec, S., Blair, L., Campbell, G. A., Wang, F., Turner, K. J., et al. (2011).**  
*Appel: An Adaptable and Programmable Policy Environment and Language (No. CSM-161). cs.stir.ac.uk. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.135.9068>*
- [54] **Unger, F. F. (2012).**  
*Health is wealth: considerations to european healthcare. Prilozi / Makedonska Akademija Na Naukite I Umetnostite, Oddelenie Za Bioloski I Medicinski Nauki = Contributions / Macedonian Academy of Sciences and Arts, Section of Biological and Medical Sciences, 33(1), 9-14.*
- [55] **van der Linden, H., Kalra, D., Hasman, A., & Talmon, J. (2009).**  
*Inter-organizational future proof EHR systems: A review of the security and privacy related issues. Realizing Security Into the Electronic Health Record, 78(3), 141-160. doi:10.1016/j.ijmedinf.2008.06.013*