CxtBAC: A family of context-based access control models for Pervasive Environments

José de Ribamar Martins Bringel Filho

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Informatique**

Arrêté ministérial : 7 août 2006

Présentée et soutenue publiquement par

M. José de Ribamar Martins BRINGEL FILHO

le **22 octobre 2010**

CxtBAC : Une Famille de Modèles de Contrôle d'accès Sensible au Contexte pour les Environnements Pervasifs

---

Thèse dirigée par **M. Hervé MARTIN**

Jury

M. Didier DONSEZ
Professeur d'informatique, Université Joseph Fourier (Grenoble 1), **Président, Examinateur**
M. Bruno DEFUDE
Professeur d'informatique, Institut TELECOM SudParis, **Rapporteur**
M. Robert LAURINI
Professeur d'informatique, Institut National des Sciences Appliquées de Lyon (INSA-Lyon), **Rapporteur**
Mme Rossana ANDRADE
Professeur d'informatique, Université Federal du Ceará (Brésil), **Examinatrice**
M. Hervé MARTIN
Professeur d'informatique, Université Joseph Fourier (Grenoble 1), **Directeur de thèse**

Thèse préparée au sein du **Laboratoire d'Informatique de Grenoble (LIG)**
dans **L'Ecole Doctorale MSTII**

Remerciements

En prenant en compte l'historique des relations contextuelles et sociales établis pendant ces quatre années de travail de thèse, je tiens à accorder l'accès de lecture sur le message de remerciement aux groupes suivants des utilisateurs.

Premièrement, je tiens à remercier à ceux qui sont annoté comme *rapporteur* (M. Bruno DEFUDE et M. Robert LAURINI), d'avoir accepté d'évaluer ce travail sous forte contrainte de temps. Également, je tiens à remercier le groupe d'*examinateur* composé par M. Didier DONSEZ et Mme. Rossana ANDRADE (merci d'être venu si loin), d'avoir accepté de participer du jury de ce travail.

Un remerciement tout particulier à Hervé Martin (le seul du group *directeur-de-thèse*) pour ses conseils, sa patience, son amitié et le soutien tout au long de ces quatre années de thèse, d'une importance fondamentale pour l'aboutissement de ce travail.

Je tiens à remercier fortement le group d'*organisme de financement* composé par *Alban - European Union Programme of High Level Scholarships for Latin America, scholarship no. E06D104158BR* (36 mois), *NII - National Institute of Informatics* (4 mois), et l'équipe *STEAMER* (8 mois). En spécial, je tiens à remercier Jérôme, Marlene et Paule-Annick qui m'avait proposé un financement dans le cadre du projet Urbasis. Ce soutien financier a été essentiel pour la réalisation de ce travail dans les meilleures conditions.

Je voudrais remercier à toute l'équipe STEAMER pour le soutien scientifiques, administratives, d'infrastructure et pour l'environnement convivial de travail. Aussi, je tiens à remercier aux collèges de bureau (D322) Marius, Dia et Christine, pour les bonnes moments partagé pendant toutes ces années. Je tiens à remercier en particulier Christine, une amie qui m'a aidé depuis le début de cette aventure (l'apprentissage du français, résoudre des problèmes personnels, etc) et qui a suivi de plus près toutes les difficultés rencontrées, toujours avec un mot de réconfort et d'amitié.

Je ne pouvais pas oublier de remercier mes collègues d'équipe qui ont également participé d'une partie de ce long parcours: Manuele, Céline, Angela, Aurélie, Bogdan, Windson, Raphael, Laurent, Benoit, Anton, Dounia et Reinaldo. Aussi, aux collèges du LIG Bogdan, Carina, Luz Maria, Fred, Yan, Vincent, Asif, Naga et tout les autres pour les chaleureuses moments partagés.

Je tiens à remercier fortement au énorme group *portugais-brésilien* composé par Carlos Tadeu, Magrão (Edson), Allen, Afonso, Carol, Beatriz, César (un faux brésilien plus vrai que de nombreux d'autres), Luciano, Andréa, Leo, Marina, Lucas, Fabi, Herman, Ana, Hyane, Pedro, Patricia, Pericles, Joelma, Hannah, Patrick, Fany, Elton, Raquel, Isaac, Shirley, Paulo Henrique, Raquel, Cecilia, et tout les autres pour les bonnes moments partagé autour d'un *churrasco*. Aussi, aux amis du group *vie-extérieur* composé par Marque, Cinara, M. Coral, Bernard, Aida, Emma,

Hamed et votre famille, pour leurs amitiés qui reste encore très fort.

Mes remerciements s'adressent également à toutes les personnes avec qui je eu l'opportunité de travailler et discuter pendant la durée de cette thèse, en particulier, Celso, Nadine, Léa, Mauro, M. Ichiro Satoh et Christian.

Je voudrais également remercier à mes parents (source d'inspiration et de force), à mes frères (amis pour toute la vie), mes cousins (en spécial à Breno et Mirna), ma tante Mirtes, ma grand-mère Luiza (in memoriam), mes oncles Ezio et Constancio (in memoriam), mon tonton Luiz Carlos et à ma belle famille. Merci beaucoup pour le soutien inconditionnel que vous m'avez apporté tout au long de mes études, je vous aime.

Finalement mais pas moins important, je tiens à remercier ma femme Suely (ma fleur) et ma belle-fille (Gabriella, ma princesse) pour avoir partagé, jour après jour, de toutes les difficultés (qui n'étiez pas rares) et les moments de joie pendant ces quatre longes années. Merci pour m'avoir accompagné dans cette dure aventure à l'outre-mer. C'est grâce à vous et surtout pour vous que je dédie ce travail. Je vous aime au fond du coeur.

Merci beaucoup à tous !!

List of Figures

List of Tables

Contents

Part I

General Introduction

Chapter 1

Introduction

Résumé: *Ce chapitre introduit la problématique dans laquelle ce travail est situé - contrôle d'accès pour les environnements pervasifs. Nous présentons les difficultés rencontrées pour appliquer les mécanismes de sécurités existantes afin de protéger les ressources en prenant en compte les scenarios distribués considérées par ce travail. Nous faisons le lien avec les besoins au niveau de modélisation et support à des politiques de sécurités flexibles et sensibles au contexte. Des problèmes liés au contrôle d'accès dans les environnements pervasifs est actuellement un défi essentiellement dû au contexte et au constant changement de comportements des utilisateurs. Nous décrivons aussi les problèmes liés au l'impacte de la vie privée et de la qualité du contexte sur les opérations de définition et vérification des politiques de sécurité sensible au contexte. Également, la motivation, les contributions et la distribution des autres chapitres de la thèse sont décrites dans ce chapitre.*

Introduction

The development of pervasive computing environments (PCE) is becoming a reality with growing advancement of mobile computing devices (e.g., smartphones with embedded sensors), sensors (e.g., RFID, indoor and outdoor location sensors), and wireless communication technologies, such as Wi-Fi (IEEE 802.11a/b/g/n [802.11 2009]), Bluetooth (IEEE 802.15.3 [802.15 2009]), ZigBee (IEEE 802.15.4 [802.15 2009]), 3G (e.g., WiMAX IEEE 802.16 [802.16 2009]), and 4G (e.g., Gigabit WiMAX IEEE 802.16m [802.16 2009]).

As computers become more pervasive in our daily life, new applications will emerge to provide users unobtrusive access to information, resources, and services. Clearly, the successful deployment of such applications will depend on our ability to secure them [Covington 2001]. Moreover, due to the dynamicity of the pervasive environment and the mobility of users, access to resources available on the environment must be granted taking into account the current situation. Therefore, this thesis addresses a major issue for pervasive computing environments (PCE): access control of *pervasive resources*[1].

[1] In this thesis the term pervasive resource refers to any kind of digital object that can be protected by an access control solution running in a pervasive environment. For instance, it can be a service, a file, an information, a printer or a URL.

Sensor-rich pervasive devices offer users, anytime and anywhere, new opportunities for creating, visualizing, retrieving, and sharing resources, such as personal documents (e.g., photos, videos), situational information (e.g., user location), and services (e.g., web services). This type of resource, namely *pervasive resource*, can be created, modified, and accessed by users in indoor or outdoor mobility situations.

In fact, nowadays it is possible to use multiple sensors embedded in pervasive devices (e.g., GPS, Bluetooth, temperature, luminosity, accelerometer, compass, proximity) to characterize the current situation of users (e.g., their location, activity and nearby pervasive devices) in order to access adapted services and information [Filho 2010b]. This kind of data is widely known in the scientific community as context information [Dey 2001]. Moreover, such contextual information can be associated automatically with pervasive resources at creation time (e.g., photos, videos) or at request time (e.g., web services), in order to improve retrieving, organization, and sharing operations of resources [Viana 2008].

From an access control point of view, the ability of pervasive devices to dynamically create and access content interacting with other surrounding devices, results in the need to provide access control mechanisms that are flexible and sensitive to the changing situations. Traditional access control solutions, such as Role-Based Access Control (RBAC) [Ferraiolo 1992], typically evaluate permission depending on the identity/role of users that are requesting access to resources. However, pervasive environment provides access on resources often to unknown entities whose identity may be uninformative or not sufficiently trustworthy [Corradi 2004a]. In fact, it is almost impossible for service providers to know in advance the identities or roles of all subjects that are likely requesting access to protected pervasive resources.

Therefore, pervasive environments call for new access control models, policies, and enforcement mechanisms that are able to dynamically adjust permission. Such adjustments should be made, if possible, by taking into account the current situation of *observed entities*[2], such as users and their actual environment.

In order to make it possible, PCE should be able to identify the current situation in which users are part, commonly known as *context*, in order to accordingly adjust permission. Dey *et al.* [Dey 2001] have proposed the following general definition of context, which is widely referenced by the scientific community: *"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."*

Currently, it is increasingly common for people to have a personal pervasive device, enabling them to share and to access resources dynamically with other people. By using their devices, users are able to simultaneously assume the role of consumers and providers of resources. Moreover, pervasive environments are rich in

[2]In this thesis the term observed entities refer to any entity that the system is able to observe and to characterize their situation. For example, it can be the resource owners, resource requestors, resources and the environment around them.

peer-to-peer interactions, where social relationships between users and information characterizing their current situation should be exploited for making access control decisions. In fact, pervasive devices should participate actively in the process of identifying the current situation of its users, by incorporating a personal context management entity embedded on these devices. Thus, context information could be used as a way of having dynamically and transparently access to resources accordingly to the current situation.

Several scientific and technological challenges should be addressed in order to allow the development of flexible and context sensitive access control systems for pervasive environments, as described previously. Among these challenges we have identified:

- *Support to context-awareness:* the specification of access control models should take into account the current context of users, resources, and the environment around them;
- *Privacy and quality-aware management of context information:* context information used to make access decisions should have high quality. Moreover, the privacy of users should be ensured by the context management framework. In summary, context management entities should provide context information with quality to context-based access control approaches, protecting user's private context information from misuse.

In following section we describe in detail each one of these challenges and our research motivation.

1.1 Research motivation

Emerging pervasive computing environments need access control mechanisms that are non-intrusive and easily adaptable to the context. Unfortunately, traditional access control mechanisms are unaware to the changing situations, requiring a complex and static authentication infrastructure in which a user has to identify himself (e.g., username and password) in order to access protected pervasive resources. However, in environments characterized by ad-hoc interactions (user-to-user and user-to-environment interactions) and users in mobility situations, access control to resources could rather preferably depend on the current context of users than on their identities [Hulsebosch 2005].

Pervasive user-generated content needs to be controlled by flexible access control approaches, offering means of defining policies based on information that characterizes the situation of users, resource, and the environment around them. Users should be able to define access control policies for protecting dynamically their resources, moving from traditional centralized access control approaches based on

identities/roles to context sensitive user-centric solutions. By coupling access control to context information, access control services can become far more user friendly and flexible.

In fact, PCE introduces new security requirements that can not be solved by traditional access control models, such as Mandatory Access Control (MAC)[Sandhu 1994], Role-based Access Control (RBAC) [Ferraiolo 1992], and Discretionary Access Control (DAC)[Harrison 1976]. These access control approaches were initially specified for closed and relatively unchangeable distributed systems, dealing with a set of known users who access a set of known resources. Furthermore, they do not take into account information that can characterize the situation of resource owners, resource requestors, resources, and the environment around them, for determining whether permission should be allowed or not [Filho 2009].

In a RBAC-based system, for instance, each user should be explicitly associated with one or more roles in order to have permission [Ferraiolo 2003]. By contrast, in a PCE we cannot make the assumption that the system is able to know previously the identity of all users, assigning roles to them. Consequently, by using a RBAC-based solution the system will be unable to apply correctly user-role associations. For example, in a conference workshop carried out in a PCE, the collaborative relationships between the participants are established in a dynamic and unpredictable manner. In this case, users are unable to determine previously the persons that will interact with them, possibly sharing or accessing resources, such as a presentation file or a research paper.

In order to grant permission in a RBAC-based system, the Office of Information Technology (OIT) should create an identity for each user and assign to it one or more roles, i.e., it is necessary to define previously the *user-role associations*. Then, users will get the permission associated with the roles assigned to them in the current session. However, these administration tasks can be simplified if permission could be granted to users according to the current situation, e.g., activity of users (e.g., participating in the workshop), location (conference room), and time (session time). For example, the administrator could define a policy to grant read access on the presentation file to everyone located in the conference room at session time.

In the real world, people are aware of their situation in a spontaneous and transparent manner, which are conscious of what is happening around them and understand how information, events, and their own actions will impact their goals and objectives, both now and in the near future. However, we can not assert the same statements for PCE. Such environments need to be explicitly informed about the situation of users in order to adapt their decisions. To achieve this feature, PCE should have many distributed entities for gathering, interpreting, deriving, inferring, and providing context information with quality to context sensitive applications and services [Filho 2010a].

In fact, as context information represents real-world situations, it is associated

with certain quality features, named *Quality of Context (QoC)*, which need to be observed and evaluated. For example, the system should be able to answer the following questions: *How old is this location information? Could we trust this information?* Furthermore, entities in charge of context management operations should protect it as well as the entities that construct and disseminate that information, possibly taking into account user's privacy requirements, named *Privacy of Context (PoC)*. For instance, users could define a security policy to disclose their location information only to their family at weekends. Moreover, such users could disclose their exact location (e.g., GPS coordinates) to everyone but with a delay of three hours.

According to the previous example we can conclude that QoC might decrease by enforcing user's privacy policies. Obviously, the quality of context information has a strong impact on the correctness of context-based services and applications. For instance, if a user who has disclosed their location with a delay of three hours requests a location-based service, then the response of this service will not be current and may not be more useful.

We describe in the following some characteristics of pervasive environments that should be considered when proposing new context sensitive access control approaches:

- **Spontaneous and unpredictable interactions between consumers and providers of resources:** it is not always possible to predict the interactions between consumers and providers of resources, since users are characterized by mobility situations, sometimes accessing/providing resources from/to others pervasive devices. Therefore, users could access services from other pervasive environments that do not belong to their main domain;

- **Pervasive environments are dynamics:** in most case, there is not a well defined organizational infrastructure. Pervasive devices can establish dynamically ad-hoc interactions with other surround devices, providing/accessing resources. In this case, the availability of resources changes with time. They can pass from the state available to unavailable, unexpectedly;

- **Resource discovery:** pervasive devices should be able to transparently and dynamically discover the pervasive resources available in the environment, according to the current situation;

- **Absence of property:** if there is an organizational infrastructure (e.g., a corporate building) behind a pervasive environment, possibly some available resources (e.g., printers, web services) are not owned by a particular user, but by the organization. In this case, users do not have the control of these resources (i.e., it is necessary to support system-level access control policies to protect them);

- **Access control requirements are dependent on the application:** according to the scenario of application (e.g., personal multimedia application, corporate management systems, smart homes) there is a different set of users that can interact with other users and resources, a set of resource types that can be controlled, a set of permission that can be assigned, and who is in charge of defining access control policies to protect the resources;
- **Diversity of entities in charge of context information monitoring operations:** there exist several entities that collaborate for constructing a global vision of the current situation in which users are part. For instance, we can use sensors embedded in personal pervasive devices and sensors distributed in the environment;
- **Quality and privacy of context information:** the quality of context information used for making decisions impacts directly on the correct behavior of context sensitive applications and services, such as an access control system. Moreover, it is necessary to offer mechanisms to protect context information against misuse since this contains personal information of users, such as their location and activity.

In the next section, we present briefly the main contribution of our work. We have taken into account the characteristics described in this section to propose a family of context-based access control model for PCE.

1.2 Thesis contribution

The development of an access control system is usually carried out with a multi-phase approach based on the following concepts: *access control policies*, *access control models*, and *access control mechanisms* [Samarati 2001]. *Access control policies* are, in fact, high-level rules that specify how access is managed by the access control mechanisms and which may access resources under what circumstances. *Access control policies* are enforced through an *access control mechanism* that translates access requests in terms of a structure provided by the access control system. For instance, an Access Control List (ACL)[3] is a familiar access control mechanism.

Access control models bridge the gap in abstraction between an *access control policy* and the *access control mechanism* implementing them. An access control model provides a formal representation of how access control policy works, allowing to verify properties and the security provided by the designed access control system. Rather than attempting to evaluate and to analyze access control systems exclusively at the mechanism level, access control models are usually written to describe the

[3] ACL is a common access control solution supported by most of the Unix and Unix-like operating systems, such as Linux, Ubuntu, Fedore, and OpenSUSE. An ACL specifies which users or system processes are granted access to objects, as well as what operations are allowed on given objects.

security properties of an access control system. Therefore, access control models are formal representations of the security policy enforced by the system and are useful for proving theoretical limitations of an access control system.

This work proposes a family of *Context-Based Access Control* models, namely $CxtBAC$, which captures access control requirements of pervasive environments. $CxtBAC$ models could be used as basis specification to develop access control mechanisms that use the context as the main concept for defining access control policies. Unlike traditional access control models, usually based on association between users and permission statically defined, in $CxtBAC$-based mechanisms this association is made dynamically according to the current situation of access, namely *access context*.

Therefore, $CxtBAC$ specifies context-based access control models, providing a basis for implementing real access control approaches and for proposing context-based access control policies. $CxtBAC$ is policy neutral and independent of implementing aspects. $CxtBAC$ family is composed by 8 (eight) elements, which was gradually defined through adding the support for new access control requirements: $CxtBAC_0$ (Base model), $CxtBAC_1$ (Access Control Hierarchies), $CxtBAC_2$ (Constraints), $CxtBAC_3$ (The Core), $Q - CxtBAC$ (Quality-Aware CxtBAC), $P - CxtBAC$ (Privacy-Aware CxtBAC), $S - CxtBAC$ (Social-Aware CxtBAC), and $QPS - CxtBAC$ (Quality, Privacy and Social-Aware CxtBAC).

This work proposes also a Quality and Privacy-Aware Context Management Framework, namely $CxtMF$, in order to provide context information to context-based access control systems implemented following the $CxtBAC$ specification. It is in charge of context management operations, considering the quality and privacy aspects of context management in its various layers. Furthermore, $CxtMF$ provides QoC-enriched context information to context-aware services and applications, such as a context-based access control system.

We have defined a general context model (*Context Ontology*) that can be easily extended to accommodate specific requirements of context sensitive applications [Filho 2010a]. This model is supported by the *CxtMF*, which makes the framework reusable for different context sensitive applications and services. We use web technologies, such as Web Ontology Language (OWL) [OWL 2009], for modeling, deriving, and inferring new context information from raw context data. From the *Context Ontology* model we define the *Access Context Ontology* to describe the situation of any relevant entity for a $CxtBAC$-based mechanism: resource owners, resource requestors, resources, and the environment around them.

In order to provide context information with quality, we propose also a quality of context model (*QoC Ontology*) that can be used to describe semantically the quality information associated with each context concept (e.g., location, activity). Moreover, we defined QoC evaluating methods that can be dynamically deployed by the $CxtMF$, providing QoC-enriched context information to context sensitive

applications and services.

In fact, *CxtBAC* is an active access control model that supports system/user-level context-based access control policies, i.e., CxtBAC can be instantiated to support both, mandatory and discretionary policies. By enforcing predefined policies, a *CxtBAC*-based system grants dynamically permission, taking into account information that characterizes the current access context. We have defined *access context* as *any information that can be used to characterize the situation of an observed entity that is relevant for making access control decisions.* These entities can be resource requestors, resource owners, resources, and the environment around them. Therefore, we have extended the context dimensions supported by the existing context sensitive access control approaches, which normally consider only the context of users and the environment when evaluating access control policies.

The administration tasks of a *CxtBAC*-based system could be deployed by following two approaches:

- *Distributed user-centric approach (discretionary):* users are able to define access control policies for protecting their resources (i.e., user-level access control policies);
- *Centralized approach (mandatory):* there is an administration office in charge of access control administration tasks (i.e., system-level access control policies).

We have instantiate a *CxtBAC model* (*Social-Aware CxtBAC*) to protect resources of mobile multimedia applications. We have developed an application, named *PPlog - Pervasive Personal Blog*, to demonstrate how mobile multimedia applications can use CxtBAC to construct access control mechanisms integrated with *CxtFM* to protect personal multimedia resources.

1.3 Dissertation outline

This document is divided into two parts: general introduction and proposition. The introduction presents the main research topics related to this work. Firstly, we discuss general concepts related to access control systems and the existing access control approaches, emphasizing the RBAC and RBAC extended models. Then, we describe in detail existing access control approaches that use context information to make access control decisions or simply to improve policy enforcing mechanisms. We have divided these existing solutions into two groups: *context-aware access control* (CAAC) and *context-based access control* (CBAC) approaches.

During the development of this work, we have observed the importance of verifying the quality of context information (QoC) used by CAAC and CBAC systems, in

order to ensure the correct behavior of policy enforcing mechanisms. Furthermore, these systems need to ensure the privacy of context information (PoC) used for making access control decisions, taking into account the user's privacy requirements. In summary, CAAC and CBAC approaches normally need to use context information with quality and, in some usage scenarios, to ensure the privacy of users. Therefore, even in this first part we present existing work related to the modeling, evaluating, and use of quality of context (QoC).

The second part of this work presents our proposition, which is composed of three parts: the proposed family of Context-Based Access Control models ($CxtBAC$); the definition of a Quality and Privacy-Aware Context Management Framework ($CxtMF$); the integration of these propositions in an access control infrastructure that implements the $CxtBAC$ built on the $CxtMF$, which was applied to mobile multimedia applications for validating our work. In the validation step, we used one approach that explores the expressive power of Web Ontology Language (OWL) [OWL 2009] for describing context-based access control policies. This approach uses context information directly described by OWL documents for enforcing context-based access control policies.

This document is organized into 8 (eight) Chapters, including this introduction, as described below:

- **Chapter 2:** it introduces the research area of access control. We present the traditional access control approaches, emphasizing the RBAC extended models;
- **Chapter 3:** it presents the existing access control approaches for pervasive environments, which are classified into Context-Aware (CAAC) and Context-Based Access Control (CBAC) solutions;
- **Chapter 4:** this Chapter presents concepts related to the quality of context information (QoC) and the existing QoC modeling and evaluating approaches;
- **Chapter 5:** it presents briefly the summary of existing works, and the research open issues that guided our propositions;
- **Chapter 6:** it describes the proposed family of context-based access control models (CxtBAC). CxtBAC family is composed of 8 (eight) access control reference models, which can be used as basis for implementing context-based access control systems;
- **Chapter 7:** this Chapter describes the proposed quality and privacy-aware context management framework. It is in charge of capturing, managing, and providing QoC-enriched context information to context-based applications and services, such as an access control system implementing $CxtBAC$;
- **Chapter 8:** this Chapter presents the instantiation of the $CxtBAC$ (Social-Aware CxtBAC model) integrated with the $CxtFM$ for protecting personal

multimedia resources. This instance of $CxtBAC$ was built on the proposed context management framework ($CxtMF$). Also, an application (PPlog) was developed, showing the use of this access control infrastructure;

- **Chapter 9:** This Chapter concludes the thesis by presenting the contributions of our work, as well as exposing the future work.

Chapter 2

Traditional Access Control Solutions

Résumé: *Ce chapitre décrit l'état de l'art en matière des solutions de contrôle d'accès en insistant sur les mécanismes traditionnels, tel que les modèles Discretionary Access Control (DAC), Mandatory Access Control (MAC), et Role-Based Access Control (RBAC). Aussi, nous présentons l'état de l'art liés aux extensions spatio-temporelles des rôles des utilisateurs (modèle RBAC) et aux aspects de généralisation du concept de rôle. L'idée générale de ce chapitre et de décrire les modèles de contrôle d'accès qui sont utilisés comme point de départ pour la définition des nouvelles propositions des mécanismes de contrôle d'accès. Également, les avantages et désavantages liés à chaque modèle sont discutés en détail dans ce chapitre.*

Contents

2.1 Introduction

By definition, access control is the process of mediating every request to resources and data maintained by a system, determining whether the request should be granted or denied [Samarati 2001]. In formal terms, *objects* represent the resources that are being protected by the system, *subjects* represent, for example, users or processes performing actions on an object, and *operations* represent all the actions that the subjects can perform on the *objects*. Security-sensitive environments should protect their resources against unauthorized use by enforcing access control mechanisms driven by access control policies.

Traditional access control systems are generally classified as *Discretionary Access Control (DAC)* [TCSEC 1985] or *Non-Discretionary Access Control (NDAC)*. In a DAC-based access control approach, the object owner or anyone else who is authorized to control the object's access specifies who have access to the object by defining access control policies. For instance, Access Control Matrix [Lampson 1974], Access Control Lists [Samarati 2001], and Capability-Based Access Control [Levy 1984] are well known DAC solutions.

All access control other than DAC are categorized as NDAC. In NDAC-based access control approaches, policies are rules that are not specified at the discretion of users. In this group, stand out the Mandatory Access Control (MAC) [TCSEC 1985], Role-Based Access Control (RBAC) [Ferraiolo 1992, Sandhu 1996], and the Attribute-based Access Control (ABAC) [Priebe 2004], which are largely implemented by conventional computer systems for protecting digital resources.

In fact, there are other types of classification for access control approaches, such as based on the type of information used for authenticating users (e.g., identity-based, group-based, role-based access controls), the content of protected objects (content-based access control), the trust relationship between resource owner and resource requestor (trust-based access control), the user's social relationships (social-based access control), and context-centric solutions.

We classify as traditional models the existing solution for controlling access of conventional resources, such as documents and services of corporate or administrative organizations. Normally, the identity of users is used to verify if they are allowed or not to access a required resource. We present in the following the most important and largely implemented access control models by current operating systems and data base management systems. These solutions are based on DAC, MAC, and RBAC models. Then, we focus on existing RBAC-Extended models, describing the proposed improvements in order to enforce dynamically access control policies.

2.2 Discretionary Access Control (DAC)

DAC is a type of access control defined by the *Trusted Computer System Evaluation Criteria - TCSEC*
[TCSEC 1985] *"as a means of restricting access to objects based on the identity of subjects and/or groups to which they belong"*. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission on to any other subject.

Discretionary term is commonly used by access control solutions that assume that every object has an owner that controls the permission to access her/his object. However, the TCSEC [TCSEC 1985] definition does not describe anything about resource owners. Technically, an access control system does not have to support the concept of owner to meet the TCSEC definition of DAC.

Discretionary access control is commonly defined in opposition to Mandatory Access Control [TCSEC 1985], sometimes termed *non-discretionary access control*. Thus, an access control system is *discretionary* or *purely discretionary* as a way of attesting that the system lacks mandatory access control. However, access control systems can implement both MAC and DAC simultaneously, where DAC refers to the ability that subjects have to transfer permission among each other, and MAC refers to the imposed constraints upon the first.

Therefore, a purely DAC is an user-centric access control approach that prevent illegitimate access to resources, offering users all the rights about the objects they create. Moreover, users can grant the rights they have to others (delegation) and they can remove the granted rights. In the following we present some existing DAC-based solutions.

2.2.1 Access Control Matrix

Lampson [Lampson 1974] proposed the use of access control matrix for controlling access rights in a DAC-based system. Graham *et al.* [Graham 1972] refined the Lampson's proposition, which was posteriorly formalized by Harrison *et al.* [Harrison 1976]. The formalization proposed by Harrison *et al.* [Harrison 1976]

Table 2.1: Example of Access Control Matrix in a Unix-like operating system

User	**/home/user1**	**/home/user2**	**/tmp**
user1	read, write, execute	-	read, write, execute
user2	-	read, write, execute	read, write, execute

identified six primitive operations that can have an effect on the authorization state of an access control matrix: adding and removing a subject, adding and removing an object, and granting and removing a privilege.

The original model is called access matrix since the authorization state is represented as a matrix. An access control matrix consists of rows representing subjects and columns representing objects. Thus, the cells in the matrix define the operations that the subject can perform on the given object. A first step in the development of an access control matrix is the identification of the objects to be protected, the subjects that execute activities and request access to objects, and the actions that can be executed on the objects.

The state of a system implementing an access control matrix is defined by a triple (S, O, A), where S is the set of *subjects* which can exercise privileges, O is the set of *objects* on which privileges can be exercised, and A is the access control matrix, where rows correspond to subjects, columns correspond to objects, and an entry $A[s, o]$ reports the privileges of s on o.

Access control matrix (A) is formally defined by Equation 2.1, where $A[s, o] \subseteq A$ represents the access operations that the subject, $s \in S$, can perform on an object $o \in O$. A is the set of all the access operations that a subject can perform on an object.

$$A = A[s, o] s \in S, o \in O, A[s, o] \subseteq A \tag{2.1}$$

Table 2.1 describes a example of access control matrix implemented by a Unix-like operating system. *User1* has read, write, and execute access permission on */home/user1* and */tmp* directories, but he/she cannot access the */home/user2*. The administrator (root) has delegated to *user1* and *user2* rights to change access permission on their home directory.

Systems that implement a DAC-based solution must have a *Reference Monitor (RM)* [Anderson 1972] in charge of checking access request validity, granting/denying access on protected resources. Reference Monitor (RM) concept is an effective tool for describing the abstract requirements of secure system design and implementation. A Reference Monitor should have the following properties:

- It must be always invoked, i.e., every access is mediated;
- It must be tamper proof. It must be impossible for a intruder to attack

the access mediation mechanism such that the required access checks are not performed and authorizations not enforced;

- It must be small enough to be subject to analysis and test, the completeness of which can be assured.

Together with hardware, firmware and other software, the reference monitor in a computer system forms the *trusted computing base (TCB)*. The TCB is defined as the set of components that, if working correctly, will be enough to enforce the security policy in the system regardless of the behaviour of other components.

While the access control matrix is a good theoretical tool, it is rarely used as such in actual implementations. The matrix is likely to be sparse in systems with more than one user where objects accessed by the users of the system rarely overlap. Storing the matrix as a two-dimensional array is therefore a waste of memory space. For example, in a typical Unix-like operating system users have their own files in their own home directories (see the Table 2.1), and the only files that are commonly shared between users are the executables in the system. Therefore, access control implementations typically use either *Authorization Table*, *Access Control Lists* or *Capabilities* to represent DAC-based access control policy.

2.2.2 Authorization Table

If a DAC-based system is implemented by using authorization table, then the non empty entries of an access control matrix are reported in that table [Samarati 2001]. An authorization table is composed by three columns, corresponding to *subjects*, *actions*, and *objects*, respectively. Each tuple of this table corresponds to an access authorization. The authorization table approach is generally used by Data Base Management Systems (DBMS). In this case, authorization tables are stored and represented as relational tables of the database.

Table 2.2 shows the same example of DAC policies illustrated in Table 2.1, but represented using authorization tables. The main advantage of using authorization table instead of access control matrix is to reduce the waste of memory.

2.2.3 Access Control List (ACL)

By taking a column centric view of the access control matrix approach, each column of the matrix is translated to an access control list (ACL) [Samarati 2001]. ACL are typically stored with the object that the column represents. The ACL contains entries for each subject defining the operations that the subject can execute on the given object. Figure 2.1 illustrate the ACL create from the access control matrix presented in Table 2.1.

In an ACL-based solution it is often difficult to see which objects are accessible

Table 2.2: Example of Authorization Table

User	Authorization	Object
user1	own	/home/user1
user1	read	/home/user1
user1	write	/home/user1
user1	execute	/home/user1
user1	read	/tmp
user1	write	/tmp
user1	execute	/tmp
user2	own	/home/user2
user2	read	/home/user2
user2	write	/home/user2
user2	execute	/home/user2
user2	read	/tmp
user2	write	/tmp
user2	execute	/tmp

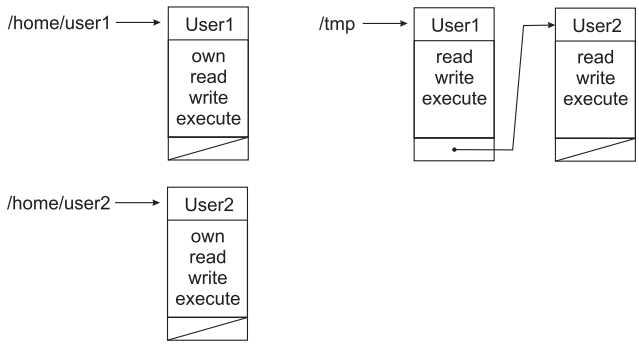


Figure 2.1: Example of Access Control List.

to a given subject. This is rarely a problem, since it is usually more interesting to get the list of subjects that are allowed to access a given object. If it is necessary to find all objects accessible to a given subject, it is possible by checking each protected object in the system. In the ACL represented in the Figure 2.1, for example, it is necessary to check all ACL of protected objects (/home/user1, /home/user2, /tmp) in order to find the objects accessible to a user (User1).

For practical reasons access control lists are often truncated when they are implemented by operating systems, restricting the assignment of authorizations to a limited number (usually one or two) of named groups of users, while individual authorizations are not allowed. For instance, in most Unix-like operating systems the ACL associated with a file contain only three subjects: user (u), group (g), and others (o). Authorization for each file can be specified for the file's owner (u), for the group to which the file belongs (g), and for *the rest of the world* (o), meaning all the remaining users.

In fact, there exists two motivations for simplifying ACL in operating systems: 1) in most files in a Unix-like operating system are accessed only by a few subjects or alternatively by a group of subjects, resulting in very sparse ACL; 2) complete ACL would need to be updated whenever a new subject is added to a system resulting in the management software having to go through all existing ACL of all the files in the system.

2.2.4 Capabilities

A DAC-based system can alternatively be implemented with a row centric view of the access control matrix, where each row of a matrix is translated to a *capability*. Each user has associated a list created from the correspondent row of access control matrix (capability), indicating for each object her access permission. In a system supporting capabilities, it is sufficient for a subject to present the appropriate capability to gain access to an object. Figure 2.2 illustrates an example of capabilities.

Capability-based access control systems share capabilities with users according to the *principle of least privilege*[1], and to the operating system infrastructure necessary to make such transactions efficient and secure. In theory, a system with capabilities removes the need for any access control list or similar mechanism by giving all entities all and only the capabilities they will actually need.

Because capabilities are often stored with the subject and the possession of a capability implies authority, it is important that a capability implementation protects the integrity of the capabilities. More specifically, capabilities must be *unforgeable* and *non-transferable*.

[1] Principle of least privilege (minimal privilege or just least privilege) requires that in a particular abstraction layer of a computing environment, every module (such as a process, a user or a program on the basis of the layer we are considering) must be able to access only such information and resources that are necessary to its legitimate purpose

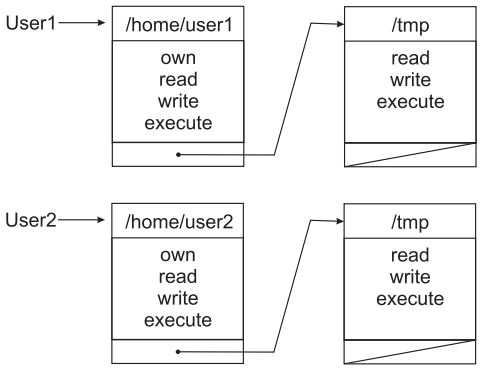


Figure 2.2: Example of Capabilities.

Capability represents an advantage in distributed systems since it permits to avoid repeated authentication of a subject: a user can be authenticated at a host to acquire the appropriate capabilities, then he/she presents them to obtain access to the various servers of the system. However, capabilities are vulnerable to forgery, i.e., the capabilities can be copied and reused by an unauthorized third party. Another problem in the use of capability is the enforcement of revocation, meaning invalidation of capabilities that have been released. A number of capability-based computer systems were developed in the 1970s, such as the Cambridge CAP computer [Wilkes 1979].

2.2.5 Advantages and disadvantages

The flexibility and simplicity of DAC-based solutions are the key reasons why DAC is widely known and used by most existing operating systems. However, DAC solutions has some limitations. For example, in a multi-domain setting the centralized nature of DAC solutions introduces some problems, such as difficult to deploy and delegate access permission. We describe below the main disadvantages of using DAC for protecting resources:

- **Global policy:** DAC let users decide the access control policies on their data, regardless of whether those policies are consistent with the global policies. Therefore, if there is a global policy, DAC has trouble to ensure consistency;
- **Information flow:** information can be copied from one object to another, so access to a copy is possible even if the owner of the original does not provide access to the original copy. This has been a major concern for military applications;
- **Malicious software:** DAC policies can be easily changed by the owner, so a malicious program (e.g., an untrustworthy software) running by the owner can change DAC policies on behalf of the owner;

- **Flawed software:** similarly to the previous item, flawed software can be instructed by attackers to change its DAC policies.

In operating systems that implement DAC, processes are able to run programs (e.g., Trojan Horse) which cannot be trusted for the operations they execute. For this reason, restrictions should be enforced on the operations that processes themselves can execute. Mandatory access control policies provide a way to enforce information flow control through the use of *labels*, as discussed in Section 2.3.

2.3 Mandatory Access Control (MAC)

Unlike DAC-based approaches, mandatory solutions enforce access control on the basis of regulations mandated by a *central authority*. MAC-based systems have their roots in the military and intelligence communities, which have based their access control on hierarchical classification levels. MAC is defined by the *Trusted Computer System Evaluation Criteria - TCSEC* [TCSEC 1985] as *"a means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (i.e., clearance) of subjects to access information of such sensitivity"*.

MAC-based systems can only protect the confidentiality[2] or integrity[3] of data, but never both simultaneously. Moreover, the subject concept used by MAC-based system has a different meaning that the considered in DAC-based solutions. While subjects in DAC-based solutions typically correspond to users or groups, in MAC-based systems subjects refer to the processes (i.e., programs in execution) operating on behalf of users. This distinction allows the MAC-based systems to control the indirect accesses caused by the execution of processes, which is the main security problem of DAC-based solutions.

The most common MAC solutions is the multilevel security (MLS) [Bell 1973], the Ken Biba model [Biba 1977], and the Chinese Wall model [Brewer 1989].

2.3.0.1 Multilevel security (MLS)

Multilevel security (MLS) [Bell 1973, Bell 1974, Bell 1976, LaPadula 1973] is based on the classifications of subjects and objects in the system. Objects are passive entities storing information and subjects are active entities that request access to the objects. The MLS model concentrates on the confidentiality of data. It prevents information from *flowing downwards* in the classification system, i.e., from a higher level of classification to a lower one.

[2] Confidentiality has been defined by the International Organization for Standardization (ISO) in ISO-17799 "as ensuring that information is accessible only to those authorized to have access".

[3] Integrity means that data cannot be modified without authorization.

The access class is one element of a partially ordered set[4] of classes. The partial order is defined by a *dominance* relationship, which is denoted by $\geq$.

Most commonly the classification of access (access class) is defined as consisting of two components: a *security level* and a *set of categories*. The security level is an element of a hierarchically ordered set, such as Top Secret (TS), Secret (S), Confidential (C), and Unclassified (U), where $TS > S > C > U$. The set of categories is a subset of an unordered set, whose elements reflect functional or competence areas, such as Financial, Administration, and Research.

A subject in an MLS system is allowed to access an object only if its access class is greater or equal to the access class of the object. For example, a user with access class Secret S is able to read and write Secret (S), Confidential (C), and Unclassified (U) objects, but not Top Secret (TP) objects.

Definition 1. *The dominance relationship $\geq$ is then defined as follows: an access class c1 dominates $\geq$ an access class c_2 iff the security level of c_1 is greater than or equal to that of c_2, and the categories of c_1 include those of c_2.*

- *Let ℓ be the ordered set of security class and C a set of categories;*
- ***Access Class (AC)*** $= \ell \times \wp^5(C)$*, and* $\forall c_1 = (L_1, C_1), c_2 = (L_2, C_2) : c_1 \geq c_2 \Leftrightarrow L_1 \geq L_2 \wedge C_1 \supseteq C_2$*;*
- *Two classes c_1 and c_2 such that neither $c1 \geq c2$ nor $c2 \geq c1$ holds are classified as incomparable;*
- *AC satisfies the following properties: reflexivity, transitivity, antisymmetry, existence of a least upper bound and a greater lower bound.*

Mathematically, the security level access may also be expressed in terms of a security lattice [Denning 1976] (a partial order set) where each object and subject have a greater lower bound (meet) and least upper bound (join) of access rights. In fact, a security lattice is formed from the definition of access classes together with the dominance relationship between them. Figure 2.3 illustrates the security lattice obtained considering security levels S and C, with $S > C$ and the set of categories *{Financial, Administration}*.

The security level of the access class associated with a user (*clearance*) reflects the user's trustworthiness to not disclose sensitive information to other users not cleared to see it. Categories define the area of competence of users and data in order to provide finer grained security classifications of subjects and objects than classifications provided by security levels alone. They are the basis for enforcing *need-to-know*

[4] Partially ordered set (poset) consists of a set together with a binary relation that indicates that, for certain pairs of elements in the set, one of the elements precedes the other.

[5] $\wp$(S) is the power set (or powerset) of S, which is the set of all subsets of S, including the empty set and S itself.

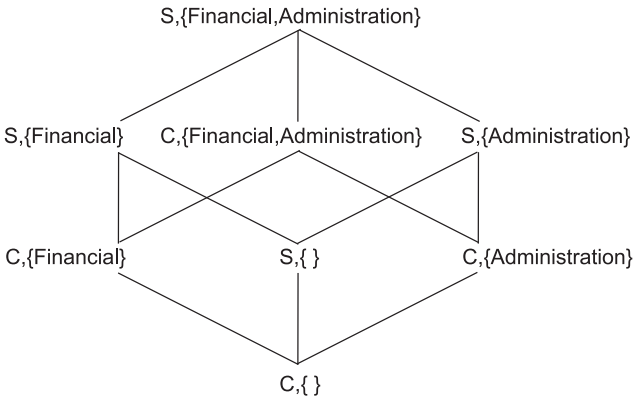


Figure 2.3: Example of lattice security.

restrictions, confining subjects to access information they actually need to know to perform their job. For instance, in the security lattice illustrated in Figure 2.3, for a user to get access on an object classified as confidential (C) and the category *Administration*, he/she needs to have at least the clearance $(C, \{Administration\})$.

Two principles formulated by Bell *et al.* [Bell 1973] must be satisfied to protect the confidentiality of objects:

- **No-read-up (simple security property):** a subject is allowed a read access to an object only if the access class of the subject dominates the access class of the object, i.e., a subject S is allowed to read object O only if $class(O) \leq class(S)$;
- **No-write-down (*-property):** a subject is allowed a write access to an object only if the access class of the subject is dominated by the access class of the object, i.e., a subject S is allowed to write object O only if $class(S) \leq class(O)$.

These two principles prevent the information flow accessible by users classified in a high level security class to be accessible by users classified at lower levels (i.e., users not cleared for it).

2.3.1 Ken Biba model

Ken Biba has proposed a MAC-based model [Biba 1977] from the principles of the Bell and LaPadula model [Bell 1973]. This model concentrates solely on data integrity, ignoring confidentiality considerations. When protecting the confidentiality of information, it is important to prevent that information flowing from high classification levels to lower classification levels. However, in a system that requires the integrity of information, it is necessary to prevent information flowing upwards, i.e.,

from lower classification levels to a higher one. In this case, subjects should always *read up* and *write down*, i.e., subjects should read data from a higher classification level and write data to lower classification levels. These goals are contrary to the goals of a confidentiality protecting system.

Therefore, the Ken Biba model controls the flow of information and prevents subjects to modify information they do not have the write access. Like for confidentiality in the Bell and LaPadula model, each subject and object in the system is assigned to an integrity classification. The classification and the dominance relationship between the access classes are defined as described in the section 2.3.0.1. For instance, integrity levels could be defined as following: Crucial (C), Important (I), and Unknown (U).

In fact, the integrity level associated with a user reflects the user's trustworthiness for inserting, modifying, or deleting information. The integrity level associated with an object reflects both the degree of trust that can be placed on the information stored in the object and the potential damage that could result from unauthorized modifications of the information. Like in the Bell and LaPadula model, categories can be used to define the area of competence of users and data.

In the Ken Biba model, the access control is enforced according to the following two principles:

- **No-read-down:** a subject is allowed a read access to an object only if the access class of the object dominates the access class of the subject, i.e., i.e., a subject S is allowed to read object O only if $class(O) \geq class(S)$;
- **No-write-up:** a subject is allowed a write access to an object only if the access class of the subject dominates the access class of the object, i.e., a subject S is allowed to write object O only if $class(S) \geq class(O)$.

By satisfying these principles, the integrity of information flowing from low objects to higher is assured. A major limitation of the Ken Biba model is that they only capture integrity compromises due to improper information flows. If both confidentiality and integrity have to be controlled, objects and subjects have to be assigned two access classes, one for confidentiality control and one for integrity control.

2.3.2 Chinese Wall model

Brewer and Nash have proposed the Chinese Wall model [Brewer 1989]. This model has its roots in the investment banking industry where it is important to internally prevent conflicts of interest. The motivation for this work was to avoid that sensitive information concerning a company be disclosed to competitor companies through the work of financial consultants. Therefore, the main goal of the Chinese Wall model is to prevent information flows which cause conflict of interest for individual users, i.e., the corporations.

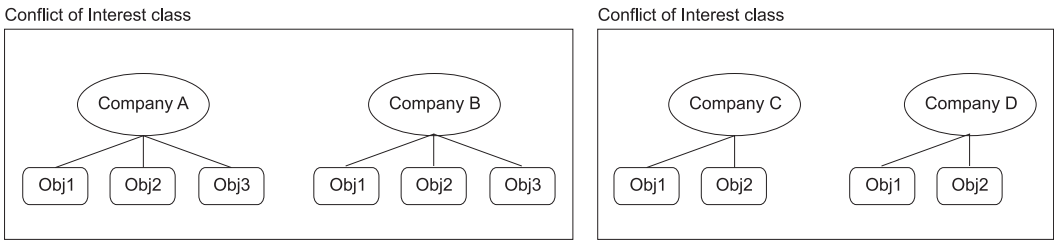


Figure 2.4: Example of data organization.

However, unlike in the Bell and LaPadula model, access to data is not constrained by the data classifications but by what data the subjects have already accessed. The model is based on a hierarchical organization of data objects and uses two access rules (read and write rules), as follows:

- **Information:** objects are items of information, each concerning a single corporation. Example: files;
- **DataSet:** company datasets define groups of objects that refer to a same corporation;
- **Conflict of interest (CoI) classes:** it defines company datasets that refer to competing corporations;
- **Read Rule (simple security rule):** a subject S can read an object O if:
  - O is in the same Dataset as an object already accessed by S, **OR**
  - O belongs to a CoI from which S has not yet accessed any information (i.e., a Dataset of an entirely different CoI).
- **Write Rule (*-property):** a subject S can write an object O if:
  - S can read O according to the Read Rule, **AND**
  - No object has been read by S which is in a different company dataset to the one on which write is performed, **AND**
  - The O contains unsanitized[6] information. Therefore, the flow of information is confined to its own company dataset.

Chinese Wall policy controls users and not processes. This is because a user could be able to acquire information about organizations that are in conflict of interest simply by running two different processes. This model is a combination of free choice and mandatory control, which initially a subject is free to access any object it wishes. Once the initial choice is made, a *Chinese Wall* is created for that

[6] *Sanitization* is the process of removing sensitive information from a document or other medium, so that it may be distributed to a broader audience.

user around the dataset to which the object belongs. In order to improve the access control of the protected objects, a Chinese Wall can be combined with DAC policies.

Figure 2.4 illustrates an example of data organization of four different corporations, namely A,B,C, and D. The two conflict of interest classes define the conflicts between the corporations A and B, and between C and D. A user of corporation A cannot read objects from the corporation B, and vice versa. The same occurs with the users of the corporations C and D. If a user of the corporation A has read an object from the corporation C, then he/she can only write objects in that corporation (C), being unable to read and write in the corporation D.

Chinese Wall model still has some limitations. For instance, the strict enforcement of the properties may result in a too rigid access control solution, and the enforcement of policies requires keeping and querying the history of the accesses. Moreover, it is necessary to add the support for exceptions and sanitization of information.

2.3.3 Advantages and disadvantages

While formal MAC models enable reasoning about the security of the systems by assuring the confidentially and integrity of information flow, in many cases these models end up being too rigid for practical deployments. Operations that should be simple, e.g., object creation and deletion, become overly complex and require compromises.

In MAC-based solution, the main problem is the correct classification of subjects and objects, such that correct access rights are enforced. Another issue is the comprehensibility of the policy specification to the MAC mechanisms.

Note that DAC and MAC models are not mutually exclusive, i.e., these two types of access control models can be applied jointly for protecting resources in a system. In this case, for granting access to a user, it is necessary to satisfy these conditions: i) to satisfy the mandatory access control policy; ii) the existence of the necessary authorization for accessing it. In fact, the discretionary policy operates within the boundaries of the mandatory policy, restricting the set of accesses that would be allowed by MAC alone.

However, one challenging problem in managing large systems using MAC and/or DAC solutions, is the complexity of security administration. Whenever the number of subjects and objects is high, the number of authorizations can become extremely large, which complicates the administration tasks. Moreover, if the user population is highly dynamic, the number of grant and revoke operations to be performed can become very difficult to manage. End users often do not own the information for which they are allowed access.

Normally, in a professional environment the corporation is the actual owner of data objects. In this case, the access control is often based on employee func-

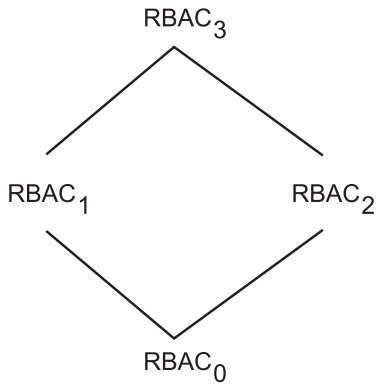


Figure 2.5: Relationship among RBAC96 models.

tions rather than data ownership. In order to simplify the administration tasks and to support function-based access control, RBAC [Ferraiolo 1992, Sandhu 1996, Ferraiolo 2003] has been proposed as an alternative approach to DAC and MAC-based solutions. Section 2.4 following we describe in detail this access control model.

2.4 Role-Based Access Control (RBAC)

Ferraiolo *et al.* [Ferraiolo 1992] proposed the RBAC model (a.k.a RBAC92 model), identifying the fundamental concepts related to the concept of roles. RBAC92 model was subsequently extended by Sandhu *et al.* [Sandhu 1996] in order to propose a RBAC conceptual framework that can be used as basis for implementing RBAC-based solutions, named RBAC96 model.

In the years that followed, RBAC model became the predominant model for advanced access control, mainly by reducing costs of deployment and maintenance. This motivated NIST[7] to call for a unified standard for RBAC in order to integrate the RBAC model published by Ferraiolo *et al.* [Ferraiolo 1992] with the RBAC framework introduced by Sandhu *et al.* [Sandhu 1996]. This proposal was published by Sandhu *et al.* [Sandhu 2000] and adopted as an ANSI[8]/INCITS[9] standard in 2004.

Sandhu *et al.* [Sandhu 1996] proposed a family of RBAC models (RBAC96 models, see Figure 2.5): $RBAC_0$ (the base model, a.k.a. RBAC core), $RBAC_1$ (it includes the $RBAC_0$ with the support to Role Hierarchy), $RBAC_2$ (it includes the $RBAC_0$ with the support to constraints), and $RBAC_3$ (it includes $RBAC_1$, $RBAC_2$, and $RBAC_0$ by transitivity). The NIST RBAC model [Sandhu 2000] there are four levels of increasing functional capabilities: i) Core RBAC, also named Flat RBAC; ii) hierarchical RBAC; and iii) constrained RBAC; iv) symmetric RBAC. These

[7] National Institute of Standards and Technology.
[8] American National Standards Institute: http://www.ansi.org/
[9] InterNational Committee for Information Technology Standards: http://www.incits.org/

levels are cumulative and each adds exactly one new requirement.

2.4.1 $RBAC_0$: RBAC Core model

The RBAC96 core model ($RBAC_0$) has four main elements: Users (U), Roles (R), Permission (P), and sessions (S). A user represents a human activity or an autonomous agent in a computer system, while a role is a job function or job title within the organization that represents authority and responsibility conferred on a member of the role. A permission is an approval of a particular mode of access to one or more objects in the system.

According to the NIST RBAC model [Sandhu 2000], permission is always positive and confers the ability to the holder of that permission to perform some action(s) in the system. The NIST model does not rule out the use of so-called *negative permission* which deny access. The nature of a permission depends directly on the implementation details of a system and the kind of system that it is (e.g., read and write permission on files in a file system, INSERT and DELETE operations on a table of a data base).

In the RBAC96 model [Sandhu 1996], each session is a mapping of one user to possibly many roles, i.e., a user establishes a session during which the user activates some subset of roles. The following definition formalizes the above discussion.

Definition 2. *The $RBAC_0$ Model is composed of the following components:*

- *U, R, P, and S (users, roles, permission, and sessions respectively);*
- *$PA \subseteq P \times R$, a many-to-many permission to role assignment relation;*
- *$UA \subseteq U \times R$, a many-to-many user to role assignment relation;*
- *user: $S \rightarrow U$, a function mapping each session s_i to the single user $user(s_i)$ that is constant for the session's lifetime;*
- *roles: $S \rightarrow 2^R$, a function mapping each session s_i to a set of roles $roles(s_i) \subseteq \{r|(user(s_i), r) \in UA\}$. The permission assigned to a user is the union set resulting from the sets of permission assigned to each role activated in the session to that user.*

Figure 2.6 illustrates the $RBAC_0$ model. The basic concept of this model is that users are assigned to roles (user assignment), permission is assigned to roles (permission assignment), and users acquire permission by being members of roles. In a RBAC model, user-role and role-permission assignment can be many-to-many, which is represented by a double-headed arrow. The NIST RBAC model named this RBAC96 model of Flat RBAC. The main difference between them is that the concept of session is not explicitly a part of flat RBAC. In fact, a session corresponds

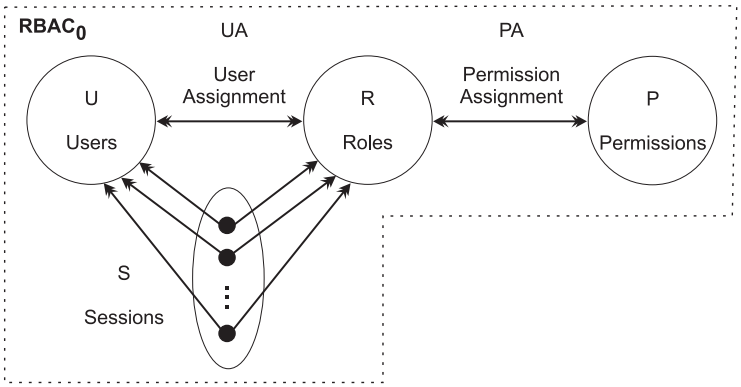


Figure 2.6: $RBAC_0$: The core model [Sandhu 1996].

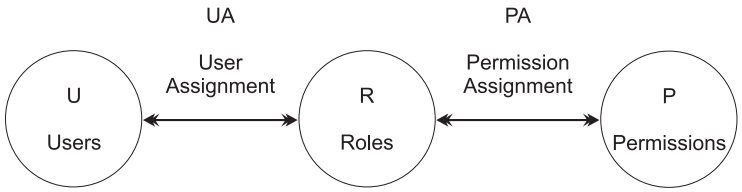


Figure 2.7: NIST Flat RBAC: The core model [Sandhu 2000].

to a particular occasion when a user signs on the system to carry out some activity, which can vary widely from system to system (e.g., some systems activates all user's roles and others the user is given a choice to activate and deactivate roles in a given session at the user's discretion). Moreover, Flat RBAC requires support for user-role review (i.e., to determine which roles a given user belongs to and which users a given role is assigned to) and role-permission review (i.e., to determine which permission is assigned to a role and which roles a permission is assigned to). Figure 2.7 presents the NIST Flat RBAC model, where sessions are not explicitly present in that model.

2.4.2 $RBAC_1$: Hierarchies

$RBAC_1$ of RBAC96 model introduces Role Hierarchies (RH) using as basis the $RBAC_0$. Figure 2.8 illustrates the $RBAC_1$. RH is a natural means for structuring roles to reflect the hierarchical organization of a company. For example, in a technology company we could have the following roles: project member, test engineer, programmers, and project supervisor. Figure 2.10 presents these roles structured following the company organization of authority and responsibility. By convention, more powerful roles (i.e., senior roles) are shown toward the top and less powerful roles (i.e., junior roles) toward the bottom. In the example illustrated by Figure 2.10, the project supervisor role inherits from both test engineer and programmer roles. The formal definition of $RBAC_1$ is given below.

Definition 3. *$RBAC_1$ model has the following components:*

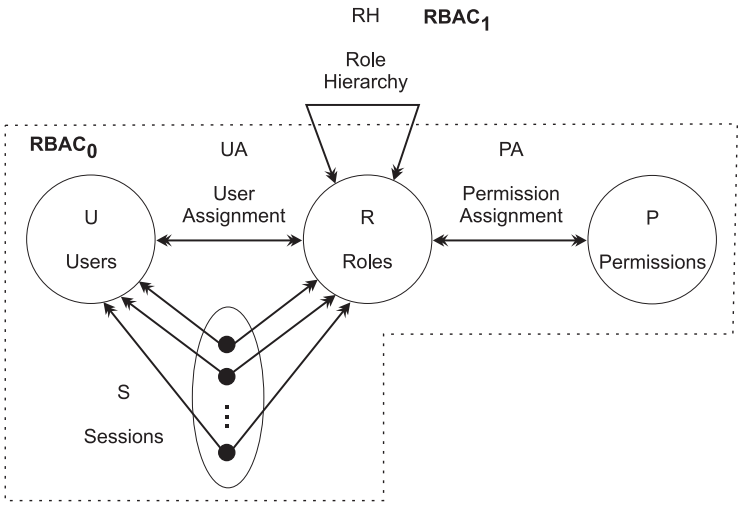


Figure 2.8: $RBAC_1$: Role Hierarchy (RH) [Sandhu 1996].

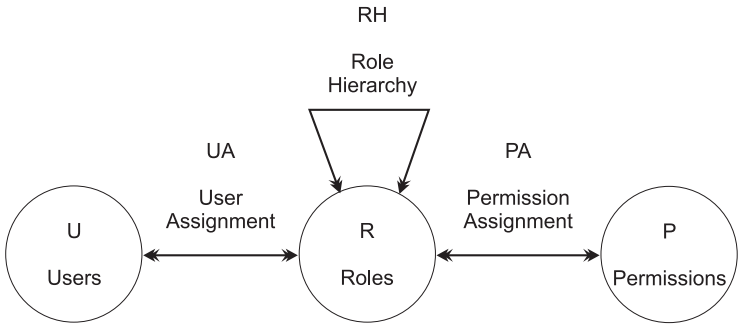


Figure 2.9: NIST Hierarchical RBAC [Sandhu 2000].

- *U, R, P, S, PA, UA, are unchanged from* $RBAC_0$*;*
- $RH \subseteq R \times R$*, is a partial order on R called the role hierarchy or role dominance relation* ($\geq$)*;*
- $Roles : S \rightarrow 2^R$ *requires* $roles(s_i) \subseteq \{r | (\exists r_1 \geq r_2)\,[(user(s_i), r_1] \in UA\}$*, and session* s_i *has the permission resulting from the union set of permission assigned to the current role and each dominated role in the HR.*

$RBAC_1$ model introduces also the concept of *private role*, which blocks upward inheritance of certain permission. In the NIST RBAC model, the $RBAC_1$ is named Hierarchical RBAC (see Figure 2.9). This model defines two types of hierarchies: *General Hierarchical RBAC* and *Limited Hierarchical RBAC*. The first supports an arbitrary partial order to serve as the role hierarchy, while the second may impose restrictions on the structure of the role hierarchy, such as to be represented by trees or inverted trees. Moreover, hierarchical NIST RBAC model presents two distinct interpretations of a role hierarchy: inheritance hierarchy (members of a senior role

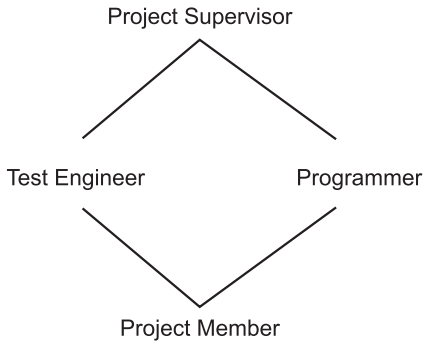


Figure 2.10: Example of Role Hierarchy.

in the hierarchy are regarded as inheriting permission from juniors), and activation hierarchy (activation of a senior role does not automatically activate permission of junior roles).

2.4.3 $RBAC_2$: Constraints

Constraints are an important aspect of RBAC96 model used to define higher-level organizational policy, such as mutually disjoint roles. If the management of RBAC is decentralized, constraints become a mechanism by which senior security officers can restrict the ability of users who can exercise administrative privileges.

Constraints can be applied to UA and PA relations, to sessions, user, and role functions associated with a session. When applied to these relations and functions, constraint returns a value of acceptable or not acceptable. Constraints are formally defined by the following definition:

Definition 4. *$RBAC_2$ is unchanged from $RBAC_0$ except for requiring that there be a collection of constraints determining whether or not values of various components of $RBAC_0$ are acceptable. Only acceptable values will be permitted.*

Figure 2.11 illustrates the $RBAC_2$ model. The most frequently mentioned constraint is the *mutually exclusive* roles. A user can be assigned to at most one role in a mutually exclusive set. This type of constraint supports separation of duties. The mutual exclusion constraint on permission assignment (PA) is a useful means of limiting the distribution of powerful permission. Moreover, it is possible to define constraint on user assignment (UA), such as *cardinality constraints* (e.g., the maximum number of members in a role). A role hierarchy can be considered as a constraint where a permission assigned to a junior role must also be assigned to all senior roles. However, it is preferable to support hierarchies directly rather than indirectly by means of redundant assignment.

Unlike RBAC96 model, NIST RBAC model adds constraints to the hierarchical RBAC model (the equivalent to the $RBAC_1$ of RBAC96 model). In this model,

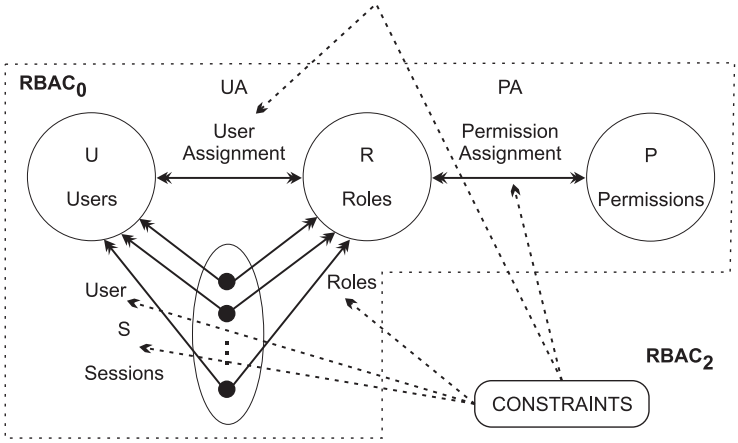


Figure 2.11: $RBAC_2$: Constraints [Sandhu 1996].

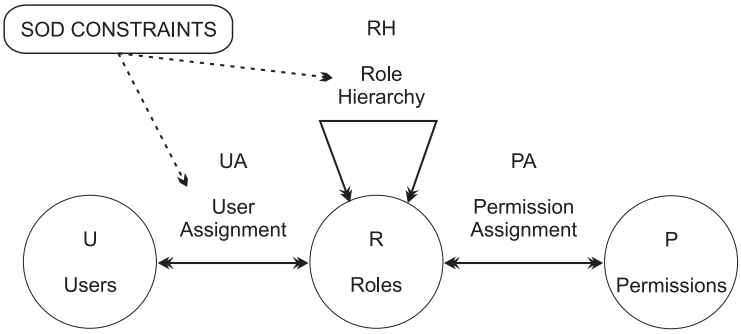


Figure 2.12: NIST Constrained RBAC - Static SoD [Sandhu 2000].

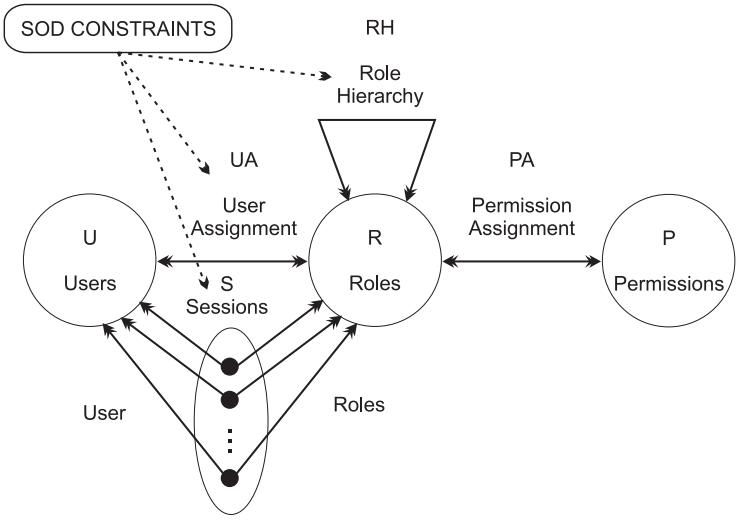


Figure 2.13: NIST Constrained RBAC - Dynamic SoD [Sandhu 2000].

constraints may be associated with the user-role assignment (static separation of duty - SSD), or with the activation of roles within user sessions (dynamic separation of duty - DSP). The separation of duty (SoD)[10] is used to enforce conflict of interest policies that the organization may employ to prevent its users from exceeding authority when accessing the protected resources. The motivation to implement it is to ensure that fraud and major errors cannot occur without the involvement of multiple users performing different tasks in the organization. To support this functionality, it is necessary to apply before the principle of least privilege (see the definition in the section 2.2).

The NIST model supports both static and dynamic SoD, but leaves open which of these should be implemented. Static SoD enforces constraints on the user-role assignments. Such constraints are inherited within a role hierarchy. For instance, if a user is authorized for the Cashier role, then that user is unauthorized for the Cashier Supervisor role. See Figure 2.12 these two types of constraints. Dynamic SoD addresses potential conflict-of-interest issues at the time a user-role assignment is authorized. For example, a user may be authorized for both Cashier and Cashier Supervisor roles. However, if a user acting in the Cashier role attempted to switch to the Cashier Supervisor role, then the RBAC system would require the user shutdown her/his current user-role assignment before assuming the Cashier Supervisor role. Figure 2.13 illustrates the NIST constrained RBAC that supports dynamic SoD.

2.4.4 $RBAC_3$: Consolidated Model

$RBAC_3$ combines $RBAC_1$ and $RBAC_2$ (and the $RBAC_0$ by transitivity) to provide simultaneously role hierarchies and constraints. As a result, constraints can be applied to the role hierarchy itself, as indicated by the dashed arrow to RH in Figure 2.14.

$RBAC_3$ model is named symmetric RBAC in the NIST RBAC model. From the two NIST constrained model (static and dynamic SoD) described in the previous section, authors defined two symmetric RBAC models by extending the support of constraints on permission-role assignments. Figure 2.15 and Figure 2.16 illustrate these two models, respectively.

2.4.5 Management of RBAC models

Management of RBAC-based systems consist of performing the following set of activities: i) defining roles and role hierarchy; ii) granting and revoking membership to the set of specified roles within the system; iii) defining the permission-role assignments applying the principle of least privilege; iv) defining constraints; v) reviewing

[10] Separation of duty requires that for a particular set of transactions, no single individual is allowed to execute all transactions within the set. Example: in a corporation, no single individual should be capable of executing both a payment and to authorize it.

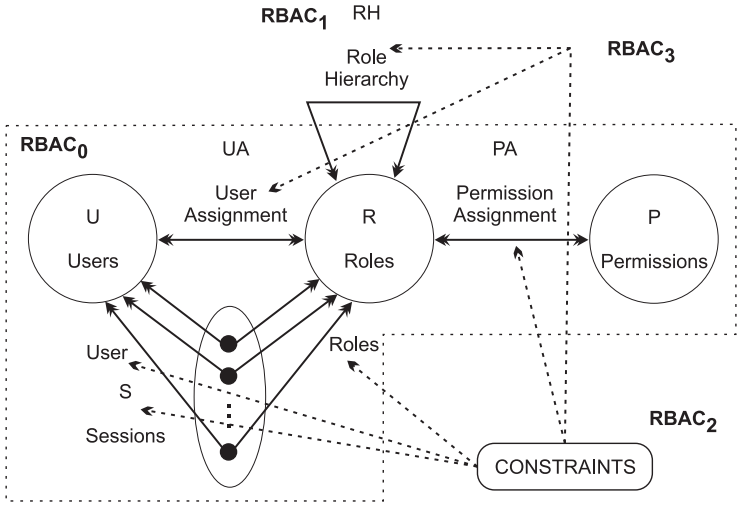


Figure 2.14: $RBAC_3$: The consolidated RBAC96 model [Sandhu 1996].

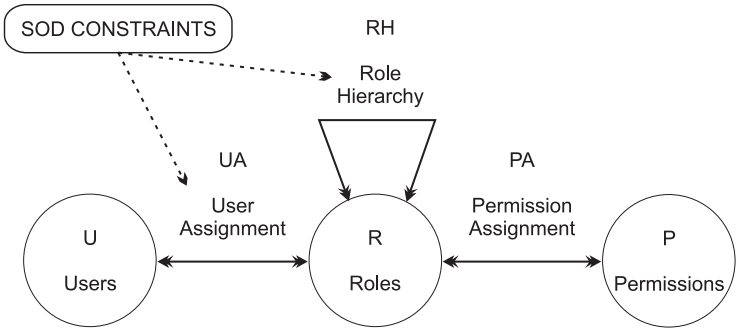


Figure 2.15: NIST Symmetric RBAC - Static SoD [Sandhu 2000].

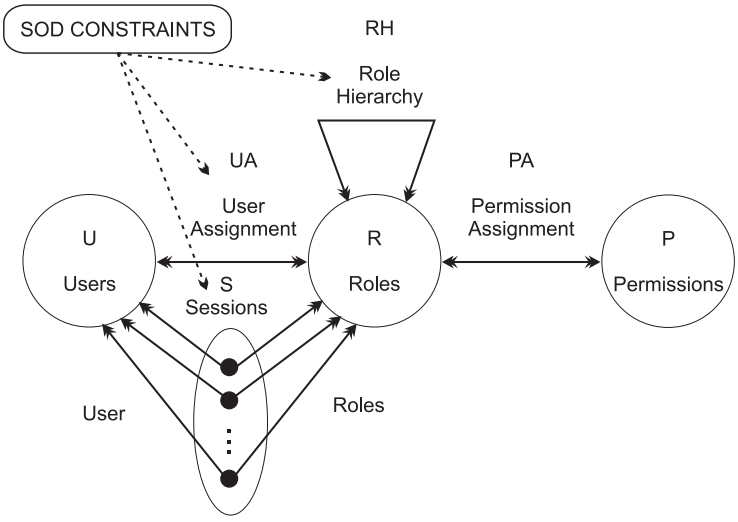


Figure 2.16: NIST Symmetric RBAC - Dynamic SoD [Sandhu 2000].

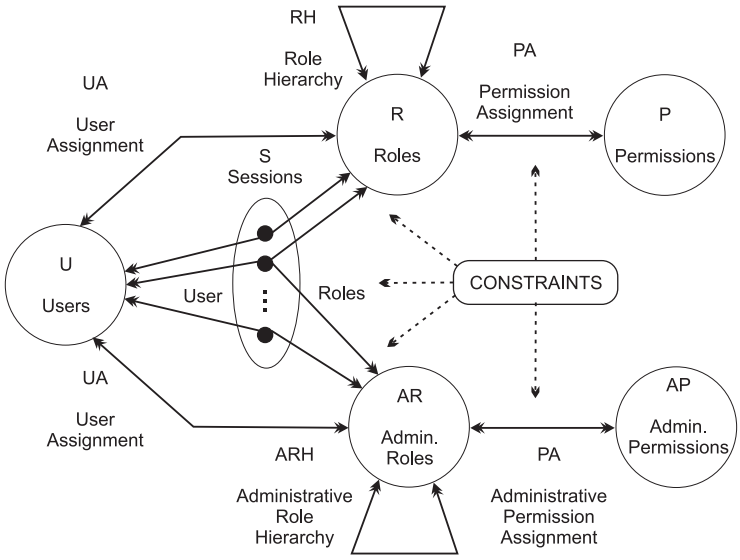


Figure 2.17: RBAC96 Administrative Model [Sandhu 1996].

constraints, roles and the user-role and permission-role assignments during the entire life cycle of the system.

The RBAC model can be used for managing RBAC itself [Sandhu 1996]. Sandhu *et al.* proposed a management model for RBAC illustrated in Figure 2.17. The top half of the model is similar to the $RBAC_3$ and the constraints are applied to all components. The bottom half of Figure 2.17 is a mirror image of the top half for administrative roles and administrative permission. Administrative roles AR and administrative permission AP are disjoint from the regular roles R and permission P, respectively. Thus, the administrative RBAC (ARBAC) can be used to manage the RBAC.

2.4.6 Advantages and Disadvantages

RBAC can be configured to support a wide variety of access control policies, including traditional discretionary access control (DAC) and mandatory access control (MAC). In the RBAC model, the user is assigned to a subset of roles when he/she starts a session. During a session, although roles can be activated or deactivated based on constraints such as role conflict or prerequisite roles. In RBAC-based systems user's access privileges are not changed based on context information but in the roles that she/he performs in an organization.

Therefore, the user and permission assignments are statics and do not take into account any contextual information from the environment when assigning permission, such as time and the location of users. In fact, traditional RBAC models cannot be used to capture security-relevant informations from the environment, which could have an impact on access decisions.

For instance, in a health care system the access to medical records of patients could be dynamically controlled depending on the location of users (e.g., doctor, nurse) at request time. Normally, a doctor that is not localized in the hospital should not have access to medical records of patients. Moreover, this access is allowed only during their work shift. Section 2.5 presents some existing approaches that extend RBAC model in order to dynamically make user-role and role-permission assignments.

2.5 Extended RBAC Models

In this section, we present some existing access control models and mechanisms that extend the RBAC model in order to dynamically enforce RBAC policies. The proposed extensions could be based on one or more of the following aspects:

- *Supporting environment information:* these solutions take into account some information that can be used to characterize the environment (e.g., time, location), users, and the protected resources;
- *Adding new entities into the RBAC:* these approaches add new entities on the RBAC model for taking into account dynamic aspects of the environments;
- *Dynamic user and permission assignments:* Unlike the traditional RBAC, some approaches make user-role and role-permission assignments, dynamically;
- *Extending the constraints:* some proposal have added new types of constraints.

We differ these approaches from context-aware and context-based solutions described in the Chapter 3, because they do not make explicit use of the context concept.

2.5.1 Temporal dimension of roles

RBAC models presented in the section 2.4 do not address the requirement related to temporal constraints on roles. For example, in the case of part-time staff in an organization, which is authorized to work only on working days between 9 AM and 1PM, the role assigned to it should be enabled only during the aforementioned temporal intervals. Thus, RBAC systems should be able of enabling and disabling roles according to temporal constraints defined for activating/deactivating them.

To cope with these requirements, Bertino *et al.* proposed the Temporal-RBAC (TRBAC) model [Bertino 2001] that extends the RBAC model in order to support temporal constraints on enabling/disabling roles. They defined the *Role Enabling*

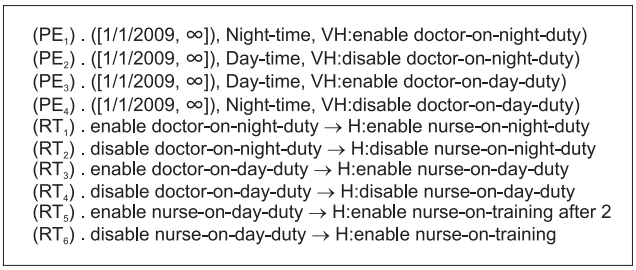


Figure 2.18: Example of Role Enabling Base (REB) [Bertino 2001].

Base (REB) in order to describe temporal constraints on the enabling of roles, which is composed by *periodic events* (PE) and *role triggers* (RT).

Periodic events have the form *(I, P, p:E)*, where I is a time interval, P is a period expression, and $p{:}E$ is a prioritized event expression. Role triggers have the form $E_1, \ldots, E_n, C_1, \ldots, C_K \rightarrow p : E\ after\ \Delta t$, where the E_is are simple event expressions, the C_is are role status expressions, $p : E$ is a prioritized event expression, and Δt is a duration expression.

Figure 2.18 illustrates an example of REB for a medical domain. VH (Very High) and H (High) denote prioritized event expressions with $H \prec VH$. The periodic events (PE) and role triggers (RT) in the REB state that the *doctor-on-night-duty* role must be enabled during the night (see PE_1 and PE_2), whereas the role *doctor-on-day-duty* must be enabled during the day (see PE_3 and PE_4). Role triggers RT_1 and RT_2 state that the role *nurse-on-night-duty* must be enabled whenever the role *doctor-on-night-duty* is. Role triggers RT_3 and RT_4 impose the same constraint for *doctor-on-day-duty* and *nurse-on-day-duty*, respectively. Finally, role triggers RT_5 and RT_6 specify that the role *nurse-on-training* must be enabled only during the daytime when two hours after role *nurse-on-day-duty* is enabled.

Joshi *et al.* [Joshi 2005] have extended the model proposed in [Bertino 2001] that only addresses the role enabling constraints. They proposed a Generalized Temporal Role Based Access Control (GTRBAC) model that allows specification of a comprehensive set of time-based access control policies, including temporal constraints on role enabling, role activations, user-role and role-permission assignments. Moreover, GTRBAC model extends the syntactic structure of the TRBAC model and its event and trigger expressions subsume those of TRBAC. Unlike TRBAC, GTRBAC allows expressing role hierarchies and separation of duty (SoD) constraints for specifying fine-grained temporal semantics.

The approaches described in this section proposed RBAC extensions in order to take into account temporal constrains on the RBAC components, such as user assignments, permission assignment, and role hierarchy. In some scenarios, however, it is desirable that users are not able to assume roles when they are not located in the supposed location for the accomplishment of their tasks. For example, a doctor should not have access to their patient records when he/she is not located in the

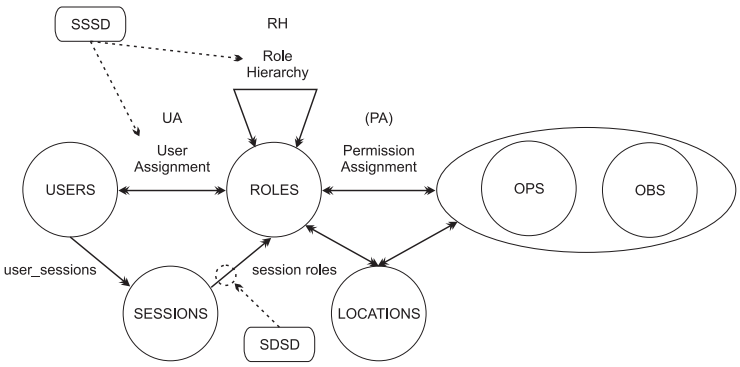


Figure 2.19: Spatial Role-Based Access Control Model (SRBAC) [Hansen 2003].

hospital. In the next section we describe RBAC extensions proposed in order to support such location-based access control policies.

2.5.2 Spatial dimension of Roles

In order to protect access to data in location-aware services, applications require the definition of spatially aware access control policies. Location information needs to model physical environments and the system should be able to identify the location of entities in that model. Numerous location models have been proposed and can be globally categorized into classes [Jiang 2002]: *hierarchical models*, which follow a topological, descriptive or symbolic representation of the physical environment (e.g., room, floor, building); *Cartesian location model*, which use metric or geometric coordinates (e.g., GPS[11]). A location information can be classified as absolute (i.e., the exact location of an entity in a model) or relative (i.e., location of an entity in relation to the location of another).

Hanser *et al.* [Hansen 2003] proposed the Spatial Role-Based Access Control Model (SRBAC), which extends the RBAC model in order to constrain the set of permission available to roles that a user may activate at a given location. Permission sets depend on spatial information within the same active role, thus SRBAC reduces a number of roles specified within the system, simplify security administration.

SRBAC model consists of the following five basic component sets (see Figure 2.19): $Users$, $Roles$, $Permission(PRMS)$, $Sessions$, and $Locations(LOC)$. *Locations* are represented by means of symbolic expressions called *location expressions* that describe location domains identifiable by the systems. In the following we present a summary of SRBAC definitions:

- $USERS$, $ROLES$, $PRMS$, $SESSIONS$, and LOC, represent the finite set of users, roles, permission, sessions, and locations respectively;

[11] Global Positioning System

- $UA \subseteq USERS \times ROLES$, the user assignment;
- $assigned\_users(r : ROLES) \rightarrow 2^{USERS}$,
  the mapping of a role onto a set of users.
  Formally, $assigned\_users(r) = \{u \in USERS | (u, r) \in UA\}$;
- $PA \subseteq ROLES \times LOC \times PRMS$, the relation that assigns a permission to a role available in a location;
- $assigned\_permission(r : ROLES, l : LOC) \subseteq 2^{PRMS}$, the mapping of a role r onto a set of permission based on location. $assigned\_permission(r, l) = \{p \in PRMS | (r, l, p) \in PA\}$;
- $user\_sessions(u : USERS) \subseteq 2^{SESSIONS}$, assigns a user onto a set of sessions;
- $session\_roles(s : SESSIONS) \subseteq 2^{ROLES}$, the mapping of each session to a set of roles;
- $avail\_session\_permission(s : SESSIONS, l : LOC) \subseteq 2^{PRMS}$, the permission available in a session for a location,
  $\bigcup_{\{r \in session\_roles(s)\}} assigned\_permission(r, l)$.

SRBAC supports two types of separation of duties: *Spatial Static Separation of Duty* (SSSD) and *Spatial Dynamic Separation of Duty* (SDSD). SSSD enforces constraints on the assignment of users to roles with regards to location. This implies that if a user is assigned to a role in a given location, the user cannot be assigned to another role in this location if these roles are conflicting. Thus, a user may never activate two roles that share a SSSD relation for a given location. SDSD is enforced on permission assigned to roles that are activated in a user's session. SDSD allows users to be assigned to two or more roles that are not conflicting when activated in separate sessions for a given location.

A well-known spatial-aware model by the scientific community is GEO-RBAC. This model was proposed by Bertino *et al.* [Bertino 2005, Damiani 2007] as an extension of the RBAC, in order to deal with spatial and location-based information when making access control decisions. In the GEO-RBAC model, spatial entities are used to model objects, user positions, and geographically bounded roles. Roles are activated by the access control model based on the physical position of users, which are assigned to a logical position representing the feature (e.g., the road, the town, the region) in which they are spatially located. GEO-RBAC consists of three components referred to as Core, Hierarchical, and Constrained GEO-RBAC:

- Core GEO-RBAC specifies the basic concepts of the model that are used by the other components: notion of spatial role, role schema, real/logical position, activated/enabled role;

- Hierarchical GEO-RBAC extends the concept of hierarchy by introducing two novelties: i) two distinct hierarchies, one over role schemas and one role instances; ii) The formal definition of role activation and enabling in the presence of hierarchies;

- Constrained GEO-RBAC supports the specification of separation of duty constraints for spatial roles and role schemas.

In the GEO-RBAC model, objects have a geometric representation compliant with the OGC[12] simple feature geometric model [Consortium 1999]. The geometry of an object can be of type point, line, polygon, or recursively be a collection of disjoint geometries. All geometries contained in a reference space (i.e., a polygon) is denoted by the term *GEO*, and that reference space is denoted with *Minimum Bounding Box* (MBB)[13].

GEO-RBAC assumes that resources consist of data about entities of the real word that may occupy a position (named *features*). Features can be classified as spatial (they are associated with a location) or non-spatial (they are not associated with any location) features, which are represented by F_s and F_{ns} respectively ($F_s \cap F_{sn} = \oslash,\ and\ F = F_s \cup F_{sn}$).

The central idea of GEO-RBAC is the distinction between the concept of role schema (R_s) and role instance (R_i) (or spatial role). In fact, a role schema defines common properties of a set of spatially aware organizational functions with a similar meaning. Role schema specifies the type of logical locations and the granularity of the position that the users playing that role may occupy. A role instance is a role fulfilling the constraints defined at schema level. Therefore, a role instance has the same name of the schema role name whereas the spatial boundary of the role is a spatial feature with a precise semantics.

Figure 2.20 illustrates the Hierarchical GEO-RBAC model. R_i and R_s represent the set of role instances and role schemas, respectively; $RPOS$ is the set of real positions; U, SES, OPS, OBJ, and $PRMS$ are the set representing users, sessions, operations, objects, and permission, respectively; and RH_i and RH_s are Role Instance Hierarchy and Role Schema Hierarchy, respectively.

In the following, we present a summary of the relationships between the entities of the model.

- $SPA_s : R_s \times PRMS$, a many-to-many mapping permission-to-spatial role schema assignment relation;

- $SPA_i : R_i \times PRMS$, a many-to-many mapping permission-to-spatial role instance assignment relation;

[12] Open GeoSpatial Consortium. Site: http://www.opengeospatial.org/standards
[13] The smallest rectangle completely enclosing a set of points.

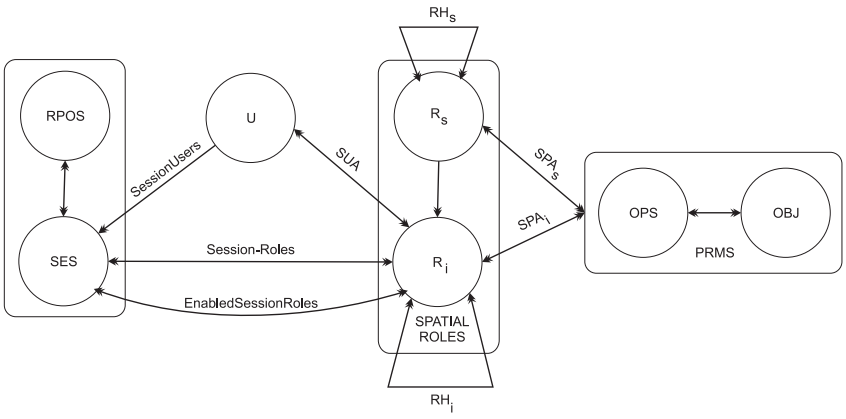


Figure 2.20: Hierarchical GEO-RBAC [Bertino 2005].

- $SUA \subseteq U \times RI$, a many-to-many mapping user-to-spatial role instance assignment relation;
- $SessionUser : SES \rightarrow U$, the mapping from a session s to the user of U;
- $SessionRoles : SES \rightarrow 2R_i$
  with $SessionRoles(s) \subseteq \{r|(SessionUser(s), r) \in SUA\}$.

$SessionRoles(s)$ correspond to the roles that can be potentially activated in a session s. However, depending on the user position during that session, only a subset of such roles is enabled and permission granted. Enabled roles are the basis for determining whether to grant or reject an access request. An access request is a tuple $< s, rp, p, o >$, stating that the user of session s located at real position rp wants to perform operation p on object o, thus a $< s, rp, p, o > \in SES \times RPOS \times OPS \times OBJ$. An access request can be satisfied at real position rp, if permission (p, o) belongs to the set of permission assigned to the roles that are enabled in s when the session user is in position rp. See [Bertino 2005, Damiani 2007] for more details about this model.

Zhang *et al.* [Zhang 2006] proposed a location-aware extended RBAC model, named LRBAC. Unlike GEO-RBAC, it describes the logical location domain according to the security policy of an organization, not fully in geometric ways. LRBAC is formally modeled for dealing with spatial restrictions in an access control system. They have introduced the concept of spatial role, effective role (like RBAC roles), and spatial role hierarchies. LRBAC allows modeling objects, user locations, and geographically bounded roles. The roles are automatically activated/deactivated by the position of the user. The evaluation of policies takes into account both the activated role of a requester and the his location. In this case, permission assigned to users depend on their location and the objects to which permission must be granted are located in the controlled environment.

Spatial role (SR) combines roles with logical location domain that indicates the

spatially bounded role. A spatial role is a pair $(r, ldom)$, where r is the role name and $ldom$ the logical location domain of the role. The logical location domain defines the boundaries of the space in which the role can be assumed by the user. The same spatial role can be associated with different location domains. We describe a summary of LRBAC model in the following:

- U, SR, OP, O, S, $RLOC$, $LDOM$ stand for users, spatial roles, operations, objects, sessions, real locations, and logical location domains, respectively;
- $PRMS = 2^{OP \times O}$, is the set of permission;
- $PA : PRMS \times SR$, is a many-to-many mapping permission to spatial role assignment relation;
- $AssignedPrms : SR \rightarrow 2^{PRMS}$, the mapping of spatial roles onto sets of permission. Formally, $AssignedPrms(sr) = p \in PRMS | (p, sr) \in PA$;
- $UA \rightarrow U \times SR$, a many-to-many user to spatial role assignment relation;
- $AssignedSession : U \rightarrow 2^S$, assigns a user onto a set of sessions;
- $AssignedUser : SR \rightarrow 2^U$, the mapping of spatial role onto sets of users. Given a spatial role $(r, ldom) \in SR$, $AssignedUser((r, ldom)) = \{u \in U | (u, (r, ldom)) \in UA\}$;
- $SessionUser : S \rightarrow U$, is a function mapping each session s to the single user $SessionUser(s)$ that is constant during a session;
- $SessionRoles : S \rightarrow 2^{SR}$, is a function mapping each session s to a set of spatial roles $SessionRoles(s) \subseteq \{(r, ldom) \in SR | (SessionUser(s), (r, ldom)) \in UA\}$.

In the LRBAC, $SessionRoles(s)$ corresponds to the roles that can be potentially activated in session s. If a user is assigned to several roles, it is up to her/him to decide which $SessionRoles(s)$ will be activated. Roles integrated into spatial information are dynamic in nature and users do not select the role to be activated directly. In fact, depending on the location in which a user is situated during the session, only a subset of such roles is effective and permission granted. Roles are automatically (de)activated by the environment.

The location-aware RBAC models presented in this section take into account spatial constraints when enforcing access control policies. However, it is desirable to consider simultaneously the spatial and temporal dimensions when defining and enforcing access control policies. In the next section we present some existing RBAC-extended approaches that take into account spatial-temporal dimensions in their models.

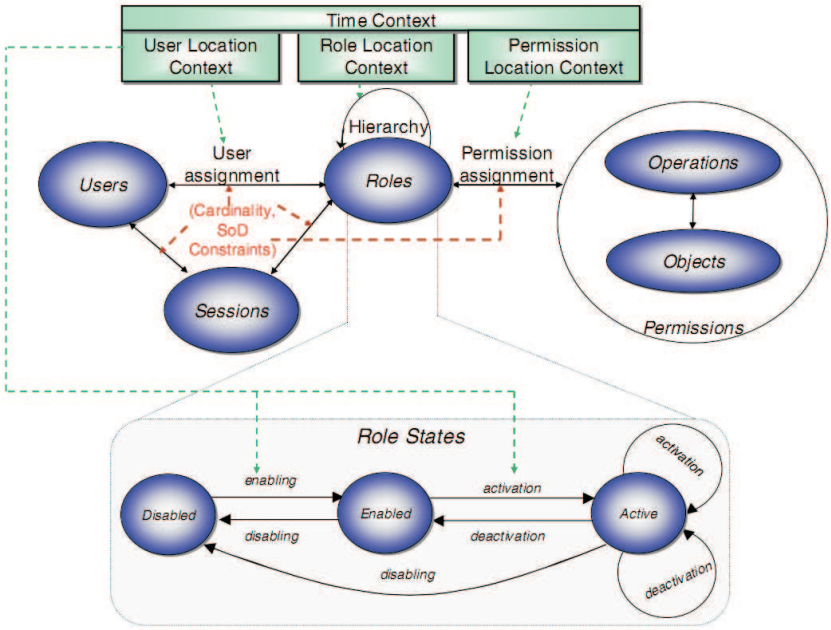


Figure 2.21: LoT-RBAC model [Chandran 2005].

2.5.3 Spatial-temporal dimension of Roles

Chandran *et al.* [Chandran 2005] proposed a Location and Time-based RBAC (LoT-RBAC) model by extending the GTRBAC model [Joshi 2005]. LoT-RBAC addresses access control requirements of highly mobile and dynamic environments to provide both location and time based access control. Lot-RBAC uses a fine-grained spatial model including detailed location hierarchy and the notion of relative locations. Figure 2.21 illustrates the LoT-RBAC model. LoT-RBAC uses the notion of role being in three states introduced by Joshi *et al.* in the GTRBAC model [Joshi 2005]: *enabled*, *disabled* and *active*. The authors argue that the main RBAC entities (i.e., users, roles, and permission) can have its own location, named *location context* (see in Figure 2.21).

Basically, enabled roles in location l at time t can be activated by an user if he/she satisfies the location constraints associated with the role activation. To allow these state changes based on time, LoT-RBAC uses enabling, assignment, and activation of roles according to the location context. Therefore, role activation occurs when temporal and spatial constraints are satisfied. However, permission assignments are not dependent on location and time. It means that when a role is activated all the permission associated with the role can be invoked.

In [Ray 2007, Ray 2008] Ray *et al.* proposed a spatio-temporal role-based access control model based on the RBAC model. The authors consider two types of locations: *physical* and *logical*. Users and objects are associated with locations that correspond to the physical world. These are referred to as the physical locations.

A physical location is formally defined by a set of points in a three-dimensional geometric space. Moreover, they consider two kind of information to represent the time: *time instant* and *time interval*. This model has the same set of components as the RBAC model, but they are associated with location and time. In the following we present some definitions of this model:

- $UserLocations(u, t)$ and $UserLocations(u, d)$ that gives the location of the user at time instant t and during the time interval d, respectively;
- $ObjLocation(o, t)$ and $ObjLocation(o, d)$ takes as input the tuples (object o, time instance t) and (object o, time interval d), respectively, and return the location associated with the object;
- $RoleAllocLoc(r)$ gives the set of locations where the role can be allocated;
- $RoleAllocDur(r)$ gives the time interval where the role can be allocated. Some role s can be allocated anywhere, in such cases $RoleAllocLoc(s) = universe$. Similarly, if role p can be assigned at any time, $RoleAllocDur(p) = always$;
- $RoleEnableLoc(r)$ gives the location where role r can be activated and $RoleEnableDur(r)$ gives the time interval when the role can be activated;
- The predicate $UserRoleAssign(u, r, d, l)$ states that the user u is assigned to role r during the time interval d and location l;
- The predicate $SessionUser(u, s, d)$ indicates that a user u has initiated a session s for duration d;
- The predicate $SessionRoles(u, r, s, d, l)$ states that user u initiates a session s and activates a role for duration d and at location l;
- $PermRoleLoc(p, r)$ specifies the allowable locations that a user playing the role r must be in for him to get permission p. $PermObjLoc(p, o)$ specifies the allowable locations that the object o must be in so that the user has permission to operate on the object o. $PermDur(p)$ specifies the allowable time when the permission can be invoked;
- $PermRoleAcquire(p, r, d, l)$. This predicate is true if role r has permission p for duration d at location l;
- The predicate $PermUserAcquire(u, o, p, d, l)$ means that user u can acquire the permission p on object o for duration d at location l.

The authors also integrated location and temporal constraints into the two types of hierarchy identified by Joshi *et al.* [Joshi 2005] : *permission inheritance hierarchy*[14] and *role activation hierarchy*[15]. They also describe the impact of location and temporal constraints on the static and dynamic separation of duties.

[14] A senior role x inherits the permission of a junior role y.
[15] a user assigned to a senior role can activate a junior role

Aich *et al.* [Aich 2007] proposed a Spatio-temporal Role Based Access Control, named STARBAC. In [Aich 2009] the authors enhanced the capabilities of this model in order to include separation of duty and access control evaluation process (Enhanced Spatiotemporal Role Based Access Control - ESTARBAC). The authors proposed a spatio-temporal extension to the temporal role enabling and disabling approach as proposed by Bertino *et al.* in [Bertino 2001]. According to Bertino *et al.* [Bertino 2001], only enabled role can be activated by the user. Typically, a role in an organization is disabled by default, i.e., it is not ready for activation by the user. The transition of role from its disabled to enabled state is what is named *role enabling* and the reverse transition is typically known as *role disabling*. Therefore, STARBAC model allows to write constraint expressions which enable or disable role based on spatio-temporal factor (e.g., user request time, resource location).

STARBAC assumes both subject and resource to be potentially mobile in nature and hence, checks the location context of both subject and object against the spatial constraints. The model deals with logical location which is typically application dependent. It assumes a mapping which unambiguously maps the physical position (or point) into a set of logical locations. The time information is evaluated against the temporal constraints defined for the application. The granular point in temporal reference is a time instant.

STARBAC uses *Role Control Commands* to define expressions which encode the spatio-temporal resource access policy. They define the COND set which constitutes the condition part of role control commands. In the following we describe formally some STARBAC definitions:

- *Condition set (COND):* The set COND is the generic set of conditions. It consists of the following conditions: elements of SCOND (i.e., spatial COND), elements of TCOND (i.e., temporal COND), and elements of STCOND (i.e., spatio-temporal COND);

- If cd_1 and cd_2 are elements of COND then so are $cd_1 \wedge cd_2$ and $cd_1 \vee cd_2$;

- *Role Control Command:* The Role control command has the form $< c, command >$ where $c \in COND$ and command is either a command (e.g., enable r_1, disable r_2, $|r_1, r_2 \in R$).

An example of STARBAC role control command is $< (Office, Officehour), enableCLERK >$, where *Office* is an element of $SCOND$, $Officehour$ is a periodic interval included in $TCOND$ and the Role $CLERK$ is defined in STARBAC role set R. The set of the STARBAC role control commands defined for an organization constitutes STARBAC Control Base (SCB).

2.5.4 Generalized Roles

Michael J. Covington *et al.* [Covington 2000, Moyer 2001] have addressed the problem of securing applications that will access and control information resources in the home of the future, i.e., smart or aware homes. Their vision of the future is one which homes will have networked information appliances that are accessible via the Internet. Thus, intruders could, in theory, enter in the home digitally, since the security physical mechanisms (e.g., burglar alarms, dead-bolts) will offer little or no protection from these virtual attacks. Unlike a physical burglar, an electronic intruder can attack the aware homes at any time, from any location. Financial loss, public embarrassment and even physical harm are just a few of the many potential negative consequences of a breach in the digital security of smart homes.

In this scenario, some environment information, such as time and location, can be used to improve the traditional RBAC model in order to offer a more flexible access control mechanism. A real scenario of applicability is, for instance, the following: if an intruder is able to discover the identity of a real user (e.g., login and password) he/she might use it to try remote access of the protected resource of a smart home. By verifying the location of a user that is requesting access on resource, the access control system is able to deny the permission since he/she is not located on the room where the permission is allowed (e.g., the permission to turn-on a TV is allowed only if he/she is located on the living room).

From their point of view, an access control policy should constrain access to information or resources based on several factors, including attributes about the subject, the resource or the environment. For example, subjects can be classified as *resident* or *guest*, or even as *adult* or *child*. Then, access rights can depend on the subject's attributes, such as her identity, location, or even based on environmental factors (e.g., the temperature or the time of day). In addition, access to information objects or resources may depend on security-relevant attributes of the object's state.

In order to take into account this type of information, Covington *et al.* [Covington 2000, Moyer 2001] propose a Generalized Role-Based Access Control (GRBAC) model, which is an extension of traditional Role-Based Access Control (RBAC). It enhances traditional RBAC by incorporating the notion of *object roles* and *environment roles*, with the traditional notion of *subject roles*. These new types of roles allow one to define rich, easy-to-understand security policies without having significant technical knowledge of the underlying computer systems that implement those policies.

By defining these three types of roles, GRBAC uses information gathered from environment sensors (e.g., time, location) as a determining factor for making access decisions. The definition of environment roles allows the model to partially address the problem of context-unawareness in the traditional RBAC-based approaches. This extension unifies ideas from several existing access control models into one model that captures all security-relevant state in a system. The unification of all

relevant state into a single concept (roles) makes access control policies significantly easier to define and implement in GRBAC than in other models. In the following, we present the main characteristics of GRBAC:

- *Environment role:* it is based on any system state that can be accurately collected, such as location and time;

- *Object roles:* it allows the access control mechanism to capture various commonalities among the objects in a system, and use these commonalities to classify the objects into roles. Object roles can be based on any classifiable property of an object, including its date of creation, object type (image, source code, streaming video, etc.), sensitivity level (secret, top secret, etc.), or information about the object;

- *Role activation:* separation of duty and role precedence [Sandhu 1996] are both related to an authorized role set, because as the size of an authorized role set grows, separation of duty and role precedence become more difficult to manage. GRBAC solves this problem by using role activation. In this case, a subject must explicitly declare which roles he intends to use at anytime. Roles that have been declared active constitute the subject's active role set. Thus, only roles in the active role set can be used to execute operations;

- *Complex algorithm for making access decision:* In RBAC, if subject S wants to access object O, S must possess role R that is authorized to execute operation OP, such that can access O. In GRBAC, the access mediation algorithm is similar, but slightly more complex. Subject S possesses a set of subject roles, and object O possesses a set of object roles. In addition, the system keeps track of a set of environment roles.

In a GRBAC-based system, for S to access O, S must possess some subject role R_S, such that:

1. $\exists$ some object role R_O, owned by O;

2. $\exists$ some environment role R_E that is currently active;

3. $\exists$ some operation OP that allows R_S to access R_O when R_E is active.

Figure 2.22 presents the basic RBAC definition and rules. Clearly, the access mediation rule of the GRBAC is more complex than the corresponding rule for traditional RBAC.

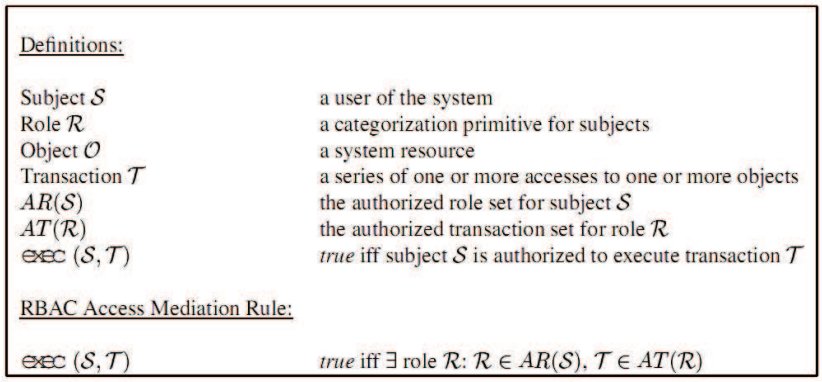


Figure 2.22: Basic RBAC Definitions and Rules [Covington 2000].

2.6 Conclusion

DAC solutions have as main advantages the flexibility and simplicity [TCSEC 1985]. Access control matrix (ACM) [Lampson 1974, Harrison 1976], authorization tables, access control list (ACL) [Samarati 2001], and capabilities [Wilkes 1979] are examples of DAC-based solutions presented in this Chapter. Despite its advantages, DAC-based solutions have some problems, such as the following: DAC do not support control of information flow and system-level access control policies.

Unlike DAC-based approaches, mandatory access control solutions [TCSEC 1985] offer means of assuring the confidentially and integrity of information flow. Moreover, MAC-based solutions supports the definition of global policies (i.e., system-level policies) and they could be used to prevent conflicts of interest. Multilevel security (MLS) [Bell 1973], the Ken Biba model [Biba 1977], and the Chinese Wall model [Brewer 1989] are examples of existing MAC solutions discussed in this Chapter. However, the main problems with MAC-based solutions are the correct classification of subjects and objects, and the complexity of security administration.

RBAC model was proposed [Ferraiolo 1992, Sandhu 1996, Ferraiolo 2003] in order to simplify the administration tasks and to support function-based access control. Basically, in RBAC-based models users are assigned to roles and roles are assigned to permission. Then, users are assigned to a subset of roles when he/she starts a session, getting the permission assigned to that set of roles. However, in the RBAC models the user-role and role-permission assignment are defined statically. Moreover, they do not take into account environment information when assigning roles and permission.

RBAC extensions were proposed in order to assign dynamically roles and permission, taking into account some environment information, such as time [Bertino 2001, Joshi 2005] and location [Hansen 2003, Bertino 2005, Damiani 2007]. However, context is not composed by only time and location information. Thus, Michael *et al.*

[Covington 2000, Moyer 2001] generalized the concept of roles in order to take into account other situational information.

Like RBAC models described in Section 2.4, RBAC-extended solutions presented in Section 2.5 have the same problems when applied in pervasive environments: we can not assume that it will be possible to assign a role to any user on the environment, given the mobility of users. Therefore, Chapter 3 presents existing access control solutions specifically defined for pervasive environments.

Chapter 3

Access Control Approaches for Pervasive Environments

Résumé: *Ce chapitre décrit l'état de l'art en matière de contrôle d'accès en précisant les caractéristiques particulières des environnements pervasifs. Nous proposons une classification des solutions basée sur le type de support à la connaissance du contexte. Le premier groupe est composé par des solutions qui ont pris comme point de départ le modèle RBAC. L'autre groupe est composé par des propositions qui ne prennent en compte que les informations contextuelles lors de la prise de décisions d'accès sur les ressources protégés. Nous présentons aussi un tableau de comparaison pour synthétiser les différentes approches existantes, dont nous pouvons observer que la majorité des modèles se base sur la notion de rôles pour délivrer les autorisations d'accès. Nous avons remarqué dans l'ensemble des propositions existants l'insuffisance au niveau de la modélisation du contexte et des difficultés pour recueillir correctement ces informations situationnelles. Ces aspects constitueront le point de départ pour les contributions de ce travail.*

Contents

3.1 Introduction

We present in this Chapter the existing access control approaches for pervasive environments. Some approaches are based on one or more traditional solutions described in Chapter 2 (e.g., RBAC, MAC, DAC). These solutions take into account specific access control requirements of pervasive environments when making access control decisions, such as context-awarenesses. We have divided the existing approaches in two groups: Context-Aware Access Control (CAAC) and Context-Based Access Control (CBAC) solutions.

3.2 Context-Aware Access Control (CAAC) solutions

In this section, we describe existing approaches that use context information as a way for assigning dynamic permission to users. Generally, these solutions use context information as a means of improving an existing model that is not context-dependent in its basis, such as RBAC model. On one hand, these solutions could work without using any context information. On other hand, the expressive power of access control policies will be more limited, i.e., it will not benefit from the flexibility of supporting context information.

3.2.1 Environment Roles

Covington *et al.* [Covington 2001] proposed an Environment Role-Based Access Control Model based on their earlier work in which they proposed a generalization of the basic RBAC model, named GRBAC [Covington 2000]. This new model allows policy designers to specify such environmental context through a new type of roles named *environment roles*.

In a system that implements this approach, there may be a very large number of environment roles. Role activation of environment roles is based on conditions in the environment where a request is made. These could include time, location or other contextual information that is relevant to access control. The state of the environmental conditions must be captured via sensors that are embedded in the environment. Thus, at access time the system must determine which of those roles are active in order to grant/deny permission.

For instance, suppose an access request is made at 5:30 p.m. on Monday, May 3, 2010, under a CPU load of 65% and a network load of 45%. To mediate the access request, the system must gather information about which environment roles are currently active. There may be an environment role called *high CPU load (over 70%)*, as well as roles for *Monday afternoons*, *weekdays*, and *business hours*. All of these roles are active at the time of the request. Also, unlike traditional access control models where requests are made explicitly by subjects, requests in smart en-

vironments (e.g., Aware Home) may be generated based solely on the environmental conditions.

They have formalized the Environment Role-Based Access Control Model following the RBAC96 specification [Sandhu 1996]:

- From $RBAC_0$, they keep U, R, P, and S. These capture users, roles, permission and sessions respectively;
- This model add ER and EC, where ER refers to *Environment Roles* and EC captures the *Environment Conditions* that are used to define such roles. To some degree, EC is analogous to U because the credentials associated with a user allow it to assume roles in R. Similarly, values of variables in EC allow certain roles in ER to be activated.

This model has the relations UA, PA, and EA, that define the associations between subject roles, users, permission assignments, and environment roles. These relations are as follows:

- $UA = U \times R$. This comes from RBAC and defines what roles in R a user from U is allowed to assume;
- $PA \subseteq P \times R \times 2^{ER}$. This captures permission that is assigned to a user role when a given set of environment roles is active. Thus, PA not only associates a permission with a subject role but makes it conditional on a set of active environment roles. Clearly, permission may change for a single subject role accessing a resource if the environmental conditions vary between requests.

The following functions define what user and environment roles can be activated:

- User: $S \rightarrow 2^R$. In a given session S, a set of roles can be activated for a user;
- Request: $EC \rightarrow 2^{ER}$. Although some environment roles can be activated for the duration of a session, changing conditions will require other roles to be evaluated every time. Thus, based on the environmental conditions, a set of environment roles are activated at the time of a request.

Figure 3.1 illustrates examples of Subject (a) and Environment Role Hierarchy (b). A request that requires permission p can be granted if (1) $<p, r, e\text{-}set> \in PA$, (2) the subject role r is in the active role set of the user making the request, and (3) the environment roles that are active in the current environmental conditions EC contain the roles in e-set.

The system administrator is responsible for defining environment roles by using a prolog-style logical language for expressing policies, named Generalized Policy Definition Language (GPDL) [Covington 2000]. Statements are used to define roles, sub-role relationships, transactions, and policy rules. The syntax is described below:

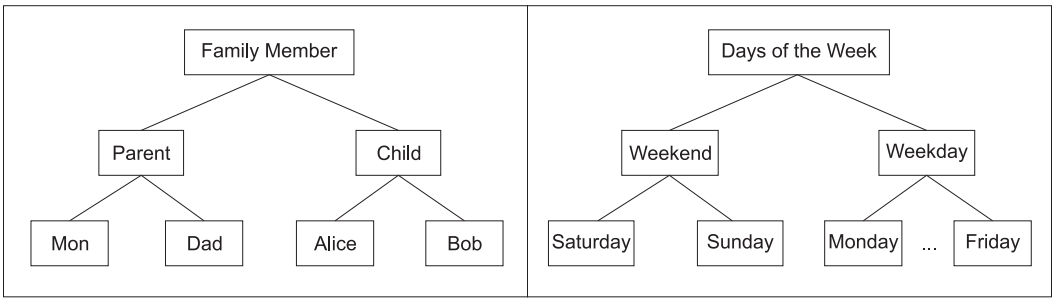


Figure 3.1: Examples of Subject and Environment Role Hierarchy [Covington 2001].

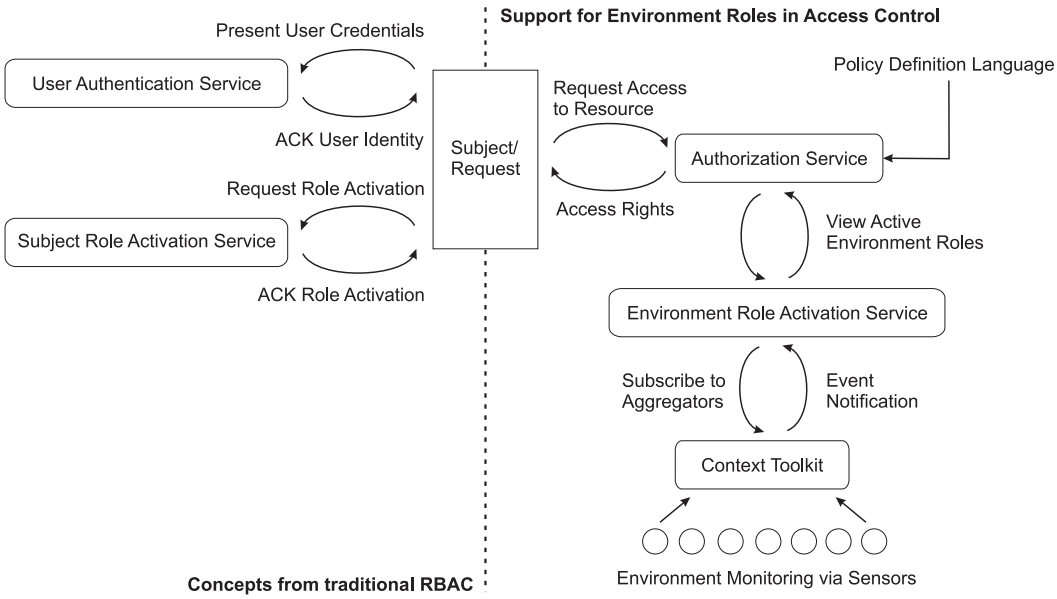


Figure 3.2: Transactions with Environment Roles [Covington 2001].

Definition of Environment Roles: *erole(erole_name).*
Examples: *erole(weekend)*, *erole(business_hours)*;

Role relationships: *role_rel(erole_name, entry_condition)* and *role_rel(parent_role, child_role).* Examples: *role_rel(business_hours, 08:00 < time_of_day < 17:00)*, and *role_rel(sunday, day_of_week=SUNDAY)*;

Error rules: in order to keep track of errors due to conflicting definitions in the rule base, it is necessary to have error rules. Example: *error(erole1, erole2).* In this example, the rule states that given two environmental rules *erole1* and *erole2*, the system cannot simultaneously activate both of the rules.

In order to manage environmental context information the authors proposed an architecture based on the Context Toolkit that has been developed at Georgia Tech

[Dey 1999]. This toolkit provides abstractions for assessing environmental state which could be used to manage environment roles. This architecture addresses issues such as role activation and authorization based on environment roles.

Figure 3.2 illustrates the proposed access control architecture, showing the access request from beginning to the end. For instance, suppose that a user U wants to use a service. Whether done implicitly via sensors or explicitly, U presents credentials to the system and is provided with a set of active subject roles. Ultimately, these subject roles will help to determine the resources U is allowed to access. These transactions are fundamentally consistent with those found in traditional RBAC. With a set of active roles, U will be able to request access to a particular resource in the environment.

User's request is forwarded to the centralized authorization service that enforces the current security policy. A policy grants access to U under certain conditions. In order to verify those environmental conditions, the authorization service contacts the environment role activation service. The environment role activation service, which interacts securely with the Context Toolkit [Dey 1999], has already received notification from the aggregators. This set of active roles is returned to the authorization service. The set of environmental active role, along with the subject role and resource request, provides a match to the rule specified in the security policy. Access rights are therefore granted to U.

In this approach, a pervasive environment is composed by many devices and services which are centrally administered. Authorization service ensures that access rules are consistent across all resources and allows for any resource to enforce security policies. A client or subject desiring to access a protected resource must first contact an authorization server to obtain the required credentials.

However, it is likely that not all of environment roles are relevant to the access control decision that must be made. Testing every environment role on every access control mediation would be prohibitively expensive, so the system should employ an efficient means of role entry testing for environment roles. They solve this problem by using the environment role activation service to automatically activate roles when appropriate. By maintaining an internal data structure of all active roles, the environment role activation service can efficiently interact with the authorization service for making access control decisions.

3.2.2 Space roles

Sampemane *et al.* [Sampemane 2002] proposed an access control model for Active Spaces[1]. This model provides support for both discretionary and mandatory access control policies, using role-based access control techniques for easy administration of

[1] Active Spaces are physical spaces augmented with heterogeneous computing and communication devices along with supporting software infrastructure.

users and permission. The model recognizes three kinds of user roles: *system roles*, *space roles*, and *application roles*. System roles are assigned when user accounts are created, and define users' generic permission for certain classes of resources within the entire system. Within each Active Space, access control policies are expressed in terms of space roles.

Active Spaces within a system has an administrator who sets access control policies for resources. When users enter a space, their system role is mapped into an appropriate space role automatically. Application roles allow an application to specify a customizable access control policy. For instance, a presentation application may require that only the *presenter role* be allowed to control the slides in the presentation. Application roles are mapped into space roles and access control is performed on these resulting space roles.

This model supports four distinct space modes of collaboration:

- i) *individual:* it allows a single user in a space all the rights that are given by her role;
- ii) *shared:* a group of users share the space without any special trust relationship between them;
- iii) *supervised-use:* some users need more permission than the group to complete an activity;
- iv) *collaborative:* users in a space trust the people they are working with, and are able to delegate their permission to the group.

An access control request has three parameters: a subject making the access request, a system object, and the specific object right (or method) being requested. Access rights to objects are traditionally stored in access control lists (ACL) or capability control lists (CL), which are implemented by an access matrix. The specification of the proposed model is given below:

- U: set of USERS
- R_{sys}: set of SYSTEMROLES
- R_{space}: set of SPACEROLES ($R_{sys} \subseteq R_{space}$)
- R_{grp}: set of GROUPROLES ($R_{grp} \subseteq R_{space}$)
- R_{app}: set of APPROLES ($R_{app} \subseteq R_{space}$)
- R_{dev}: set of DEVICEROLES
- S: set of SERVICES (objects in the system)
- OD: set of OWNEDDEVICES ($OD \subseteq S$)

- AL_s: set of $\{\langle r : roles; m : methods\rangle\}$ (access control list for a service $s \in S$)
- A: set of all $AL_s : s \in S$ (conceptual Access Matrix)
- $Mode$: enum $\{Ind, Shared, Collab, Super\}$ (space modes)
- C: set of CREDENTIALS which are one of
  typeof(u: USERS; r: SYSTEMROLES)
  owns(u: USERS, o: OBJECT)
  exports(s: SERVICE; m: METHODS)
- URA: set of $\{\langle u : USERS; r : ROLES\rangle\}$ (User-role assignment)
- AS: Current Active Space; users, services and ALs
- CU: set of users currently in space AS
- CRT: set of $\{\langle u : USERS; r_{sys} : SYSROLES; r_{space} : SPACEROLES\rangle\}$ (Current role assignment for users in space AS)
- $SysAdm \in R_{sys}, SpaceAdm \in R_{sys}$

In the following, we describe some functions included in the model (see the full specification in [Sampemane 2002]):

- $currentrole(r, mode) : ROLES \times MODES \rightarrow SPACEROLES$. This function returns the current space role according to the activated role and mode of space access;
- $allow(u, s, m) \wedge typeof(u, r) \in C$
  $\wedge(s \in S) \wedge exports(s, m)\ \wedge(currentrole(r, mode), m) \in AL_s \rightarrow$ **true**. This function checks credentials of a requester, and return true if the requested method is allowed.

For instance, consider an active space in a university with two types of users: *student* and *faculty*. In this system, there are four system roles: student, faculty, sysadm, spaceadm. In a given active space AS_1 (smart room), there are two protected devices: projector (P) and writeboard (B). When a user enters alone into the room, he/she must use his/her credential for attesting his/her *system role* (i.e., *student*). Then, the space starts a session with the *space mode* set to *individual* and assigns to that user a *currentrole* of *student*. In this mode, the AL show that this student is allowed to read and write the whiteboard. This space configuration is shown in Figure 3.3.

Access Control Matrix		
Role	**P**	**B**
Student	r	r, w
Faculty	r, c	r, w
Sysadm	r, c	r, w
Spaceadm	r, c	r, w

Current Role Translation		
user	**sysrole**	**spacerole**
u1	student	student

Mode
ind

Figure 3.3: Space configuration: Individual Session [Sampemane 2002].

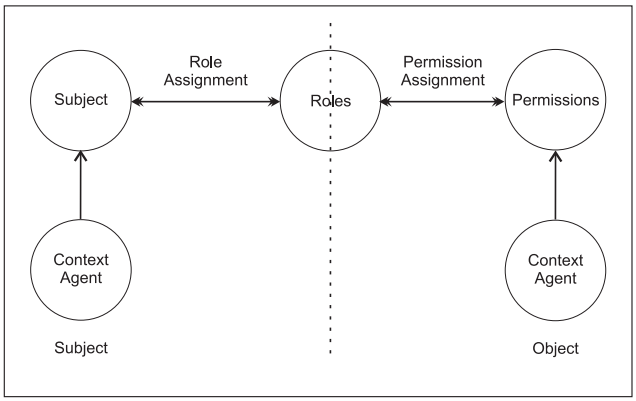


Figure 3.4: Dynamic Role-Based Access Control model [Zhang 2004].

3.2.3 Dynamic user-role and role-permission assignments

Zhang *et al.* [Zhang 2004] proposed a Dynamic Role-Based Access Control model, named DRBAC, that dynamically grants and adapts permission to users according to current context. DRBAC model combines the required credentials of users and the current context when making user-role and role-permission assignments. DRBAC addresses two key requirements of pervasive environments: i) access privileges of users change according to the current context of users; ii) a resource must adjust its access permission when its system information (e.g., network bandwidth, CPU usage, memory usage) changes. DRBAC dynamically adjusts *user-role assignments* and *role-permission assignments* based on context information. Figure 3.4 illustrates the main entities and the relationship between them of DRBAC access control approach: subject, context agent, role, permission.

Each user is assigned to a role subset from the entire role set. A resource has permission subsets for each role that will access that resource. During the interactions with an access control system implementing this approach, state machines are maintained by delegated access control agents representing the subjects, to navigate the role subset (*Role State Machine*), and the object, to navigate the permission subset for each active role (*Permission State Machine*). The state machine consists of state variables (i.e., role, permission), which encode its state, and commands, which transform its state. These state machines define the currently active role and its assigned permission in order to navigate the role/permission subsets according

to the changes in the context. Differences between DRBAC model and the RBAC model are described below:

- There exist a new entity, named ENVS, which represents the set of context information in the system. DRBAC incorporates an authorized *Context Agent* to collect context information;
- In a session, the active role will be changed dynamically among the assigned roles for each interaction;
- Context information is used to decide which role is active, i.e., dynamic user-role assignment (UA);
- Each role has assigned a set of permission, and the context information is used to decide which permission is active for that role, i.e., dynamic role-permission assignment (PA);
- There is a *Role State Machine* for each user, and a *Permission State Machine* for each role. Roles and permission are used as state variables, respectively. *Context Agents* are in charge of collecting context information, generating predefined events in order to trigger transitions in the state machines.

Kim *et al.* [Kim 2005] have proposed a similar Context-Aware Access Control model that extends RBAC via some functional components. Figure 3.5 illustrates the global infrastructure of a context-aware access control mechanism that implements the proposed model. In such model users are assigned to roles and roles are assigned to permission as in the RBAC model. Thus, users acquire permission through the roles. Default UA is a mapping that assigns a role to a user. Each user is assigned to a set of roles. Default PA is a mapping that assigns permission to a role.

Every role is assigned to a set of permission. Default roles are assigned to the users by the traditional RBAC, and then the role is activated or deactivated according to the changing context information of users. The context information is used to decide which role is active and which permission is active for that role. Default UA and PA are changed to context-aware UA and PA by applying the state checking matrix (SCM) to deal with context information. As a result, the model uses the context-aware UA and PA assignments, which dynamically grant and adapt permission to users according to the current context information of users.

In this model (see Figure 3.5), there are traditional RBAC elements and three new important components: *state checking agent*, *state checking matrix(SCM)* and *context-aware agent*. State checking agent maintains the role subset for each user, monitoring the environment status of users and dynamically changing their active roles.

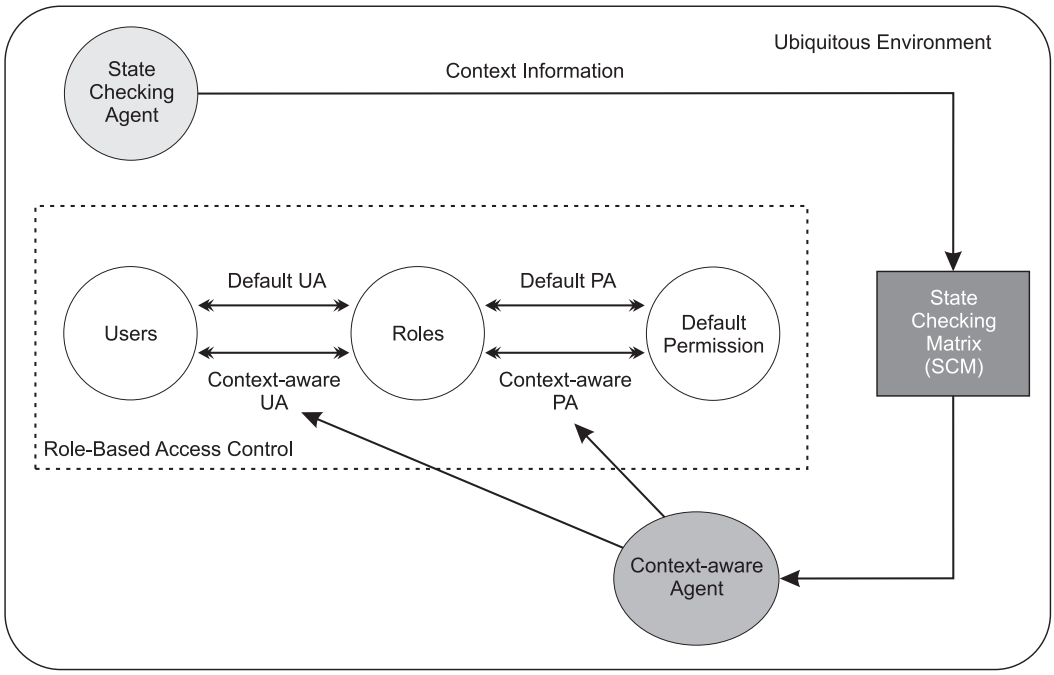


Figure 3.5: Context-Aware Access Control Model [Kim 2005].

	Location1	Location2	Location3	Location4
UserX(R1)	Active	Inactive	Inactive	Inactive
UserY(R2)	Active	Active	Inactive	Active
Admin(R3)	Inactive	Active	Active	Inactive

Figure 3.6: SCM for location information [Kim 2005].

State checking matrix deals with the context information (e.g., location, time, and resources such as network bandwidth and memory usage), (de)activating roles of users. *Context-aware agent* maintains the permission subset for each role, monitoring changes on the state checking matrix and dynamically changing default UA and PA to context-aware PA and UA.

In this model, access control decisions depend on the current status of all monitored context information. A role is only activated when all context elements restricting that role are activated. For instance, consider that the system uses location and time as monitored context information for (de)activating roles and permission. Then, it is necessary a SCM for each type of context information. Figure 3.6 and Figure 3.7 illustrate examples of SCM for location and time information, respectively.

In this example, user's role is only active when location and time are activated. The concept of activeness of role can be described as follows:

	Time1	Time2	Time3	Time4
UserX(R1)	Active	Active	Inactive	Inactive
UserY(R2)	Active	Active	Active	Active
Admin(R3)	Inactive	Active	Active	Inactive

Figure 3.7: SCM for time information [Kim 2005].

$$\textbf{Activeness of role} = Context(Context_1,\ Context_2, \ldots,\ Context_n) = Context(Active,\ Active, \ldots,\ Active) = \textbf{Active} \quad (3.1)$$

In the case of using only location and time information, the activeness of role is described as following:

$$\textbf{Activeness of role} = Context(Location,\ Time) = Context(Active,\ Active) = \textbf{Active} \quad (3.2)$$

By verifying the activeness of roles in the *location2* at *time4* from the tables illustrated in Figure 3.6 and 3.7, only the *R2* will be activated. See this mapping bellow:

Activeness of *R1* = Context(Location2, Time4)= Context(Inactive, Inactive) = Inactive

Activeness of *R2* = Context(Location2, Time4)= Context(Active, Active) = Active

Activeness of *R3* = Context(Location2, Time4)= Context(Active, Inactive)= Inactive

Therefore, state checking matrix (SCM) is used to deal with context information, deciding the activeness of roles by mapping the status of context information (e.g., location and time) when user's context information changes.

3.2.4 Role Context and Context Role

Kumar *et al.* [Kumar 2002] have proposed the Context-Sensitive RBAC Model (CS-RBAC) that extends the RBAC model by introducing the notions of role context and context filters. A role context can be composed by user and object contexts, which is dynamically assigned to the users by verifying the Boolean constraint expressions named *context filters*. Figure 3.8 presents the CS-RBAC model. A formal definition of context-sensitive RBAC model is presented below:

- *R:* set of roles;
- *U:* set of users;

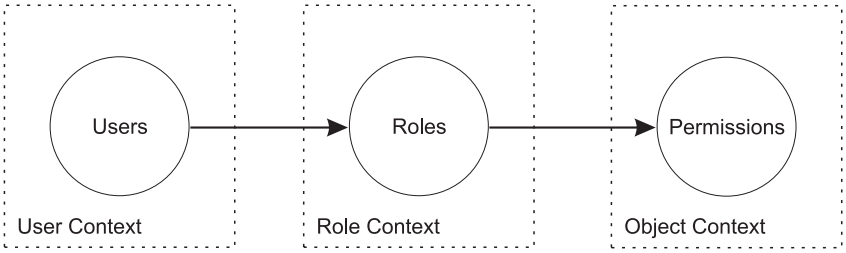


Figure 3.8: The Context-Sensitive RBAC model [Kumar 2002].

- *C:* set of classes of protected objects;
- *M:* set of methods/operations on the object classes in C;
- *OC:* set of all object instances;
- O_z : set of instances/objects of a given class Z;
- $P \subseteq M \times C$, $\{(m, c)|m \in M, c \in C, m \in methods(c)\}$, is the set of permission;
- $PA \subseteq P \times R$, the many-to-many permission-to-role assignment relation;
- $UA \subseteq U \times R$, the many-to-many user-to-role assignment relation;
- UC : the set of all security-relevant attributes of the user, i.e. the user context;
- OC : the set of all security-relevant attributes of all target object classes;
- OC_z : the set of all security-relevant attributes of target object of a given class Z (object context);
- *Contextual Constraint CC:* $\bigcup_{L \in C} \{2^{UC} \times 2^{OC_L}\} \rightarrow 2^{U \times O}$, is a function, for a role, mapping a pair of *user context* and *object context* to a set of individual (user, object) pairs;
- *Role Context* is defined as $RC =< UC, OC, CF >$, a three-tuple consisting of the user context, the object context and *Context Filter*;
- *Context Filter CF:* $U \times O \rightarrow \{0, 1\}$, is a function that returns true if, for a role, the given (user, object) pair belongs to the set of (user, object) pairs allowed by the contextual constraint CC of that role;
- *Context-Sensitive Permission* S is defined over $U \times P \times O$ as $\{(u, p, o)|\exists r \in$ roles(u) such that (p, r) $\in$ PA $\wedge$ CF(u, o) = true $\wedge$ u $\in$ U $\wedge$ o $\in$ O $\wedge$ p = $\{(m, c)|$ m=method(c) and c=class(o)$\}\}$

To enforce permission in CS-RBAC, the access control system first identifies the user's role memberships that permit the expected operation. For each assigned role, the corresponding role context is identified by verifying the values from the context

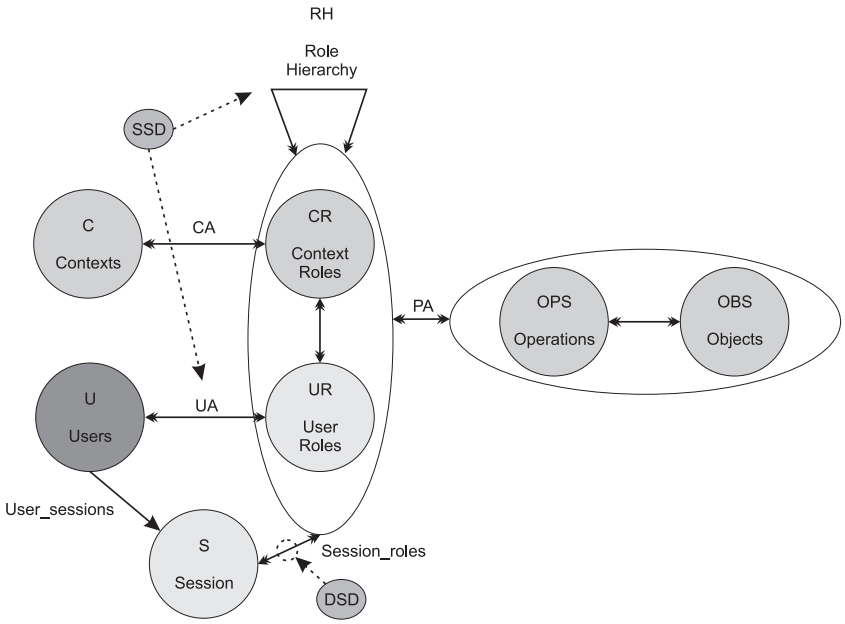


Figure 3.9: Context-Role Based Access Control Model (CRBAC) [Park 2006].

of users and objects, and its context filter is evaluated. If any of the filters evaluates to true, the permission will be allowed.

Park *et al.* [Park 2006] have proposed another model, called Context-Role Based Access Control (CRBAC), that adds the context-role notion to RBAC model. Context-role represents the environment state of the system when making access control decisions. Figure 3.9 presents the CRBAC components and the relationships between them, which is formally defined in the following:

- *U(users):* U represents a set of user;
- *C(context):* C represents a set of context information in the system. C captures the context information that is used to define *context role*;
- *R(roles):* R represents a set of roles. A role is composed by two roles: user roles and context roles;
- *UR(user roles):* UR represents a set of user roles. It is equal to ROLE in traditional RBAC;
- *CR(context roles):* CR represents a set of context roles, which are used to capture security-relevant context information about the environment for use in CRBAC policies;
- *P(permission):* P represents a set of permission;
- *S(sessions):* S represents a set of sessions. A role is activated for user during each session. Activated role is a mapping between user roles and context roles;

- *CRBAC* supports *Static Separation of Duties (SSD)* on the *Role Hierarchy* and *UA* associations;
- *CRBAC* supports *Dynamic Separation of Duties (DSD)* on *session-role* associations;
- $UA \subseteq U \times UR$, a many-to-many mapping user-to-user role assignment relation;
- $assigned\_users(ur : UR) \rightarrow 2^U$, the mapping of user_role ur onto a set of users. $assigned\_users(ur) = \{u \in U | (u, ur) \in UA\}$
- $R \subseteq 2^{(UR \times CR)}$, the set of roles;
- $PA \subseteq P \times R$, a many-to-many mapping permission-to-role assignment relation;
- $assigned\_permission(r : R) \rightarrow 2^P$, the mapping of role r onto a set of permission;
- $user\_sessions(u : U) \rightarrow 2^S$, the mapping of user u onto a set of sessions;
- $session\_roles(s : S) \rightarrow 2^R$, the mapping of session s onto a set of roles. $session\_roles(s_i) \subseteq r \in R | (session\_users(s_i), r) \in UA$.

Like in RBAC models, a transaction specifies a particular action to be performed in a CRBAC system. A transaction of CRBAC is a tuple in the form of $< user\_role,\ context\_role,\ permission >$. A policy database consists of a transaction listing paired with a permission bit for each transaction (i.e., allow or deny).

In [Li 2008], Li and Cao have proposed a CRBAC model similar to that proposed by Park *et al.* [Park 2006]. However, they have proposed three types of context roles: time-related, location-related, and trust-related context roles. Furthermore, they present an algorithm for activating roles based on these context roles.

3.2.5 Role States

Chae *et al.* [Chae 2006] have proposed an access control model that supports context information by managing three role states: assign (a role is assigned to a user), disable (the role is deactivated when constraints are unsatisfied), and enable (the role is activated when constraints are satisfied). Roles are assigned to users when starting the sessions (like in RBAC model), then the system checks time and location constraints on roles in order to change the role state (i.e., disabled or enabled).

Figure 3.10 shows the possible role states of the proposed model. The components of this model that differ from the RBAC elements are described below:

- $Location\_hierarchy:\ LH \subseteq Locations \times Locations$
- $Constraints\_UA(c : Constraint) \rightarrow UA$

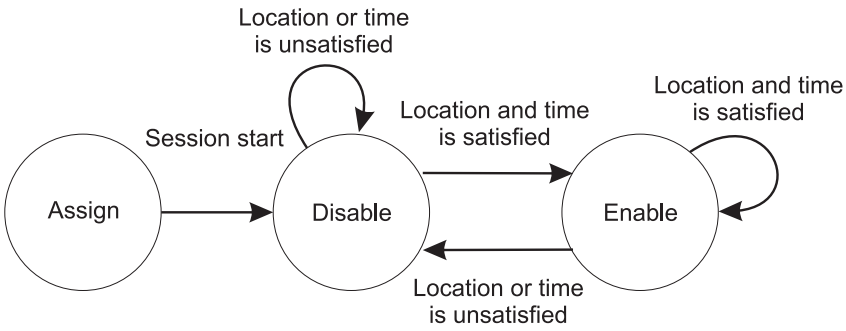


Figure 3.10: Role states [Chae 2006].

- $Constraints = Location|Times|Locations \times Times$
- $Assigned\_users(r : Roles) \rightarrow 2^{Users}$
- $Assigned\_permission(r : Roles) \rightarrow 2^{Permission}$
- *Constraint Expression:* given a role, a constraint expression c is defined as $c = L_c|T_c|L_c \times T_c$
- *User/role/constraint Expression* : given a user u, roles r and constraints c, the proposed assignment expression is defined as $u : \{c, R\}$.

For instance, the temporal constraint expression
$Alice : \{13:00 : 18:00, enable\ Part-timeNurse\}$ enables the *part-timeNurse* role from 1 p.m. to 6 p.m. The spatial constraint expression $Bob : \{Room302, OperatingRooms, enable\ patientRecords\}$ enables Bob to access patient records when he is in his office or in any operating room. Finally, the spatial-temporal constraint expression
$Alice : \{20:00 : 4:00, OperatingRooms, enablepart-timeNurse\}$ enables the role *part-timeNurse* when Alice is in any operating room, from 8 p.m. to 4 a.m.

3.2.6 Context-based constraints

Neumann *et al.* [Neumann 2003] proposed an approach that uses special RBAC constraints to base certain access control decisions on context information. Context constraint is defined as a dynamic RBAC constraint that checks the actual values of one or more contextual attributes for predefined conditions. If these conditions are satisfied, the corresponding access request can be permitted. Accordingly, a conditional permission is an RBAC permission which is constrained by one or more context constraints.

The authors differentiate constraints between *static* and *dynamic* constraints. Static constraints refer to constraints that can be evaluated directly at design time of an RBAC model (e.g. static separation of duties). However, dynamic constraints

can only be checked at runtime according to the actual values of specific attributes, or with respect to characteristics of the current session (e.g. dynamic separation of duties, or time constraints). Another classification criterion used by the authors is the distinction of *endogenous* and *exogenous* factors. Endogenous constraints are constraints that relate to intrinsic properties of an RBAC model, and inherently affect the structure and construction of a concrete instance of an RBAC model (e.g., static separation of duties - SSD). Exogenous constraints are constraints that apply to attributes that do not belong to the core elements of an RBAC model (e.g., time constraints that restrict role activation to a specific time interval).

Constraints can also be subdivided in *authorization constraints* and *assignment constraints*. Authorization constraints are constraints that place additional controls on access control decisions. Thus, even if a subject is in possession of permission that grants a certain access request, the access can only be allowed if the corresponding authorization constraints are fulfilled at the same time. For example, such constraints can be applied to implement access control policies based on access histories. Assignment constraints are constraints that control the assignment of permission and roles (e.g., maximum and minimum cardinalities, or separation of duty constraints).

However, this approach is based on the notion of *context constraints*. A *context constraint* specifies that certain context attributes must meet certain conditions in order to permit a specific operation. With respect to the categories mentioned previously, context constraints are *dynamic exogenous authorization* constraints. A context constraint is defined through the terms *context attribute*, *context function*, and *context condition*:

- *Context attribute:* represents a certain property of the environment whose actual value might change dynamically (e.g., time, date, or session-data), or which varies for different instances of the same abstract entity (e.g. location, ownership, birthday, or nationality). Each context attribute CA represents a variable that is associated with a domain CA which determines the type and range of values this attribute may take;

- *Context function:* it is a mechanism to obtain the current value of a specific context attribute. For example, a function date() could be defined to return the current date. A context function can also receive one or more input parameters. For example, a function $age(subject)$;

- *Context condition:* it is a predicate (i.e., a Boolean function) that compares the current value of a context attribute either with a predefined constant, or an other context attribute of the same domain. The corresponding comparison operator must be an operator that is defined for the respective domain. All variables must be ground before evaluation. Therefore each context attribute is replaced with a constant value by using the according context function prior

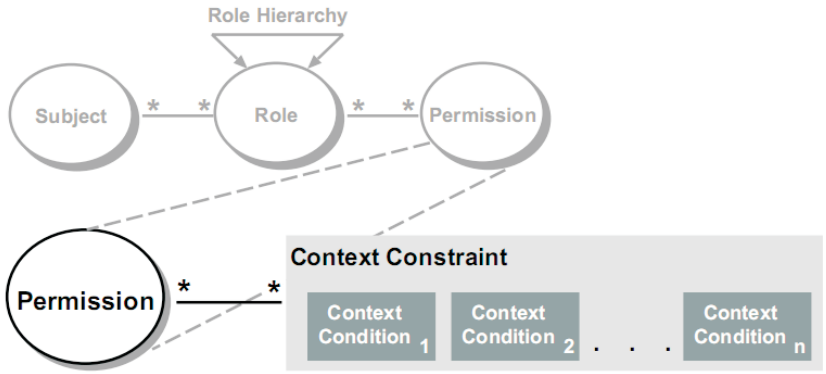


Figure 3.11: RBAC permission with context constraint [Neumann 2003].

to the evaluation of the respective condition. Examples for context conditions can be $cond1 : date() = 2003/01/01$;

- *Context constraint:* it is a clause containing one or more context conditions. It is satisfied if all its context conditions hold. Otherwise it returns false.

Context constraints are used to define *conditional permission*. *Conditional permission* is permission that is associated with one or more context constraints, and grants access if and only if (iff) each corresponding context constraint evaluates to *true*. Therefore conditional permission grant an access operation if the actual values of the context attributes captured from the environment fulfill the attached context constraints. The relation between context constraints and permission is a many-to-many relation (see Figure 3.11).

3.3 Context-Based Access Control (CBAC) Solutions

In this section we present the existing solutions that use context as the central concept to assign permission to users. The approaches described here consider that pervasive service provisioning requires a paradigm shift from subject-centric (e.g., identity, group, role) to context-centric access control solutions. Therefore, access control solutions for pervasive environments should consider context as a first-class principle to guide both policy specification and enforcement process.

3.3.1 Context attributes

Covington *et al.* [Covington 2006] have defined an access control model that uses *contextual attributes* as central concept to capture the dynamic properties of a mobile environment, including attributes associated with users, objects, transactions, and the environment. A contextual attribute represents a measurable contextual

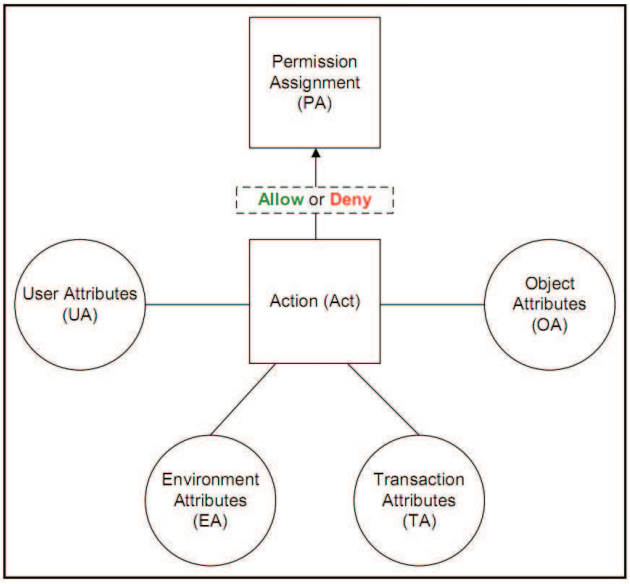


Figure 3.12: CABAC Model [Covington 2006].

primitive, such as location, time, temperature. It can be associated with any of the following entities:

- The user(s) making the access request;
- The object or resource being accessed;
- The access transaction itself.

The Contextual Attribute-Based Access Control Model (CABAC) is not an extension of existing access control models, removing the need to specify *actors* in the system. CABAC uses situations described as contextual attributes to define access policy. Figure 3.12 illustrates the CABAC model. *User attributes (UA)* capture properties of the subject that initiated the access request. *Object attributes (OA)* are properties that describe resources being protected by the access policy. *Environment attributes (EA)* describe properties of the physical environment at the time a transaction takes place. Finally, *transaction attributes (TA)* capture information about the transaction as it takes place. Moreover, they defined three other entities: Environment *Attributes (EA)*, *Action(Act)*, and *Permission Assignments (PA)*.

Environment Attributes share many characteristics with the other classes of attributes. These could include temperature, ambient noise, or other contextual information that is relevant to access control. Although some environment attributes can be activated for the duration of an entire session, changing conditions will require other attributes to be evaluated every time. Thus, based on the environmental conditions (EC), a set of environment attributes are activated at the time of a request.

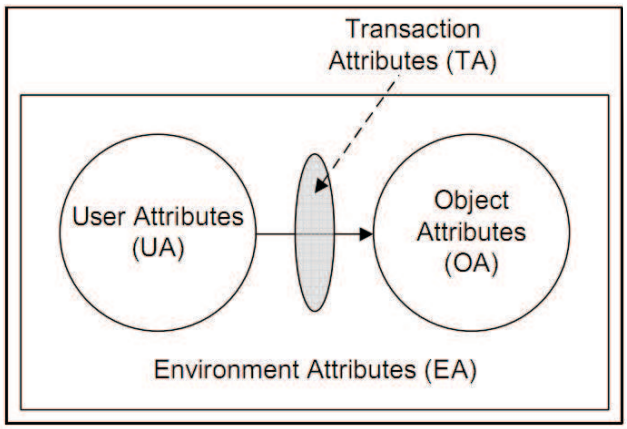


Figure 3.13: Transaction Overview using Contextual Attributes [Covington 2006].

Permission Assignments capture the privileged actions that a subject is authorized to hold or exercise on an object. The authorization is determined based on user attributes, object attributes, transaction attributes, and environment attributes. The following function captures the rights that are assigned to a user when a given set of environment attributes are active and she is attempting to access an object with a particular set of object attributes: $(< Act, UA, OA, EA, TA >, Perm) \in PA$, where $Perm = \{Allow,\ Deny\}$

Permission assignment (PA) not only associates permission with the user attribute(s), but makes it conditional on a set of active environment attributes. Rights may change for the same user accessing a resource if the object attributes, environment attributes, or even user attributes vary between requests. A CABAC request will be granted access rights if and only if (iff):

1. The policy rule assigning a specified action (Act) to an access request exists with the specified user attributes (UA), object attributes (OA), environment attributes (EA), and transaction attributes (TA) that match those specified in the set of permission assignments (PA);

2. The user attributes (UA) are active for the user making the current request;

3. The object attributes (OA) are active for the object being accessed by the user;

4. The environment attributes that are made active by the current environmental conditions (EC) are contained in the set EA;

5. The transaction attributes (TA) are active for the current transaction (see Figure 3.13).

3.3.2 Access Control Mechanisms

In this section we present some access control solutions that use context information as central concept for enforcing access control policies. These solutions are independent of any existing access control models, i.e., they do not use any access control model as basis for guiding their solutions.

3.3.2.1 UbiCOSM

Corradi *et al.* [Corradi 2004a, Corradi 2004b] proposed a dynamic and flexible security middleware, called UbiCOSM (Ubiquitous Context-based Security Middleware), that adopts context as the basic concept for security policy specification and enforcement processes. In UbiCOSM context-based access control policies are expressed at a high level of abstraction in terms of metadata and they are cleanly separated from the service logic.

UbiCOSM allows both administrators and users to specify access control policies in order to avoid illicit accesses to resources. UbiCOSM focuses on three main aspects: context-centric access control, active context view provisioning to mobile users, and privacy support in the propagation of user context information. UbiCOSM access control decisions depend on dynamic context attributes, such as resource state and availability, in addition to more traditional attributes, e.g., the identity/role of user requesting a resource access. UbiCOSM distinguishes two different kinds of context (see Figure 3.14: physical and logical.

- *Physical contexts:* it identifies physical spaces delimited by specific geographical coordinates. A user operates in a particular physical context depending on their current location. At any time, one user can belong to only one physical context. Physical contexts define specific boundaries for access control policy management: each physical context holds references to the protected resources;
- *Logical contexts:* it identifies logical states of both physical contexts and entities composing an ubiquitous service deployment scenario, e.g., users and resources. Logical states depend on logical properties, such as temporal conditions, resource availability and status, user activities. At any time, entities may be associated with different logical contexts.

UbiCOSM adopts a RDF-based[2] standard format for context representation to overcome heterogeneity of data representation over different architectures (see Figure 3.14). Both physical and logical contexts have a *Name* that uniquely identifies the *context*, a *Type* qualifying the context (*logical* or *physical*), and a set of *Activation Conditions* that represent the constraints on physical/logical conditions.

[2]Resource Description Framework (RDF) - http://www.w3.org/RDF/

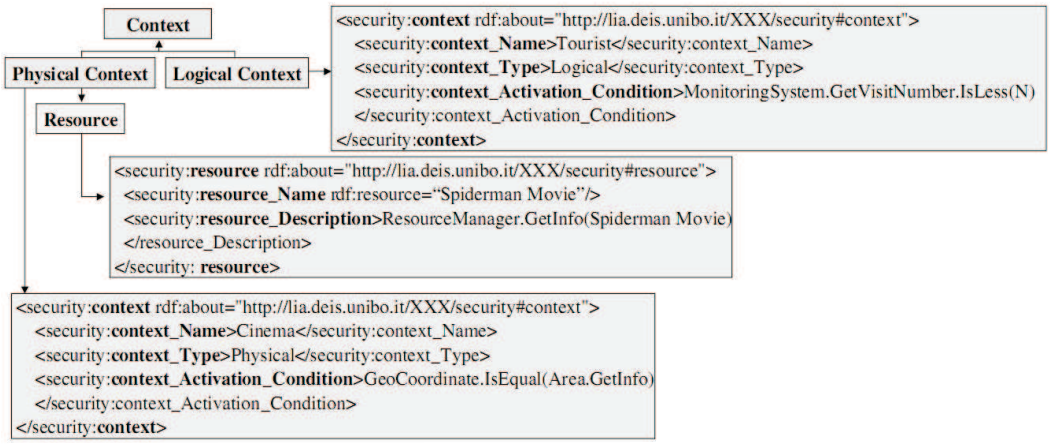


Figure 3.14: UbiCOSM Context Model [Corradi 2004a, Corradi 2004b].

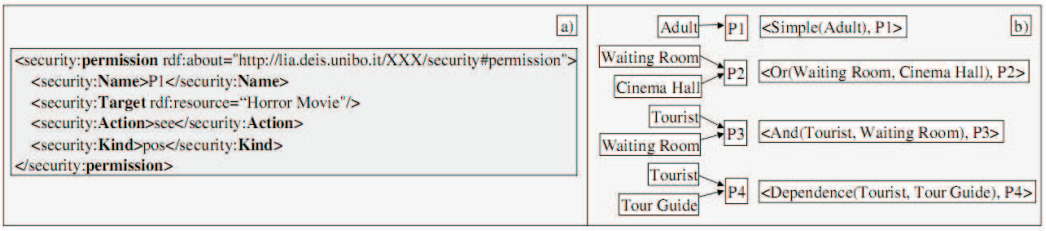


Figure 3.15: Permission and control policies [Corradi 2004a, Corradi 2004b].

UbiCOSM allows users to specify their security requirements at a high level of abstraction in terms of metadata. Metadata are declarative rules that describe both user/device/resource profiles and authorization policies. Metadata permit to separate security logic from security control and facilitate automated security reasoning.

Figure 3.15 illustrates an example of UbiCOSM permission that includes a *Name*, an *Action* specifying an allowed operation, a *Target* representing the resource on and a *Kind* representing the positive or negative meaning of that permission.

UbiCOSM access control policies are expressed in terms of tuples with the following format: $< association\_Name(context\_collection),\ permission >$. The first argument identifies a collection of one or more contexts to which associate the set of permission.

Hulsebosch *et al.* [Hulsebosch 2005] proposed a framework for context-sensitive access control (CSAC) to resources. The framework consists of setting up an access control architecture related to context-aware service provisioning, conceiving context-sensitive access control, and user authentication on the basis of context verification.

Figure 3.16 illustrates the CSAC infrastructure. They use location and velocity (including its direction) as contextual information. For privacy purposes, the true

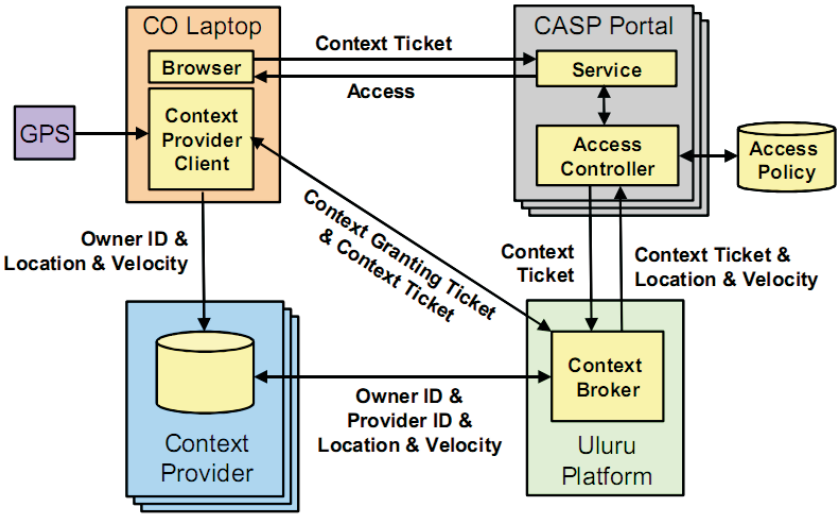


Figure 3.16: CSAC infrastructure [Hulsebosch 2005].

identity of the *context owner (CO)* is decoupled from his context information. Two types of tickets are used: the *Context Ticket* and the *Context Granting Ticket*. The *CO* and the *trusted context broker (CB)* know both the *Context Ticket* and *Context Granting Ticket*. The *Context Ticket* is issued by the *CB* to the *CO* and contains a pseudonym that the *CB* uses to link the *CO* and her *context provider (CP)* (i.e., it contains the *Owner ID* and *Context Provider ID*). The *Context Granting Ticket* is issued by the *CB* and instantiates the association between the *CO* and his *CP*. Only the CB has access to the *Context Granting Ticket* and uses this ticket to create new *Context Tickets* when the *CO* decides to use another context-controlled service. *CASP* only knows the *Context Ticket* and uses it for communication with the *CB*.

The location information of context providers (CP) is managed by the context broker (CB) service running on the Uluru Platform[3]. A Portal application was developed to demonstrate the use of context-sensitive access control (a Java web application). A location based access policy determines whether authorization is needed for the requested resource. Several types of access policies could be implemented. The access controller can define a geographical area that grants anybody who is inside this area access to a service. If the user leaves the area access to the service will be lost or denied.

3.3.2.2 ACA^2

Yokoyama *et al.* [Yokoyama 2006] have proposed an *Anonymous Context Aware Access Control Architecture* (ACA^2) based on an analogy to the public telephone service. Users can anonymously access services supported by their context through

[3] http://www.telin.nl/index.cfm?language=en&project=ULURU

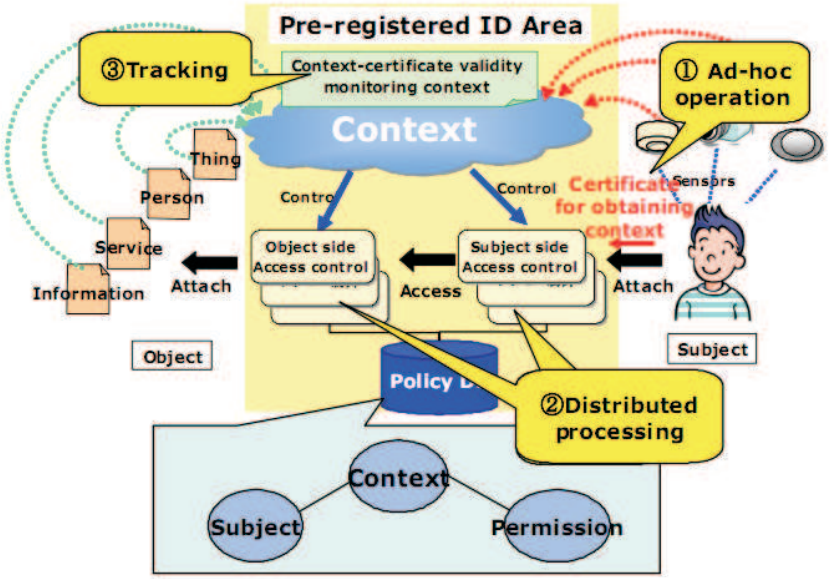


Figure 3.17: ACA^2 Architecture[Yokoyama 2006].

preregistered software components, named proxies.

Figure 3.17 illustrates an overview on that architecture. ACA^2 has three main features: (1) *ad-hoc operation*, (2) *revocation of access rights due to context changes*, and (3) *context certificates based on a streaming system*. The architecture is composed by the following elements:

- *Context Servers:* this group of servers is in charge of providing context information;
- *Proxies:* this set of pre-registered elements is in charge of providing terminal attachment points with the system;
- *Terminals:* devices used by subjects and objects;
- *Subject and Object:* Service consumer and service provider, respectively;
- *Sensors:* Context sources;
- *Message Service:* authentication and communication services in charge of delivering peer-to-peer messages between context servers, proxies, and sensors.

The message service of the ACA^2 is illustrated in Figure 3.18. *Subject* (i.e., user of *Subject_Terminal*) is the service user, *Object* (user of *Object_Terminal*) is the service provider, and *Sensors* are generators of information that establish the current context of both *Subject* and *Object*. *Subject_Proxy* and *Object_Proxy* are software components deployed with the system beforehand to represent the

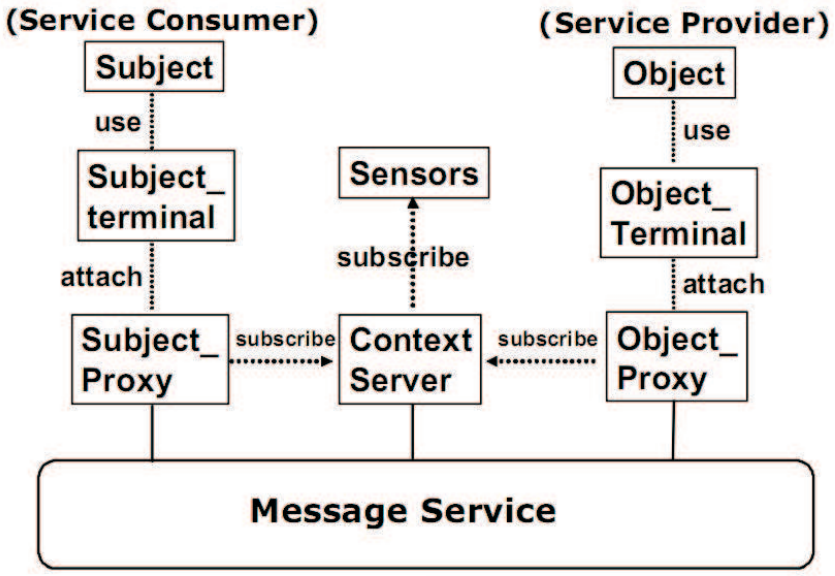


Figure 3.18: Message service of the ACA^2 [Yokoyama 2006].

Subject and *Object*, respectively. In order to give access on resources, Subject sends a context-collection-source certificate (it corresponds to coins in a public-telephone connection) to *Subject_Proxy*. A context-collection-source certificate describes how *Subject context* is collected, and states that a reliable third party can guarantee the legitimacy of that context. On receiving the $context - collection - source$ certificate, *Subject_Proxy* collects context based on the information described in that certificate. When a *Subject* requests access on a *Object* via *Subject_Proxy*, the *Object_Proxy* representing that *Object* notifies the *Subject_Proxy* about the context conditions for grating that access based on predefined policies.

3.3.2.3 Semantic-based Approach

Toninelli *et al.* [Toninelli 2006] have proposed a semantic access control approach based on context-aware policies. This solution treats context as a first-class principle for policy specification and adopts a hybrid approach to policy definition based on Description Logic (DL) ontologies and Logic Programming (LP) rules. This semantic-based approach allows description of contexts and associated policies at a high level of abstraction, enabling their classification and comparison. The authors adopted a resource-centric approach to context modeling: contexts are associated with the resources to be controlled and represent all and only those conditions that enable access to the resources.

Contexts act as intermediaries between the entities requesting access to resources and the set of operations that can be performed on these resources. Access control policies define for each context how to operate on the associated resource(s). A

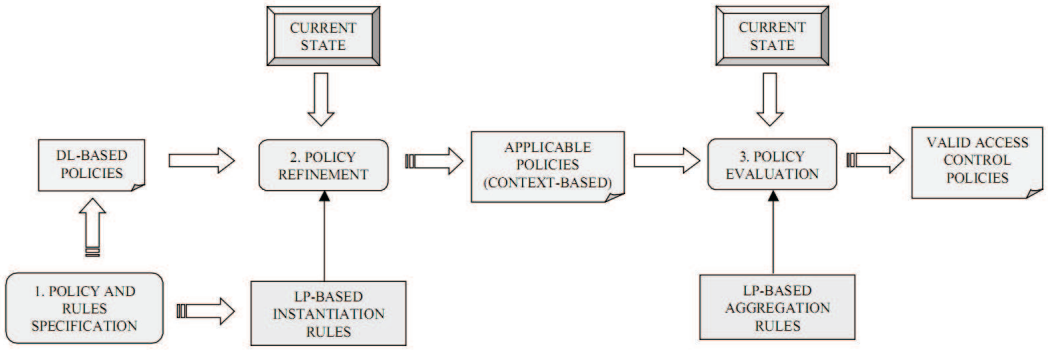


Figure 3.19: The semantic access control approach [Toninelli 2006].

protection context consists of all the characterizing information that is considered relevant for access control, logically organized in parts that describe the state of the resource associated with the protection context. The semantic access approach consists of three distinct phases (see Figure 3.19): *policy specification*, *policy refinement*, and *policy evaluation*.

In the policy specification phase resource administrators specify OWL-based policies representing ontological associations between actions and protection contexts ontology definitions. The *protection contexts* may have attribute values assigned to constants or may be variables. By adopting an object-oriented terminology, OWL-based policies can be viewed as policy types: they define the actions that are allowed in a set of context types. In order to be enforced in the real world, policy types need to be transformed into policy objects that associate sets of actions with specific instantiated contextual conditions. In the policy specification phase, administrators have to define aggregation and evaluation rules to enable effective enforcement and adaptation of OWL policies.

3.4 Conclusion

In this Chapter we presented some existing access control solutions for pervasive environments. We divided these solutions in two main groups: CAAC and CBAC approaches. Figure 3.20 illustrates a comparative table among the existing solutions described in this chapter. We consider as comparison parameters the following characteristics: base model, permission assignment, approach type, context-awareness approach, context model, context gathering approach, and policy implementation.

- *Base model:* this criterion classifies the solution according to the type of model used as basis for its definition;
- *Permission assignment:* it classifies the solution according to the central entity which acts as an interface between users and permissions;

Approach	Base Model	Permission Assignment	Approach type	Context-awareness approach	Context Model	Context gathering Approach	Policy implementation
ERBAC [Covington 2001]	RBAC, GRBAC	Role	Model	Environment Roles	-	Context Toolkit	GPDL
Access control for active spaces [Sampemane 2002]	RBAC	Role	Model	System, space, and application roles	-	-	Access Matrix
DRBAC [Zhang 2004]	RBAC	Role	Model	Dynamic user-role and role-permission assignments	-	Context Agent	Role and State machine
CAAC [Kim 2005]	RBAC	Role	Model	Dynamic user-role and role-permission assignments	-	Context-aware Agent	State Checking Matrix (SCM)
CS-RBAC [Kumar 2002]	RBAC	Role	Model	User, role, and object contexts	-	-	-
Context-Role RBAC [Park 2006]	RBAC	Role	Model	User and context roles	-	-	-
CRBAC [Li 2008]	RBAC	Role	Model	Context roles	Time, location, and trust	-	Specific algorithm
CAAC [Chae 2006]	RBAC	Role	Model	Role states (assign, disable, enable)	Location and time	-	-
CAAC [Neumann 2003]	RBAC	Role	Model	Context constraints, conditional permission	-	xorBAC	xorBAC
CABAC [Covington 2006]	-	Context	Model	User, Object, Environment, and Transaction attributes	-	-	-
UbiCOSM [Corradi 2004a]	-	Context	Middleware	Physical and logical context	-	CARMEN [Bellavista 2003]	RDF-based approach
CSAC [Hulsebosch 2005]	-	Context	Framework	Context and Context Granting ticket	Location and velocity	Ulutu Platform	CSAC framework
ACA^2 [Yokoyama 2006]	-	Context	Architecture	Context certificates	-	-	-
CBAC [Toninelli 2006]	-	Context	Semantic approach	Resource context	-	-	OWL-based approach

Figure 3.20: Comparative table among existing solutions.

- *Approach type:* it classifies the solutions according to the type of the proposed approach, which can be an access control model, a middleware/architecture in charge of enforcing permissions or only a conceptual description (e.g., semantic approach);
- *Context-awareness approach:* it describes the approach used to support contextual information for making access control decisions;
- *Context model:* it indicates the context model and context information explicitly used to support context-sensitive access control policies;
- *Context gathering approach:* this criterion describes the type of solution used for gathering relevant context information for their correspondent solutions;
- *Policy implementation:* it describes the technology used to implement and enforce context-sensitive access control policies.

We can observe that most solutions are based on the RBAC model and use Role as main entity to assign permission to the users. Moreover, there are few solutions [Covington 2006, Corradi 2004a, Corradi 2004b, Hulsebosch 2005, Yokoyama 2006, Toninelli 2006] that consider the context to the central concept for assigning permission to the users. Moreover, we observe that only the work of Covington *et al.* [Covington 2006] describes formally an context-based access control model based on the context attribute concept.

With regarding the context model, we observe that most existing solutions do not specify a model to represent relevant context information for making access control decisions. Often, they are limited to location and time information [Chae 2006, Li 2008, Hulsebosch 2005]. Moreover, they do not clearly define the set of observed entities and do not describe how context information used for decision making is gathered from the environment.

Finally, we can still observe that is increasingly evident the use of context information by access control systems in order to provide flexible access control mechanisms for pervasive environments. However, the quality, security, and the privacy of context information used for making context-based access control decisions are fundamental to the viability and reliability of such propositions. This occurs because contextual information has the same function than identity of users and roles in traditional access control solutions.

Chapter 4

Quality of Context

Résumé: *Ce chapitre présente les concepts relatifs à la qualité de l'information du contexte qui pourrons être utilisé pour activer des règles de contrôle d'accès, ainsi les façons de la modéliser et d'estimer. Nous avons identifiée l'importance de vérifier la qualité de l'information contextuelle utilisée pour la prise de décision. En fait, la vérification de la qualité du contexte lors de la vérification des politiques de contrôle d'accès pourra réduire la probabilité de faire des mauvaises décisions. Donc nous présentons les limitations des solutions existantes à propos de la modélisation, d'estimation et d'utilisation de l'information de qualité associés au contexte pour améliorer les opérations de gestion d'information contextuelles et, dans l'escope spécifique de ce travail, les mécanismes de sécurité sensibles au contexte.*

Contents

4.1 Introduction

Context-aware services and applications expect that context information used to adapt their functionalities is correct and reliable. Context-aware services face problems in using this information due the unawareness about the quality of information (QoC) [Buchholz 2003]. In fact, context information has an innate characteristic of imperfection and its quality is highly influenced by the way it is acquired [Henricksen 2004]. Generally, a PEC has various sources of context information (e.g., physical and logical sensors, user interfaces) distributed in the environment, which the quality of sensed information can differentiate among them.

In an inter-organizational pervasive environment, context information can be affected by many error source such as the following [Henricksen 2004, Krause 2005]:

- **Unavailability of context:** context information might be unavailable or unknown when making context-aware decisions. For instance, if a context-based access control system does not know the current location of users, it will be unable to enforce any location-based access control policy;
- **Inapplicability of context:** context information might be out-dated or inapplicable to the current situation. For example, an out-dated location information of a user may be inapplicable to offer useful content adapted to their situation (e.g., a list of nearby restaurants). Thus, context-aware applications could make erroneous content adaptation;
- **Physical restriction of sensors:** physical constraints and external influences, like temperature and humidity, might affect the accuracy of the sensed data;
- **Context refinement:** a wrong or inaccurate context information might be derived from other inaccurate low-level context information. For example, a service that uses the location information represented as a GPS coordinate to derive the real address of users might return wrong information if that coordinate is inaccurate;

- **Malicious context provider:** malicious context providers might distribute wrong context information to their context consumers, affecting directly any context-based decision made using that misinformation;
- **Ambiguity of context:** context information is often ambiguous. For example, a profiled context information about the location of a user named *Bob* (e.g., a information obtained from his agenda) certifies that *Bob* is working in his office, while his current location obtained from a GPS integrated to his personal phone certifies that *Bob* is in a restaurant next to the building of his work. In this case, what location information should the system consider true?
- **privacy of context:** privacy requirements of users with regard their context information might affect the detail level of disclosed context information. For example, a user might disclose his/her location but only with a reduced precision (e.g., the city name where he/she is located).

Henricksen *et al.* [Henricksen 2002] have described as characteristic of context to be *imperfect* the following assertions:

- Context information may be incorrect if fails to reflect the true state of the real entity that it describes;
- Context might be inconsistent if contains contradictory information (e.g., the current activity of a user is *working*, whereas his/her outdoor location indicates that he/she is in his/her home);
- Context can be incomplete if some aspects of the situation are unknown by the system.

Moreover, Henricksen *et al.* [Henricksen 2002] argued the need for taking into account quality dimensions when modeling context information [Indulska 2003]: *"... context models will need to specify a range of characteristics of context information, including temporal characteristics (freshness and histories), accuracy, resolution (granularity), [and] confidence in correctness of context information ..."*

On one hand, imperfect context information can influence directly decisions made by context-aware applications, leading to wrong conclusions. On the other hand, quality of context (QoC) could be used by context-aware systems in order to improve the following context processing steps:

- *Selection of context sources:* by using QoC information associated with the provided context information, context management systems will be able to select context providers and sensors. Only the context providers/sensors that are sensing context information that meet the quality requirements defined by the context management systems will be used;

- *Context conflict resolving:* in situations where the same type of context information characterizing an entity is available from two or more context providers/sensors, the quality parameters associated with that information can be used for conflict resolving. Moreover, QoC can be used to maintain the consistency of contextual information;
- *Improving context-based decision making:* QoC information can be used by context management systems and context consumers to improve their context-based decision making, reducing the likelihood of making a faulty decision.

In face of the exposed previously, we conclude that PCE needs modeling, evaluating, and management process of QoC assigned to context information. These requirements are more critical when the context information can be gathered from multi-domain pervasive environments. Moreover, in the same domain the users' personal devices can act actively in the construction of the context of his/her user.

Therefore, possessing the knowledge about the QoC plays an important aspect for using effectively context to make adaptation decisions. Before discussing in detail modeling, evaluating, and management process of QoC, we present in Section 4.2 some QoC definitions.

4.2 QoC Definitions

According to Buchholz *et al.* [Buchholz 2003], the concept *quality of context information* (QoC) differs from the terms *quality of services (QoS)* and *quality of devices (QoD)*. In fact, context information can exist without the presence of services and physical sensors in the pervasive environment. For example, a user might enter manually his/her location by filling in a user-friendly application interface. In this case, any service or physical sensor was used to gather automatically that context information. Thus, the quality aspects are inherent to the context information and do not characterize the process (e.g., services, sensors) used to obtain it.

One of the first papers about Quality of Context (QoC) has been written by Buchholdz *et al.* [Buchholz 2003]. They have defined the term *quality of context* and proposed the following set of QoC dimensions: *precision*, *probability of correctness*, *trust-worthiness*, *resolution*, and *up-to-dateness*. Buchholdz *et al.* defined QoC "*as any information that describes the quality of information that is used as context information. Thus, QoC refers to information and not to the process nor the hardware component that possibly provide the information*". In fact, two similar objects representing a type of context information (e.g., location) related to the same entity (e.g., user) obtained from the same context source can differ in terms of their precision, probability of correctness, trust-worthiness, up-to-dateness, etc.

Kim *et al.* [Kim 2006b] have extended that list of QoC dimensions, including accuracy, completeness, representation consistency, and access security. They took

into account concern of users. The characteristics of the used sensors, the situation of measurement, the values expressed by the context information object itself, and the granularity of the representation format have also been identified as information sources for determining the QoC [Krause 2005].

With the aim of relating quality with the value-added to the information for a context-sensitive application, Krause *et al.* [Krause 2005] have defined the following notion of QoC: *"Quality of Context (QoC) is any inherent information that describes context information and can be used to determine the worth of the information for a specific application. This includes information about the provisioning process the information has undergone (history, age), but not estimations about future provisioning steps it might run through".*

This definition distinguishes the QoC objectives from the application-dependent worth of a context information. In fact, QoC is used to estimate the worth of a context information for an application. This definition also impacts the relation between the concepts *QoC* and *QoS*. Quality agreements established between a context-aware service (CAS) and a context information service (CIS) about future provisioning steps of a context information concern to QoS of CIS, whereas information about the actual reached QoS in the provisioning steps could become part of the QoC provided. The QoC can be affected at several context provisioning steps.

4.2.1 Context-Aware Service provisioning model

The importance of evaluate the quality of context information increases when it is gathered from outside of the context-aware system domain. In order to present clearly this importance and the involvement of QoC in the context provisioning process, *Buchholdz et al.* [Buchholz 2003] propose a role model to classify context-aware services (CAS) and entities involved in the value chain of context providing in an inter-organizational environment. We mean by inter-organizational environments entities belonging to several organizations working together in order to provide context-aware services. Figure 4.1 presents this model, where actors could denote an individual, an organization, etc, offering and/or consuming services to/from other actors. Each actor autonomously operates and controls its own technical domain, e.g. sensors, network infrastructure, etc. The model has the following set of entities: context owners, CAS users, context providers, CAS providers, and CAS consumers.

An actor is able to adopt one or several roles, in which a role represents a certain field of activity and comprises a well-defined set of tasks where actors adopting this role should to fulfill. The main role is the CAS provider, which creates and deploys CAS offering and/or selling them to CAS customers. CAS customers interact with CAS providers in order to negotiate the terms of CAS usage on behalf of one or several CAS users. CAS providers obtain context information for service adaptation from context providers, which are usually operators of context sources, e.g. physical or logical sensors.

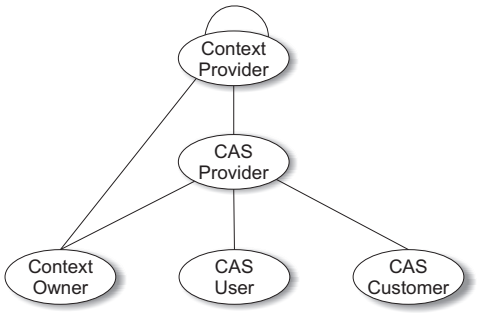


Figure 4.1: Role model for Context-Aware Services [Buchholz 2003].

For many CAS it would be useful that a CAS user has access to context information which is related to another actor (context owner). Context owners must be able to specify access restrictions regarding her context. CAS providers and context providers will interact to cooperatively provide CAS. In a multi-domain pervasive environment, the value chain of CAS provisioning is composed by the following phases [Buchholz 2003]:

- *Context sensing:* during the context gathering phase all relevant information necessary for determining an actual situation of the entities is sensed. For example, the location of users may be sensed by GPS receivers;
- *Context refinement and enrichment:* context refinement deals with the question of how to generate a usable high-level context information from sensed data offered by available context sources. In each step of context refinement, previously obtained low-level context information (e.g. sensed or derived context data) is used as parameter by refinement techniques like inference, filtering, combination, etc, in order to derive new high-level context information;
- *Context distribution:* high-level context information is delivered to CAS for further usage, which can operate in push or pull mode;
- *Context usage:* CAS uses distributed context information to modify its behavior or to adapt the content it provides.

Context providers of the CAS role model showed in Figure 4.1 perform the first three steps that compose the context provisioning phase. *Krause et al.* [Krause 2005] define a more simple CAS role model that is composed only by two entities: context-aware services (CAS) and context information services (CIS). Krause's definition of CAS represents together the CAS providers, CAS users, and CAS customers of the Buchholz's CAS role model, which is a service requesting and using context information. CIS performs the same task set of the context providers. As result, they do not consider in their model the context consumer role.

4.2.2 QoC Metrics

Buchholz *et al.* [Buchholz 2003] described the following definitions of QoC metrics considered as important for CAS (context-aware services):

- *Precision:* it describes how exactly the provided context information mirrors the reality. For instance, a location information sensed by a GPS receiver has a precision of about 4 meters. Precision might be specified on the same scale like the context information or a percentage could be used;
- *Probability of Correctness:* it denotes the probability that a piece of context information is correct. Their value is defined by the original source (e.g. context provider) in order to estimate how often the context information provided is unintentionally wrong resulting of internal problems (e.g. sensor errors). For example, a context information service that provides weather conditions have various temperature sensors distributed around the town. One of these sensors might fail providing wrong data, measuring 30 degrees Celsius while the correct value is 23 degrees Celsius;
- *Trust-worthiness*: trust-worthiness also describes how likely the provided information is correct. However, it is defined by context information services in order to rate the quality of the context providers that originally received the context information. For example, the context provider X sends the temperature of Grenoble that is 25 degrees Celsius to the context consumer Y. X states that this information has 100% of probability of correcteness. However, in the past Y received a wrong information from X. Thus, Y forwards this information to users with the remark that the source of the temperature is rather untrustworthy;
- *Resolution:* it denotes the spatial granularity in which an information was captured to represent a real world entity. For example, the temperature of a room provided by a context provider is 25 degrees Celsius. While this is on average true, in some location (e.g., near a heater) the temperature value is different. In this case, the context provider is incapable of offering temperature information at a finer spatial granularity;
- *Up-to-dateness:* it describes the age of context information. It can be specified by adding a time-stamp to context information. Very often, it would be more interesting to know how well a provided context information still accurately describes the actual situation.

As we discussed earlier, there are differences between quality of context (QoC), quality of services (QoS), and quality of devices (QoD). While QoC describes the quality of context information, QoS refers specifically to the quality of services provided. In fact, QoS is not equal to QoC because context information can exist

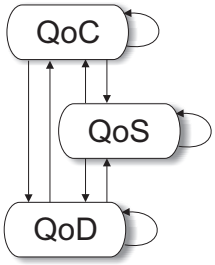


Figure 4.2: Relationships between QoC, QoS, and QoD [Buchholz 2003].

without services and QoS describes how well a service performs. In turn, services run on hardware components that possess a quality (QoD), which is any information about technical properties of devices and their capabilities. Thus, QoC, QoS, and QoD are unequal but they influence each other. For instance, the outdoor location of users can be determined using GPS receivers or, in a GSM network, taking the cell ID of the base station to which the mobile device is currently connected. Choosing one of these methods influences how quickly the mobile device can be located (QoS) and how precise the location information will be (QoC).

There are two possibilities in which one quality concept can affect another one, described in Figure 4.2: *bottom-up* and *top-down* approaches. In the first one a layer influences all layers above it (i.e., QoD affects QoS and QoC, while QoC influences only QoC). In the top-down approach a layer could have an impact on all lower layers normally by quality requirements that it poses. For example, QoC requirements like precision of 4 meters associated with the outdoor location of users affect directly on the quality of device (QoD) that should be used for sensing this context information. CAS providers requiring a high availability of a context affect the QoS parameters associated with the network technology used for transmitting this information, which should also affect the type of network equipments to be used (QoD).

High-level context information should very often be derived from one or several of other context attributes. In this case, the QoC of input data affects directly the quality of the resulting high-level context information. If a context provider wants to determine the weather conditions of a specific location at a given time, it could use the location of user obtained using a GPS receiver and the gathering time. In this case, the correctness and precision of location could affect the correctness of the derived weather condition information.

Quality of context can be used for defining QoC aggrements, determining context provider behavior, selecting appropriated context providers, adapting context refinement and distribution, and supporting fine-grained privacy policies [Buchholz 2003]. In order to allow automatic processing of QoC information it must be predictable which QoC dimensions a context information class can possibly be associated.

4.3 Modeling Quality of Context

There are some challenges in modeling QoC, such as expressiveness of QoC indicators and the need of distinguish QoC from contextual information. In fact, modeling QoC is very similar to modeling context information, where QoC information could be considered as context information itself. For instance, precision of location can be treated in the context providing process as a context information.

There is a wide range of possible forms to represent QoC information. QoC indicators like precision, up-to-dateness, probability of correctness are only common categories regarding to this diversity. In addition, we could describe their values in many ways. For instance, the accuracy of a context information could be described using the average error, the minimal error, the maximal error, and the probability distribution, which are expressed as relative or absolute values.

According to *Krause et al.* [Krause 2005], simple key-value-pairs with numeric values ranging between 0 to 1 are not expressive enough for open pervasive systems. The main problem is how to determine the worth of a number for an application, i.e., how to interpret their values. Moreover, QoC indicators which could be applied to a context information is highly dependent on the kind of context information. In the QoC modeling step it is necessary to verify dependencies among the QoC dimensions that will possibly compose the QoC model. Moreover, it is a modeling decision whether a context information class has a degree of freedom concerning a specific QoC information. For instance, “location" information could be associated with the “precision" as QoC information determined in the last gathering time, or it could be represented as “locationHighprecision" or “locationLowprecision" where the precision is included in the context information inherently.

Krause *et al.* [Krause 2005] have identified same QoC-values that could stamper context information objects: the characteristics of the sensor, the situation of the specific measurement, the value expressed by the context information object itself, and the granularity of the representation format [Krause 2005].

In order to help software engineers/developers in the task of modelling useful QoC for context-aware applications and services, Razzaque *et al.* [Razzaque 2005] proposed a QoC modeling process.

Figure 4.3 shows the steps of the methodology for quality contextual information modelling. The initial input is user’s and corresponding application’s requirements, and the final outcome of the modelling is the quality schema. Each step includes the input, the process, and the output. Figure 4.4 describes briefly each one of these steps.

Section 4.3.1 presents existing QoC models that can be used to represent and use QoC associated with context information.

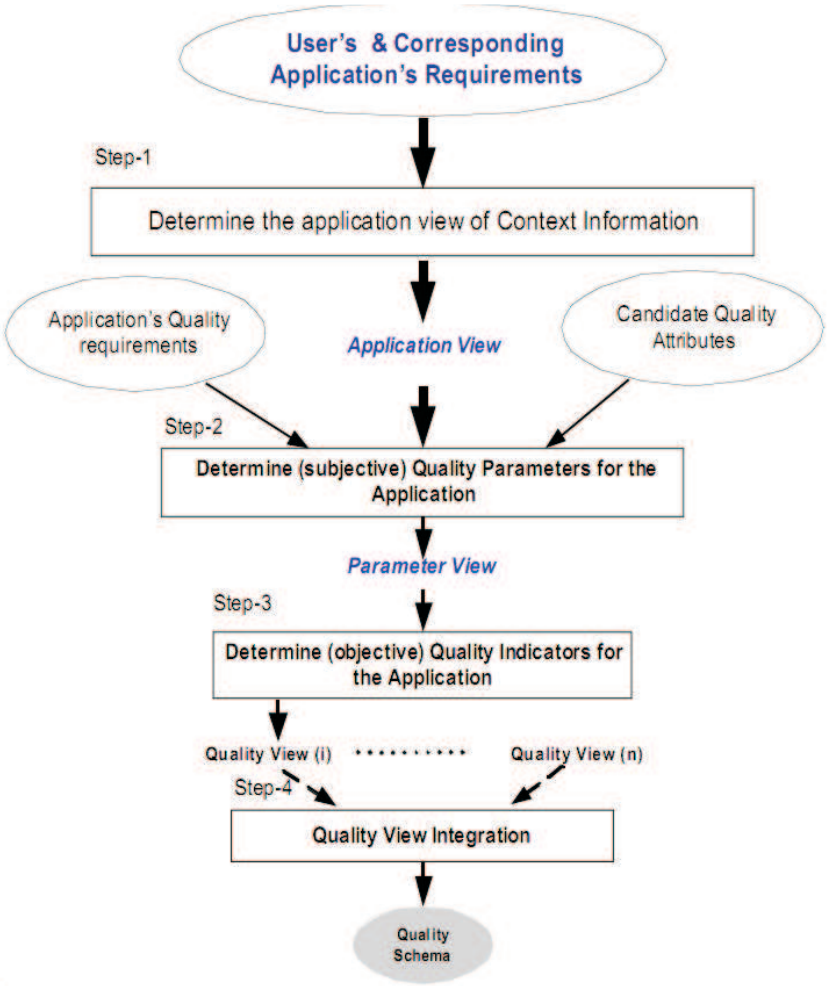


Figure 4.3: QoC modelling process [Razzaque 2005].

Step No.	Input	Output	Process
Step-1	User's and Corresponding Application's requirements	Application view	It embodies traditional context information modelling and objective is to extract and document application requirements of context information.
Step-2	Application view, application quality requirement, candidate quality attributes	Parameter view	It determines the quality parameters (like timeliness, reliability etc) to support information quality requirements.
Step-3	Parameter view(application view included quality parameters)	Quality view	It converts the subjective quality parameters into measurable characteristics or quality indicators (like timeliness to date, etc)
Step-4	Quality view/views	Quality schema	This involves the integration of quality indicators.

Figure 4.4: Steps of the QoC modelling process [Razzaque 2005].

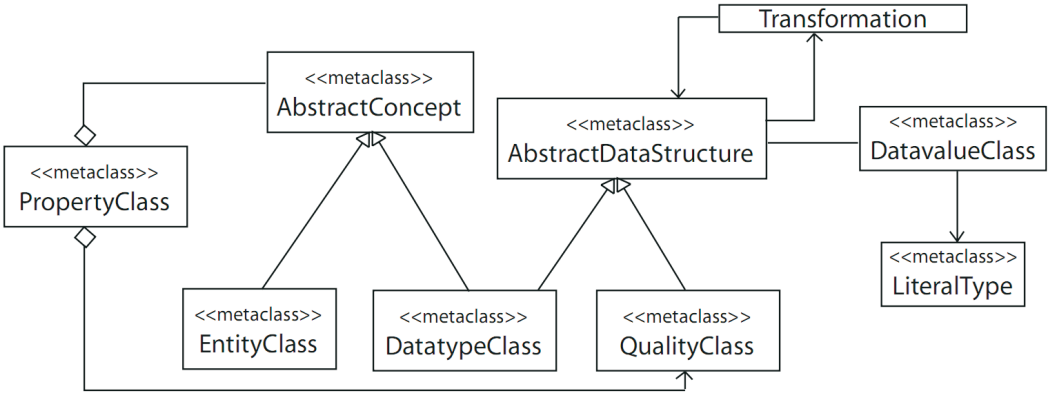


Figure 4.5: Context metamodel [Krause 2005].

4.3.1 Context Metamodel (CMM)

Krause *et al.* [Krause 2005] proposed a context metamodel (CMM) for modelling context information that include a base construct to represent quality aspects. Quality information is not of the same class as the context information value. Figure 4.5 describes the CMM, which QoC is represented by the metaclass *DatatypeClass*. QoC uses the same data constructs and transformation rules than context information. With the QoC and context information modelling, CMM meets the expressiveness requirements of QoC. CMM is as information model inside the Java-based CoCo infrastructure [Buchholz 2004].

4.3.2 QoC sources and QoC parameters

Manzoor *et al.* [Manzoor 2008] have classified QoC into two groups: *QoC source* and *QoC parameters*. According their point of view, QoC sources are quantities sensed from the environment or gathered from configuration system files. But QoC source values are not appropriate for use with an application, they are transformed to higher level values named QoC parameters, which are an suitable form for computational use. Then, context information is associated with these QoC parameters and is provided to context consumers.

Figure 4.6 shows the set of QoC sources and QoC parameters identified by the authors. In the following, we present briefly each QoC source concept, which are classified into *sensed* and *UserProfiled* information:

- *SourceLocation* is the location of the source of a context information;
- *InformationEntityLocation* is the location of the observed entity. *SourceLocation* and *InformationEntityLocation* represent the *space resolution* of a context information, that will be used to measure the *trust-worthiness* of a information;

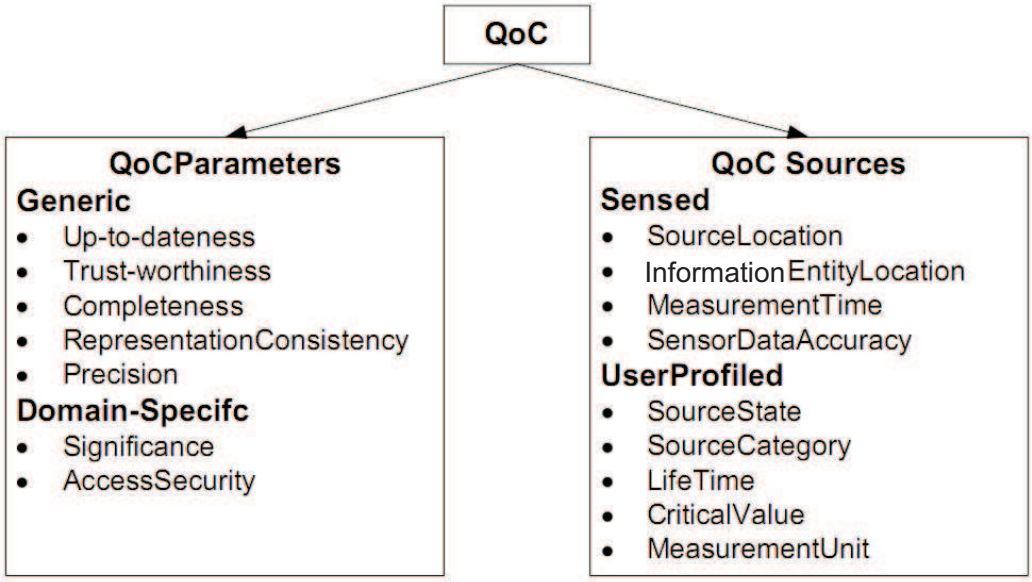


Figure 4.6: QoC classification[Manzoor 2008].

- *MeasurementTime* is the time at which context information is measured;
- *SensorDataAccuracy* is the accuracy with which a sensor can collect a context object;
- *SourceState* indicates whether the source of information is dynamic (e.g., a GPS embedded in a smartphone) or static (e.g., a temperature sensor fixed in a room);
- *SourceCategory* indicates the category of a context source (e.g., sensed, profiled, derived, and static);
- *LifeTime* is the period of time after which context information becomes obsolete and it is necessary to take its value again;
- *CriticalValue* of context information indicates that this information is crucial in a specific scenario;
- *MeasurementUnit* is used to describe the precision of context information (i.e., the unit used to describe the value of a context information).

Figure 4.7 illustrates the relationship among QoC sources and QoC parameters. The proposed set of QoC parameters are divided into generic and domain specific parameters. Generic QoC parameters are required by most context-sensitive applications, such as precision, trust, validity, representation consistency, and completeness. Domain specific QoC parameters are those parameters that are important for some specific application domains, such as *significance* and *accessSecurity*.

QoC parameter	*QoC sources used for evaluation*
Up-to-dateness	MeasurementTime, CurrentTime
trust-worthiness	SourceLocation, InformationEntityLocation, SensorDataAccuracy
completeness	Ratio of number of attributes filled to total number of attributes
significance	CriticalValue

Figure 4.7: QoC parameters and QoC sources [Manzoor 2008].

```
<xs:complexType name="QoC-Source">
    <xs:sequence>
        <xs:element name="InformationEntityLocation" type="ci:GeoLocationType"/>
        <xs:element name="Accuracy" type="ci:QoC-Decimal"/>
        <xs:element name="SourceLocation" type="ci:GeoLocationType"/>
    </xs:sequence>
    <xs:attribute name="SourceState" type="ci:SourceStateType" />
    <xs:attribute name="SourceCategory" type="ci:SourceCategoryType"/>
    <xs:attribute name="LifeTime" type="xs:duration"/>
    <xs:attribute name="CriticalValue" type="ci:CV_Integer"/>
    <xs:attribute name="MeasurementUnit" type="xs:string"/>
    <xs:attribute name="MeasurementTime" type="xs:dateTime"/>
</xs:complexType>
```

Figure 4.8: XML schema representation of QoC sources [Manzoor 2008].

```
<xs:complexType name ="QoCParametersType">
    <xs:sequence>
        <xs:element name="Up-to-dateness" type="ci:QoC-Decimal"/>
        <xs:element name="Trust-worthiness" type="ci:QoC-Decimal"/>
        <xs:element name="Completeness" type="ci:QoC-Decimal"/>
        <xs:element name="Significance" type="ci:QoC-Decimal"/>
    </xs:sequence>
</xs:complexType>
<xs:simpleType name="QoC-Decimal">
    <xs:restriction base="xs:decimal">
        <xs:minInclusive value ="0.0"/>
        <xs:maxExclusive value ="1.0"/>
    </xs:restriction>
</xs:simpleType>
```

Figure 4.9: XML schema representation of QoC parameters [Manzoor 2008].

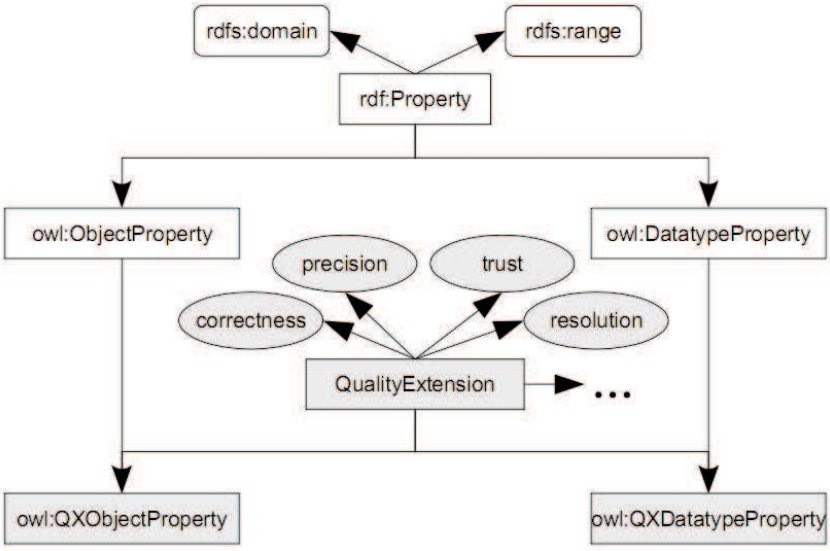


Figure 4.10: OWL language with QoC properties [Preuveneers 2006].

The authors use a XML-based representation for describing QoC source and QoC parameter values, which are illustrated in Figure 4.8 and Figure 4.9, respectively.

4.3.3 Ontology-based QoC modelling

Preuveneers *et al.* [Preuveneers 2006] proposed an extension to OWL documents for modelling context with support to quality information. In their approach, Quality of Context (QoC) parameters are modeled by means of two new property types: *QXObjectProperty* and *QXDatatypeProperty*. Figure 4.10 illustrates the proposed approach for extending OWL language.

Both property types inherit from the *DatatypeProperty* and *ObjectProperty OWL* language constructs, as well as from a self-defined class *QualityExtension*. *QualityExtension* models the following set of QoC metrics defined in [Buchholz 2003]: precision, correctness, trust, and resolution. They are represented as *DatatypeProperties* (see Figure 4.11). For example, Figure 4.12 illustrates a context information (temperature) associated with quality information described as QXDatatypeProperty.

Tang *et al.* [Tang 2007] also proposed an ontology-based approach for modelling quality of information. Unlike Preuveneers *et al.* [Preuveneers 2006] that proposed an extension to OWL language in order to represent QoC information, Tang *et al.* proposed an independent OWL-DL ontology for modelling QoC information.

Figure 4.13 illustrates the embedded property class into a common ontology-based context model. Property class replaces the function of owl property. Quality class is used by context information services to represent quality information with

```
<owl:Class rdf:ID="QualityExtension" />

<owl:DatatypeProperty rdf:about="#precision">
  <rdfs:domain rdf:resource="#QualityExtension" />
  <rdfs:range rdf:resource="&xsd;#int" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#correctness">
  <rdfs:domain rdf:resource="#QualityExtension" />
  <rdfs:range rdf:resource="&xsd;#int" />
</owl:DatatypeProperty>
...

<owl:Class rdf:ID="QXDatatypeProperty">
  <rdfs:subClassOf rdf:resource="&owl;#DatatypeProperty" />
  <rdfs:subClassOf rdf:resource="&owl;#QualityExtension" />
</owl:Class>
<owl:Class rdf:ID="QXObjectProperty">
  <rdfs:subClassOf rdf:resource="&owl;#ObjectProperty" />
  <rdfs:subClassOf rdf:resource="&owl;#QualityExtension" />
</owl:Class>
```

Figure 4.11: Serialization of OWL language [Preuveneers 2006].

```
<owl:Class rdf:ID="Sensor" />

<qx:QXDatatypeProperty rdf:about="#hasTemperature">
  <rdfs:domain rdf:resource="#Sensor" />
  <rdfs:range rdf:resource="&xsd;#int" />
  <qx:precision>95</qx:precision>
  <qx:correctness>100</qx:correctness>
  <qx:resolution>1</qx:resolution>
  ...
</qx:QXDatatypeProperty>
```

Figure 4.12: Example of context information[Preuveneers 2006].

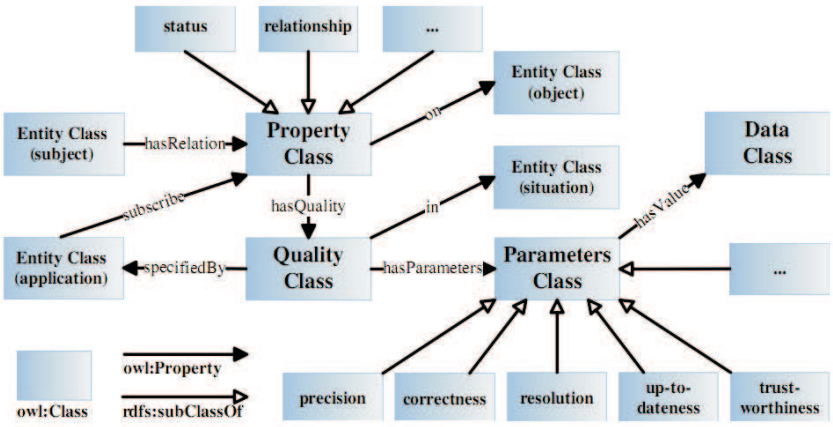


Figure 4.13: OWL-based QoC model [Tang 2007].

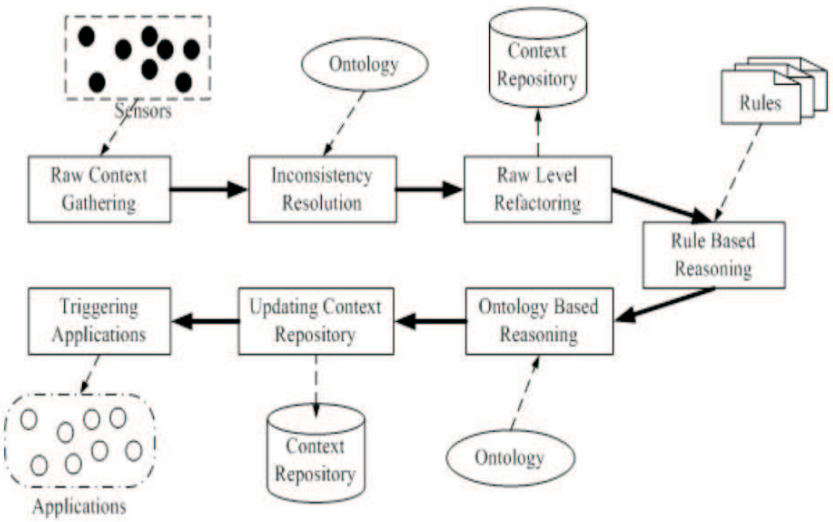


Figure 4.14: Context Management Process [Bu 2006].

different QoC-parameters. Parameters class, which defines the QoC-parameters, are associated with the current situation.

4.4 QoC management

In this section we present existing solutions of QoC management systems for pervasive environments. We describe how QoC is used internally to the context management operations.

4.4.1 Ontology-based QoC management

Bu *et al.* [Bu 2006] proposed an ontology-based approach of QoC management for inconsistency resolution based on three QoC parameters: delay time, context correctness probability, and context consistency probability. They use logic inference to process contexts. Figure 4.14 illustrates the context management process. The first step is the raw context gathering, in which raw contexts from various sensor sources are collected during a fixed short period. The second step is the inconsistency resolution.

They propose to resolve inconsistency among different raw contexts in this step because inconsistent raw contexts may lead to high-level inconsistent contexts that are more difficult to handle. Raw contexts are processed in a batch by batch manner instead of a piece by piece manner. Inconsistency in a batch of raw contexts should be cleaned prior to context reasoning so that the inconsistency of high-level contexts can be mitigated in certain degree.

The third step is the raw level refactoring, in which the context repository is updated with raw contexts, checking the dependency graphs and refactoring the ER graphs. Outdated or incorrect high-level contexts is deleted in this step. If this information is not removed in this step, it will result in serious inconsistency among contexts after reasoning. Then, the architecture apply rule-based reasoning and ontology-based reasoning based on the Jena API[1] in order to generate high-level contexts.

User-defined rules are in the form of Jena generic rules without negation and *or* operation. The two reasoners are configured as *traceable* in order to facilitate updating dependency graphs in context repository. After that, high-level contexts update the context repository and notify applications which register context triggers.

4.4.2 COSMOS

Abid et al. [Abid 2009] proposed the integration of QoC in the COSMOS framework (COntext entitieS coMpositiOn and Sharing) [Conan 2007]. They proposed three modes to transmit context information: two modes that deal with QoC information while one mode ignores QoC. The first mode consists in injecting QoC information as meta-data into the context information itself before sending it to upper layers. This mode is useful to filter context according to a particular policy. The second mode sends QoC information independently from any context information in a separate message. This mode enables to supervise the QoC of the system, with a limited overhead as only QoC data is computed and extracted. The third mode allows to transmit context information with standard child and/or parent components that cannot deal with QoC.

[1] http://jena.sourceforge.net/

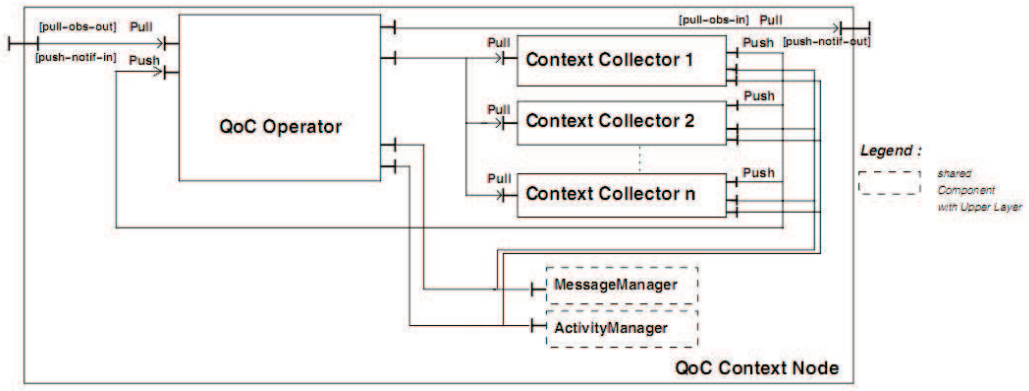


Figure 4.15: QoC Context Node [Abid 2009].

In the COSMO architecture, *QoC ContextNode* are in charge of evaluating QoC parameters values. *Contex Collector* collects raw sensed data from sensors and other information such as Measurement Time, Source Location, Data accuracy. These raw data sensors are transformed by QoC operators in order to deliver high-level QoC parameters.

- *QoC Operator:* it is responsible for extracting required data, computing QoC and supplying it to upper layers via the Message Manager (see Figure 4.15). Raw meta-data coming from different Context Collectors are analyzed by a *QoC Aware Operator* component which extracts relevant data and distributes them to *QoC Parameter Operator* components. Each *QoC Parameter Operator* computes a specific *QoC parameter* such as accuracy, precision, and up-to-dateness;

- *QoC Parameter Operator:* The choice of the nature of the *QoC Parameter Operator* component depends on what type of QoC the application needs and what computing methods are available (see Figure 4.16).

4.5 Evaluating Quality of Context

When we think about QoC evaluation, some questions should be answered: *When should QoC be evaluated? Who is in charge for evaluating it? Why is it necessary to do this? How we can use the resulting information?*

Quality of context can be evaluated in an objective or subjective way. For instance, the precision of location information could be objectively determined using a numeric value (e.g. precision of 4 meters). Whereas in a subjective way, the location information could have high or low quality for a certain purpose. In order

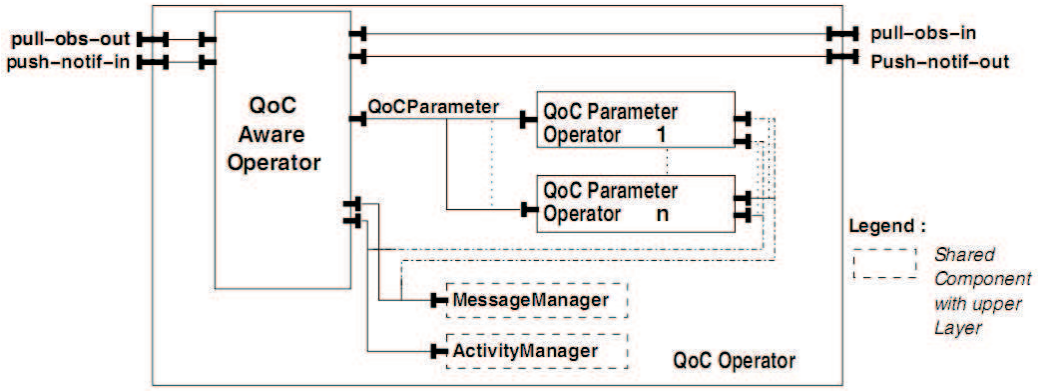


Figure 4.16: QoC Operator Architecture [Abid 2009].

to represent QoC in such a form that it can be easily used by context-aware services, it is necessary to quantify them. QoC metrics have already been proposed in the literature [Sheikh 2008, Buchholz 2003, Krause 2005, Manzoor 2008] for measuring the quality aspects of context information using quality attributes (i.e., QoC indicators). These QoC attributes are attached to the correspondent context information and will be communicated together with context information objects as meta information. Similarly to the QoC modeling, there are several ways to evaluate the quality of context information. For example, Manzoor *et al.* proposed in [Manzoor 2008] methods to evaluate the QoC as the worth of context information used by a geographical information system (GIS), in order to improve the rescue activities and the analysis of damages caused by floods. This approach provides context information enriched with QoC that is semantically described in a XML document, allowing context-aware services to know the quality of context information used without looking at its content.

Quality is a relative concept that should be measured against some well-defined standards. In order to represent QoC in such a form that it can be easily used by context-aware services and to keep the uniform representation of QoC measurement values in the context provisioning process, it is appropriate to measure QoC indicators[2] as a decimal which can have value in the range [0..1]. Maximum value 1 means that QoC indicator is in complete compliance to the given quality aspect while the minimum value 0 means total nonconformity to the aspect. Considering that QoC only depends on the piece of context it relates to, the QoC value is associated with the context management system and must not be modified during the information lifetime. This implies that all applications receive the same context information with the same QoC value. We present in this section existing work proposed in [Kim 2006b, Grossmann 2009, Manzoor 2008] for measuring the following QoC indicadors: Uncertainty (only for the location information), Inconsistency (only for the

[2]We refer QoC indicator as any information that can be used to describe an well-defined quality aspect associated with a context information. QoC indicator is sometimes referenced in the scientific community as QoC parameters

location information), Up-to-dateness, Trust-Worthiness, Completeness, Accuracy, and Significance.

4.5.1 Uncertainty

Grossmann *et al.* [Grossmann 2009] argues that imprecise sensors, like GPS, are the reason for uncertainty. Sensor values in general are not given exactly, but through a range of values. The authors proposed a method for measuring uncertainty for location information using as basis the probability density function (PDF)[3]. They assume that data providers (i.e., sensors) specify a normal PDF. The proposed method tries to express that, with some probability, context consumers are not sure or do not know the value.

Definition 5. *An uncertain position P is represented by a special PDF $p : \mathbb{R} \rightarrow \mathbb{R}_0^+$ with $0 \leq \int_{-\infty}^{\infty} dx \leq 1$. With the probability $1 - \int_{-\infty}^{\infty} p(x)dx$, the value is unknown (NULL).*

Besides representing uncertain positions, it is required a means for measuring how uncertain a information is. For this, they adopt the concept of differential entropy[4]. In order to be able to use this definition, they restrict the PDF to have a lower bound l and and upper bound u, with

$$p(x) = \left\{ \begin{array}{l} > 0, l \leq x \leq u \\ 0 : otherwise \end{array} \right\} \tag{4.1}$$

Definition 6. $u(P) = -\int_l^u \log_2 p(x)dx$ *is the uncertainty of position P.*

This definition restricts the form of the PDF and may not be adequate for cases where the probability for the value being NULL is greater than 0. However, we can only apply this definition to values directly retrieved from data providers.

4.5.2 Inconsistency

Grossmann *et al.* [Grossmann 2009] argues that inconsistency occurs when different data providers offer the same context information, e.g., different sensors measure the same context aspect. This leads to a finite number of alternatives for one value. For example, to measure the inconsistency of two positions they use the arithmetic mean of the smallest possible distance and the largest possible distance between the positions, as defined below:

Definition 7. *The smallest and largest possible distance between two positions P_1 and P_2 are $d_{min} = max(0, max(l_2 - u_1, l_1 - u_2))$, $d_{max} = max(u_1, u_2) - min(l_1, l_2)$. The inconsistency of the two positions is*

[3] http://mathworld.wolfram.com/ProbabilityDensityFunction.html
[4] http://planetmath.org/encyclopedia/DifferentialEntropy.html

$$i(P_1, P_2) = \frac{d_{min} + d_{max}}{2} \tag{4.2}$$

4.5.3 Up-to-dateness

Before presenting the existing approaches to measure the up-to-dateness, we discuss some related concepts proposed in the literature. Mazoor *et al.* [Manzoor 2008] define the up-to-dateness as "*the degree of rationalism to use a context object for specific application at a given time*". Sheikh *et al.* [Sheikh 2008] use the term $freshness$ in place of $up - to - dateness$, defining it as "*the time that elapses between the determination of a context information and its delivery to a requester*". According to Buchholz *et al.* [Buchholz 2003] and Kim *et al.* [Kim 2006b], the Sheikh's freshness definition is similar to the *up-to-dateness* concept, which is define as the age of context information.

Sheikh *et al.* [Sheikh 2008] use yet an other QoC indicator named *temporal resolution*, which is defined as "*the period of time to which a single instance of context information is applicable*". For example, the period of time between two collected context objects is the temporal resolution of that information. In fact, the temporal resolution has the same sense that the QoC parameter *Lifetime* used by *Manzoor et al.* [Manzoor 2008] for measuring the *up-to-dateness*. At last, Bu *et al.* [Bu 2006] use an other concept called delay time, which is defined as "*the time interval between the time when the situation happens in real world and the time when the situation is recognized in the system*".

In face of these definitions, we conclude that *freshness* and *up-to-dateness* are different concepts once *up-to-dateness* has a larger sense than *freshness*. On one hand, as stated by Mazoor *et al.* [Manzoor 2008], up-to-dateness describes how current the context information is for an entity at a given time for making context-based decisions. On the other hand, *freshness* of a context object is the *age* of this information, which is a value used for determining the *up-to-dateness* of that context object.

To the best of our knowledge, in the existing work [Kim 2006b, Sheikh 2008, Manzoor 2008, Grossmann 2009] that propose QoC measuring methods there is only one solution for evaluating up-to-dateness. Sheikh *et al.* [Sheikh 2008] describe how to evaluate and use *freshness*, but as we have discussed previously, it is an information used for measuring the *up-to-dateness*, then it is not considered as a QoC indicator. *Manzoor et al.* [Manzoor 2008] proposed to take into account the *Age* (i.e., the freshness) of context information and the *Lifetime* of that context information in order to calculate the value of *up-to-dateness*. Figure 4.17 gives a pictorial depiction of all concepts related with this QoC indicator.

According to the approach of Mazoor *et al.* [Manzoor 2008] for measuring the up-to-dateness, the age of context information object $CxtObj$, $Age(CxtObj)$, is cal-

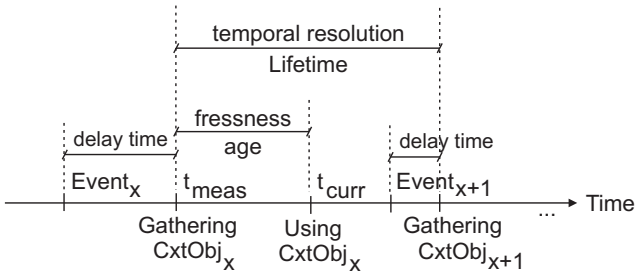


Figure 4.17: Up-to-dateness, Fressness, Age, Delay time, and Lifetime concepts.

culated by taking the difference between the system current time in the moment where this *CxtObj* is used, t_{curr}, and the measurement time of that context object, $t_{meas}(CxtObj)$, as shown by Equation 4.3:

$$Age(CxtObj) = t_{curr} - t_{meas}(CxtObj) \tag{4.3}$$

Then, the up-to-dateness of the context object $CxtObj$, $U(CxtObj)$, is calculated by Equation 4.4:

$$U(CxtObj) = \left\{ \begin{array}{l} 1 - \frac{Age(CxtObj)}{Lifetime(CxtObj)} : if\ Age(CxtObj) < Lifetime(CxtObj) \\ 0 : otherwise \end{array} \right\} \tag{4.4}$$

Therefore, the value of up-to-dateness and hence the validity of context object *CxtObj* decrease as the age of that context object increases. The QoC parameter Lifetime is determined taking into account specific requirement inherent to each context consumer and it can change depending on the type of context information. In a real implementation scenario, the QoC parameter Lifetime could be described, for example, using QoC configuration files that could be defined globally by administrators of context provisioning infra-structures, or locally by administrators and users of context-aware services.

4.5.4 Trust-Worthiness

Buchholz *et al.* [Buchholz 2003] introduce a contraction when they define trustworthiness. According to their definition, trustworthiness is used by the context provider to rate the quality of the actor from which the context provider originally received the context information. This definition is clearly opposed to the first definition of QoC, which states that QoC is about the information and not the process nor hardware component that provides the information.

Therefore, trust-worthiness indicates the belief that we have in the correctness of a context information. We identified two approaches for measuring this indicator: i)

measuring the belief that we have directly on the context information; ii) measuring the truth that the context consumers have in the entity that provided the context information.

Manzoor *et al.* [Manzoor 2008] proposed a measuring method based on the first approach. They argue that trust-worthiness of a context object is highly affected by its spatial resolution, i.e., the distance between the sensor and the entity described by the context object. The farther the distance of sensor from the entity the more will be the doubt in the correctness of information presented by that context object. Along with the space resolution, the accuracy with which the sensor collects context information also impacts the trust-worthiness of that information. Let the accuracy of the sensor data be δ. The trust-worthiness, $T(CxtObj)$, of context object $CxtObj$ is defined by Equation 4.5 [Manzoor 2008]:

$$T(CxtObj) = \left\{ \begin{array}{l} 1 - \frac{d(S,E)}{d_{max}} \times \delta : if\ d(S,E)\ <\ d_{max} \\ 0 : otherwise \end{array} \right\} \tag{4.5}$$

where $d(S,E)$ is the distance between the sensor that gathered the $CxtObj$ about the entity E and d_{max} is the maximum distance for which we can trust on the observation of this sensor. Every type of sensor will have different value for d_{max}. For example, d_{max} value for a satellite capturing the photos from the space will be a lot more than the camera held by someone to take photos in the field. Accuracy of a sensor, δ, is measured on the basis of a statistical estimation method presented in [Kim 2006b]. Trust-worthiness is useful in situations when we have more than one context object representing the same entity. There will be more confidence in the context object collected by the sensor that has a higher value of trust-worthiness.

In [Grossmann 2009], Grossmann *et al.* proposed a solution based on the second approach for measuring the trust-worthiness. They consider context providers to be differently reliable. The reliability of a context provider cannot be constituted globally, because it depends on the user and its preferences. They model trust as a triple *(belief, disbelief,ignorance)*, where the three values are from the interval $[0,1]$ and their sum is 1. In the following we present a simplified version, where the *disbelief* value is always 0. In this case, it is sufficient to specify the belief value b (the trust level in the context provider) and the ignorance value is $1-b$.

Definition 8. *The trust value of context provider i is given by $b(i), b : \mathbb{N} \rightarrow [0,1]$*

4.5.5 Accuracy

Kim *et al.* [Kim 2006b] define accuracy as the degree to which a context information is correct and reliable. It is difficult to know the correct value of information (i.e., the true value) without a mechanism of verification, such as validation performed by humans. Thus, they estimated the confidence interval of context information generated by a sensor using a statistical estimation method. Then, a context information

can be said to be accurate if the value is within the confidence interval. They use the RMSE(root mean squared error)[5] to calculate an error. In case of continuous data they use interval estimation method for confidence upper limit and confidence lower limit. The error of a sensor s_x, $RMSE(s_i)$, is defined as bellow:

$$RMSE(s_i) = \sqrt{\frac{1}{N} \times \left[\sum_{j=1}^{N} (x_j - \bar{x})^2\right]} \tag{4.6}$$

where N is the total of observed data values, x_j is the observed data value and $\bar{x}$ is the average of the observed data values. A confidence interval that estimate the true value of a sensor s_i, $TV(s_i)$, is calculated as following:

$$TV(s_i) = \left(\bar{x} - t\,(v, \alpha) \times \frac{\sqrt{V}}{\sqrt{N}}, \bar{x} + t\,(v, \alpha) \times \frac{\sqrt{V}}{\sqrt{N}}\right) \tag{4.7}$$

where the t-distribution, with $v = N - 1$ degrees of freedom, is given by the equation 4.7, and V is an error.

4.5.6 Completeness

Kim *et al.* [Kim 2006b] define completeness as "*the degree to which available context information are present*". It means that the nearer the value of completeness is 1, the more the available information is. Manzoor *et al.* [Manzoor 2008] define this QoC indicator as "*the quantity of information that is provided by a context object*", which is different from the previous definition. In [Kim 2006b] completeness has been computed as the ratio of the number of attributes available (AD) in the context management system to the total number of attribute reading (TD). This concept is measured by the following equation:

$$C(CxtObj) = \frac{AD}{TD} \tag{4.8}$$

where C(CxtObj) is the completeness of a context information, AD is the number of available output values and TD the total number of output values registered in the context management system.

Manzoor *et al.* [Manzoor 2008] have enhanced this concept by using the weights for different attributes, as all attributes of a context object do not have the same importance. They define the completeness of a context object as the ratio of the sum of the weights of available attributes of a context object to the sum of the

[5] http://www.math-interactive.com/Products/CalGraph/Help/Fit_Curve_to_Data/Root_Mean_Squared_Error.htm

weights of all the attributes of that context object. Completeness of context object $CxtObj$, $C(CxtObj)$, is evaluated by Equation 4.9:

$$C(CxtObj) = \frac{\sum_{j=0}^{m} w_j(CxtObj)}{\sum_{i=0}^{n} w_i(CxtObj)} \tag{4.9}$$

where m is the number of the attributes of context object $CxtObj$ that have been assigned a value and $w_j(CxtObj)$ represents the weight of the jth attribute of $CxtObj$ that has been assigned a value. Similarly, n is the total number of the attributes of context object $CxtObj$ and $w_i(CxtObj)$ represents the weight of the ith attribute of $CxtObj$. The value of completeness will be maximum, i.e., 1 if $n = m$. It means that all the attributes of context object $CxtObj$ have been assigned a value.

4.5.7 Significance

Manzoor *et al.* [Manzoor 2008] define this quality indicator as *"the worth or the preciousness of context information in a specific situation"*. A context object having this information gets a higher value of significance so that it will get immediate attention by the context management system and context consumer. Significance of context object $CxtObj$, $S(CxtObj)$, is evaluated by Equation 4.10 described bellow:

$$S(CxtObj) = \frac{CV(CxtObj)}{CV_{max}(CxtObj)} \tag{4.10}$$

where $CV(CxtObj)$ is the critical value of the context object $CxtObj$. This information will be gathered from a situation configuration file. This configuration file will have the information about the critical values of each type of concept in the context model for a specific situation. $CV_{max}(CxtObj)$ is the maximum critical value that can be assigned to a context object of the type that is represented by $CxtObj$.

4.6 Using QoC Information

In this section we describe how QoC can be used by application and context management framework in order to improve its functionalities.

4.6.1 Conflict resolving

Manzoor *et al.* [Manzoor 2009b, Manzoor 2009a] proposed the use of QoC information for conflict resolving. In pervasive environments conflicts can take place at

different layers of context management systems, such as context acquisition, context processing, context distribution, and application level. We describe in the following each one of these layers with examples of conflicting situations:

- *Context Acquisition:* in pervasive environments the volume of data generated by sensors makes the analysis of context impossible for a human. Sensor data may differ from each other considering the frequency of updating context, the capability of a sensor to collect the context of an entity, the accuracy of a method that is used by sensors, representation format, and the cost of gathering that context information. For example, location information of a mobile user can be gathered using GPS and GSM methods. Problems arise due to the mobility of sensors along with entities in pervasive environments. Sensors are not permanently ranked to collect the context of a particular entity. Thus, there is a need for a strategy that can dynamically decide which sensor is more reliable to collect the context of a certain entity at some specific time. QoC that have been dynamically evaluated from the information about the source of context can be used to resolve the conflicts in such situation;

- *Context processing:* in the processing layer, high level context is extracted from low level sensor data. Sensor data may not be understandable when presented directly to context consumers. It needs to be altered, fused, correlated, and translated to extract the higher level context data and detect the emergent events. QoC that provide information about up-to-dateness, trustworthiness, significance, and completeness can be used to make the reasoning on data to resolve conflicts;

- *Context Distribution:* The high mobility of sensors, unreliable wireless connections, and the nature of tasks in pervasive environments result in the acquisition of a lot of redundant and conflicting context. This redundant and conflicting context not only results in the wastage of scare resources but also can lead to undesired behavior of context-aware applications. Simple conflict resolving policies, such as drop first, drop all, can result in deleting some valuable information. In critical situation, such as a context-aware ubiquitous home for patients and healthcare applications, loss of information can result in severe situations for the people using it. Decision can better be made to discard or keep a context object on the basis of policies defined using QoC information;

- *Application:* Context-aware applications use context information to adapt their behavior to user needs and changes in the environment. If conflicts are not resolved in context information at the earlier stages, applications that take actions on the basis of that context information get in conflict while making decisions. Context-aware applications can also get in conflicts due to different priorities set by users. Information about the up-to-dateness, trustworthiness,

completeness, and significance of context information make it easy to resolve conflicts and make decisions on the basis of that context information.

Taking into account these situations, the authors propose QoC-based policies for conflict resolving. In the following, we describe each type of these policies:

- *Up-to-Dateness Based Policy:* Up-to-dateness indicates the degree of rationalism to use a context object at a specific instance of time. The up-to-dateness of a context object is calculated as the ratio between the *age* of that context object and the *lifetime* of the type of context information contained by that context object. This metric can be useful for resolving conflicts in the context objects that change their values very rapidly, e.g., the location of a fast moving vehicle. In this case, it will be more suitable to use the context object with the highest value of up-to-dateness. Whereas, up-to-dateness will not have a significant role in the case of conflicts in static information that have been profiled in the system, e.g., information about the smartphone capabilities;

- *Trustworthiness Based Policy:* Trustworthiness is the degree of the suitability of a sensor to collect the context of a specific type. The trustworthiness of a context object is calculated on the concept of space resolution and accuracy of sensor. This concept is particularly useful in resolving the conflict when we have more than one sensor collecting the context of same entity or event. For example, we have temperature sensors at different places in the living room of a smart home that is built to provide comfortable life to old people. The sensors that are installed near the electric radiator heater will be sending the higher value of the temperature of living room as compared to the sensors in the other places in the living room. To provide a comfortable temperature in the room it is more relying on the readings of the sensors that are closer to the sitting area than the sensors in the far off corners of the living room and sensor near the radiator;

- *Completeness Based Policy* The completeness of context information indicates that all the aspects of context information have been presented by a context object. The completeness of a context object is evaluated as the ratio of the sum of the weights of available attributes of context object to the sum of the weights of the total number of attributes of the context object. Completeness of a context object is particularly important to get the complete picture of the current situation of the real world. According to this policy decision is made on the basis of that context object which has more complete information about the current situation;

- *Significance Based Policy:* Significance measures the worth or preciousness of a context object. It is particularly important to mention this metric when there is a context object of high critical value. For example, if smoke sensors detect heavy smoke in the bedroom, it will be an information of high significance.

This metric can be used to generate events that need prompt actions from the applications. Applications can specify that the context objects with high values of significance should be reported on a priority basis.

From the above mentioned fundamental policies, policies can also be defined based on two or more QoC parameters depending on the requirements of a particular application. For example, a policy can also be defined by combining QoC parameters, such as up-to-dateness and trustworthiness. In such policies an average value of the mentioned QoC parameters is used to make decisions. For example, if a context aggregator uses a combining policy with the threshold value of 0.8, then all the context objects having an average of the value of up-to-dateness and the value of trustworthiness of more than 0.8 will be selected. Users of conflict resolving policies set threshold values according to their requirements considering the perspective of the use of context information.

4.6.2 Improving UI (User Interface)

Muhlhauser *et al.* [Mühlhäuser 2009] proposed the use of QoC information for improving context-aware user interfaces. For example, QoC information can be provided for cognitive user contexts: i) algorithms used to infer information from the user's behavior usually return a confidence value along with the inferred information. This value can be considered equivalent to the probability of correctness; ii) trustworthiness can be derived in a similar way as for other context sources, e.g., by measuring how often a context source returned data that proved to be useful; iii) resolution may reflect the population from which the information was derived (single user, users group, all potential users, etc.); iv) the up-to-dateness can be reflected as the time of the last user model update.

The probabilistic nature leads to inherent uncertainty of context information. This has to be considered when using it for improving the UI. E.g., any high-impact or irreversible action should be executed automatically if it relies on uncertain sensor data. Three (non exclusive) meta-concepts are proposed to cope with uncertainty at the UI:

- *Inform and mediate:* inform the user about uncertain context and let her confirm or correct the data;
- *Make multiple suggestions:* derive a weighted list of suggestions from context, not just a single one, and present them to the user for selection;
- *Adapt behavior:* consider the level of uncertainty for adjusting whether and how an action is executed and suggestions are made.

The authors distinguish three main ways to convey context quality at the UI:

- *Numerical:* a numeric value (number) is used to represent the certainty in a given action / suggestion;
- *Symbolic:* different icons represent different levels of uncertainty. For example, we can use caricatures to indicate the confidence in a given prediction;
- *Gradual graphical attributes:* use a graphical attribute like color or line thickness to convey the certainty in the context quality. For example, we can use different shades of green to mark which interaction element the user will most probably use next. Light green thereby means that the system is not very confident in the prediction whereas dark green stands for high confidence.

While it is important to convey the context quality, a drawback lies in the user's increased cognitive load. Therefore it is important that the context quality can be easily perceived and that it is conveyed in an unobtrusive manor. To this end, the authors used gradual graphical attributes like colors. Humans are used to this kind of quality depiction from everyday experience, making it intuitive to grasp. Authors advocate the presentation of one quality property at a time as opposed to various quality dimensions for the sake of simplicity.

4.6.3 Improving activity recognition systems

Villalonga *et al.* [Villalonga 2009] proposed the use of QoC for improving activity recognition system. Activity recognition in wearable computing tackles on-body systems of limited size which differ considerably from the higher level view of context aware applications and large scale context frameworks. However, user activity is a valuable piece of context and is worth to be made available to any application through context frameworks.

By connecting to context frameworks the activity recognition systems could obtain additional data from environmental sensors and even incorporate them into the recognition chain. Integrating the two systems leads to the question of how QoC is calculated in function of the performance metrics, i.e., how QoC metrics often derived from the machine learning field are mapped into the abstract QoC parameters and and how QoC should be extended to be useful in this area. In the following we describe the mapping of QoC in activity recognition system:

- *Offline Performance Metrics:* Accuracy as part of the QoC is one of the most relevant parameters as it gives an idea of the relation between the context value and reality. In wearable computing, the corresponding metrics are the offline performance metrics, i.e., accuracy, confusion matrix, precision, recall, and specificity. Even if *accuracy* is used in both domains, the concept is different since in activity recognition it is a statistical value saying how often the recognized class matches real class. The authors suggest the use of values

on the diagonal of the confusion matrix as the quantification for the accuracy parameter in the QoC of the recognized class;

- *Online Performance Metrics:* Activity recognition systems usually operate online. The continuous recognition performance metrics are particularly important to quantify the errors of the classification for this system. The authors suggest extending QoC to include performance metrics (insertions, substitutions, deletion, merge, fragmentation, overfill and underfill);

- *Cost of Context and Power Consumption:* Context frameworks do not consider resource consumption when delivering or collecting context as the only goal is to deliver high quality context. In wearable computing, however, devices are running on batteries and only limited power is available. Thus it is important to consider how much power is invested into the activity recognition. Power consumption can be traded-off for accuracy and can be used as performance metric of an activity recognition system. However, it is not a QoC measure as it does not indicate how the context represents the real world, but only informs about the cost to calculate this context value. Cost of Context is therefore a new concept, which if defined as a parameter associated to the context that indicates the resource consumption used to measure or calculate the piece of context information;

- *Delay Time and Latency:* In a large scale framework, it is the time to find the appropriate context source, processing the context information using, e.g., ontology reasoning. In wearable computing scenarios involving human assistance, the response time of an application is crucial for the acceptance by the user. In some cases, feedback must be delivered within milliseconds for meaningful interaction. Therefore latency is another vital metric in human activity recognition and needs to be integrated into QoC. There is a clear matching between the latency and the delay time parameter of the QoC. The authors recommend to use the calculated latency as quantification for the delay time parameter of the QoC measure.

4.7 Conclusion

This chapter present some existing work that propose QoC modeling, measuring, and management approaches. Figure 4.18 illustrates a comparative table among these solutions according to the following characteristics: i) proposed set of QoC indicators and QoC parameters; ii) QoC measuring methods; iii) QoC modeling approach; iv) Context Management Framework; and v) QoC purpose.

- *QoC indicators*: Buchholz *et al.* [Buchholz 2003], Kim *et al.* [Kim 2006b], Bu *et al.* [Bu 2006], Manzoor *et al.* [Manzoor 2008], Sheikh *et al.* [Sheikh 2008]

Approach	QoC indicators	QoC parameters	QoC measuring methods	QoC modeling approach	Context Management Framework	QoC purpose
Buchholz et al. [Buchholz 2003]	Probability of correctness, trust-worthiness, Resolution, Up-to-dateness	No	No	No	No	General
Krause et al. [Krause 2005]	No	No	No	CMM	Yes (CoCo)	General
Preuveneers et al. [Preuveneers 2006]	No	No	No	OWL	No	Uncertainty handling
Tang et al. [Tang 2007]	No	No	No	OWL-DL	Yes	Context-aware File system
Manzoor et al. [Manzoor 2008]	Significance	Yes	Up-to-dateness, trust-worthiness, completeness, significance	XML schema	Yes	Rescue situations
Sheikh et al. [Sheikh 2008]	Precision, Freshness, Spatial and Temporal Resolution, Probability of correctness	No	No	No	AWARENESS middleware	Privacy protection
Razzaque et al. [Razzaque 2008]	No	No	No	Process for modeling QoC	No	General
Kim et al. [Kim 2006]	Accuracy, Completeness, Representation-Consistency, AccessSecurity, Up-to-dateness	No	Accuracy, Completeness, Aggregation (average)	No	No	General
Bu et al. [Bu 2006]	Delay time, Probability of correctness, Probability of Consistence	No	No	OWL	Yes	Inconsistency resolution
Abid et al. [Abid 2009]	No	No	No	No	COSMOS	General
Grossmann et al. [Grossmann 2009]	No	No	Uncertainty, Inconsistency, Trust	No	NEXUS	General
Manzoor et al. [Manzoor 2009, 2009b]	No	No	No	XML schema	Yes	Conflict resolving
Muhlhauser et al. [Muhlhauser 2009]	No	No	No	No	No	UI improvements
Villalonga et al. [Villalonga 2009]	No	No	No	No	No	Human Activity Recognition

Figure 4.18: Comparative table among the existing work.

have proposed the most of QoC indicators described in the literature: probability of correctness, trust-worthiness, resolution, up-to-dateness, precision, freshness, spatial and temporal resolution, accuracy, completeness, representation-consistency, accessSecurity, delay time, probability of Consistence;

- *QoC parameters:* only the work described in [Manzoor 2008, Manzoor 2009a] make the distinction between QoC indicators (i.e., high-level QoC information) and QoC parameters (i.e., raw data used for measuring QoC indicators);

- *QoC measuring methods:* despite the long list of proposed QoC indicators in the literature, only three authors [Manzoor 2008, Kim 2006b, Grossmann 2009] propose methods to evaluate them;

- *QoC modeling approach:* with regarding the technology used to model QoC information, most work do not make an explicit choice. However, among those work that explicitly describe the proposed models, it is notorious the predominance of ontology-based approaches [Preuveneers 2006, Bu 2006, Tang 2007];

- *Context Management Framework:* in [Manzoor 2009b, Manzoor 2009a] [Bu 2006, Manzoor 2008, Tang 2007, Krause 2005] the authors proposed a context management framework to integrate QoC information, while in [Sheikh 2008, Abid 2009, Grossmann 2009] the authors used as basis an existing context management framework;

- *QoC purpose:* the most work propose general solutions for modelling, evaluating, and using QoC information. However, we observe the use of QoC in some specific domains, such as uncertainty handling of context [Preuveneers 2006], rescue situations [Manzoor 2009b], privacy protection of context [Sheikh 2008], and inconsistency resolution [Bu 2006].

Chapter 5

Synthesis of Related Work

Résumé: *Ce chapitre discute les travaux présentés dans les chapitres précédents, en faisant un lien avec les contributions de ce travail. Les contributions sont divisées en trois parties : la famille de contrôle d'accès sensible au contexte ; une architecture de gestion d'information contextuelle qui prend en compte les aspects de qualités associées à ces informations ; l'intégration entre ces deux proposition pour la protection des ressources dans les environnements pervasifs. Un aperçu de la proposition est présenté et nous guidera dans les chapitres suivants qui décrivent chaqu'une de ces contributions.*

Contents

5.1 Introduction

This Chapter makes a synthesis of the existing work described in Chapter 2, Chapter 3, and Chapter 4. First, we describe in section 5.2 the existing access control approaches for pervasive environments. Second, we discuss the existing QoC modelling, measuring, and management approaches in section 5.3. Finally, we present in section 5.4 an overview on the proposal and the open issues that this work is addressing.

5.2 Access Control approaches

Chapter 2 describes the main traditional access control solutions (MAC, DAC, RBAC), presenting some existing mechanisms that implement them. Among these solutions, RBAC model stands out due to the ease management of permission assignments by using the role concept (i.e., user-role assignments, and role-assignments).

However, we observe that these models are context-unaware with regarding the policy enforcement mechanism, and the user-permission assignments are statically determined in the system (section 2.2, 2.3, 2.4). In order to take into account some environmental information, RBAC model has been extended for supporting generalized roles [Covington 2000, Moyer 2001], temporal [Bertino 2001, Joshi 2005], spatial[Jiang 2002, Hansen 2003, Bertino 2005, Damiani 2007, Zhang 2006], and temporal-spatial conditions [Chandran 2005, Ray 2007, Ray 2008, Aich 2007] when making user-role assignments. Some of these extensions are described in Section 2.5. Despite incorporating concepts that allow the dynamic role activation taking into account spatial, temporal, and spatial-temporal constraints, the support to contextual information is very limited (spatial and temporal aspects). Moreover, they incorporate complexity (e.g., object and environment roles [Covington 2000, Moyer 2001]) that hind the maintainability of the system.

In chapter 3 we described access control solutions specifically proposed for pervasive environments. We divided the existing work in two groups: context-aware access control - CAAC (section 3.2) and context-based access control approaches - CBAC (section 3.3). The first group of solutions propose RBAC extensions for supporting context information. They proposed new concepts to capture relevant context information to make dynamic user-role and role-permission assignments. For instance, Covington *et al.* [Covington 2001] proposed the environment role concept, Sampemane *et al.* [Sampemane 2002] the space role concept, Kumar *et al.* [Kumar 2002] the context role concept, and Zhang *et al.* [Zhang 2004] the dynamic user-role and role-permission assignments activated by changing context.

Once these solutions are based on roles, administrators need to define user-role assignments to all possible users of the environment, associating context-dependent constraints for activating them. However, the set of roles in pervasive environments may be easily determined and fixed, while some users may be unknown by the system. Therefore, pervasive environments require new approaches to assign permission to the users that should be done in a natural way and context dependent. With this in mind, Covington *et al.* [Covington 2006] proposed contextual attributes as central concept to grant permission to the users based on policies. Other solutions has been proposed considering context as central concept for granting permission, such as UbiCOSM [Corradi 2004a, Corradi 2004b], SCAS [Hulsebosch 2005], ACA^2 [Yokoyama 2006], and a semantic-based approach [Toninelli 2006].

Despite the existence of context-based propositions, we observe the need to formally define context-based access control models that could be used as basis for

implementing context-based access control mechanisms. According to the access control requirements of a specific pervasive and its characteristics, a context-based model could be used as basis of specification to implement a mechanism that meets very well these requirements.

For example, in a pervasive environment that supports distributed context providers embedded on personal devices, it is important to verify the quality of context gathered from these sources before using that information to activate permission.

Moreover, privacy requirements of users on their context information may reduce the quality of context information used for making access control decisions. In some scenarios, users might require to get anonymous access on resources. Therefore, the access control system should be able to enforce context-based access control policies taking into account the privacy requirements of their users.

5.3 QoC Modeling and Evaluating approaches

Despite the importance of taking into account QoC when making context-based decisions [Buchholz 2003, Razzaque 2005, Preuveneers 2006, Sheikh 2007, Sheikh 2008], few works [Kim 2006b, Manzoor 2008, Grossmann 2009] have been carried out proposing QoC measuring methods. Moreover, these studies propose to evaluate the quality aspects only on raw context information, i.e. they do not consider that raw context data might be used to posteriorly generate new context information (e.g., more understandable by humans) by applying inference/derivation processes. In fact, context management systems could perform some transformation operations (e.g., inference, derivation, truncation, enrichment, etc) on context information before providing it to the context consumers.

In addition, the enforcement of user's privacy policies on context information can reduce the quality of disclosed context information. In this case, QoC values associated with the resulting context information may be unknown by the context consumers. Normally, the QoC values associated with that new context information should be equal or lower to the QoC associated with the contextual information used to determine it.

For instance, the existing QoC measuring approaches [Kim 2006b, Manzoor 2008, Grossmann 2009] are unable to answer the following questions: *what are the QoC aspects that characterize the real address of users (i.e., country, city, street, and number) derived from GPS coordinates? What is the precision of the disclosed indoor location of users?* Moreover, these studies do not describe clearly at what moment (e.g., gathering time, using time) and by whom (e.g., internal process of context management system, context consumers) QoC should be assessed.

We still observed that *QoC* is not widely used to improve context-based security

services. Moreover, QoC indicators with regarding security aspects are not explored by the existing solutions.

5.4 Overview of the proposal

In view of the open issues described in sections 5.2 and 5.3, this work proposes a family of context-based access control models (CxtBAC) and a quality and privacy-aware context management framework (CxtMF).

CxtBAC offers the basis of specification for implementing context-based access control systems for pervasive environments. Permission is assigned to users taking into account context information that characterizes any entity considered as relevant for making access control decisions, such as resource owner, resource requestor, resource itself, and the environment around them. This set of information is named *access context*. We proposed an ontology, named *AccessCxt* (see Chapter 7), that represent the relationship among context concepts describing the situation of these entities at access request time.

CxtBAC is composed by eight access control models: $CxtBAC_0$ (the base model), $CxtBAC_1$ (hierarchies), $CxtBAC_2$ (constraints), $CxtBAC_3$ (core model), $Q-CxtBAC$ (Quality-Aware CxtBAC), $P-CxtBAC$ (Privacy-Aware CxtBAC), $S-CxtBAC$ (Social-Aware CxtBAC), and $QP-CxtBAC$ (Quality and Privacy-Aware CxtBAC). According to specific requirements of pervasive environments (e.g., quality-awareness), any of the models can be used as basis to implement a context-based access control system.

The main characteristics of *CxtBAC* is described bellow:

- *CxtBAC* offers a way for assigning permission to users based on the concept of *access context*;
- *CxtBAC* is a basis of specification for implementing context-based access control solutions;
- *CxtBAC* supports discretionary and mandatory policies;
- Each *CxtBAC* model takes into account different requirements of pervasive environments, such as quality, privacy, and social-awareness;
- *CxtBAC* supports context information associated with resource owner, resource requestor, resource, and the environment for specifying access control policies.

CxtMF is in charge of gathering, managing, and providing context information for context-based application and services, such an instance of *CxtBAC* family. We defined three ontology that are used as basis to represent context information

associated with the following observed entities: user (CxtUser ontology), resource (CxtRes ontology), environment (CxtEnv ontology). These extensible ontologies can accommodate new context concepts for characterize the situation of their correspondent entity. Moreover, we defined another ontology (AccessCxt) from these three ontologies in order to aggregate the relevant context for making access decisions.

CxtMF is based on components that can be dynamically deployed to perform the following context management services: context reasoning, context obfuscation, and QoC evaluating. In order to support QoC evaluating operations, we defined a QoC ontology and QoC measuring components. QoC information is still used by *CxtMF* to improve itself (e.g., selecting context sources based on QoC thresholds). We present the main characteristics of *CxtMF*, as follows:

- *QoC* ontology classifies quality information into two groups: QoC indicators (QoCI) and QoC parameters (QoCP);
- *CxtMF* is able to evaluate QoC of inferred, modified, and derived context information;
- *CxtMF* is modular, offering points of adaptation for its internal management features (e.g., support to QoC evaluating and context reasoning process);
- *CxtMF* integrates new QoC indicators for describing quality of context from security aspects;
- *CxtMF* integrates new QoC evaluating methods.

In order to implement a real access control solution using as basis our work, developers/software engineers should follow the process illustrated in Figure 5.1:

1. *Identifying Access Control Requirements:* in this step the developer/software engineer should identify the main access control requirements that will guide the development of the access control system, such as quality, privacy, and social-awareness;
2. *Selecting a CxtBAC Model Element:* from the list of requirements identified by the developer/software engineer, one or other *CxtBAC* is selected as basis of specification;
3. *Modeling relevant Context Information:* if the context model defined in *CxtMF* (i.e., *AccessCxt* ontology) do not meet the needs of the pervasive environment, new concepts can be specified and added to set of defined ontologies;
4. *Selecting a policy representation and enforcement approach:* it is necessary a policy representation and enforcement approach for evaluating the access control policies. For example, policies can be described by using a semantic

approach (e.g., owl ontologies combined with SRWL rules) or an existing policy language such as XACML[1];

5. *Integrating a context management system:* in this step, the context-based access control solution should be integrated with a context management system, such as the $CxtMF$ proposed in this work;

6. *Deploying the access control solution:* the implemented solution should be deployed and verified before being available to the users/administrators;

7. *Management of the access control system:* this last step corresponds to the life cycle of the access control solution. Users/administrators should define and manage access control policies based on the context concepts described in the defined context ontology (i.e., the *AccessCxt* ontology or its extensions).

In the next chapters, we present in detail our work ($CxtBAC$ models, $CxtMF$, and the integration between them for protecting multimedia resources).

[1] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

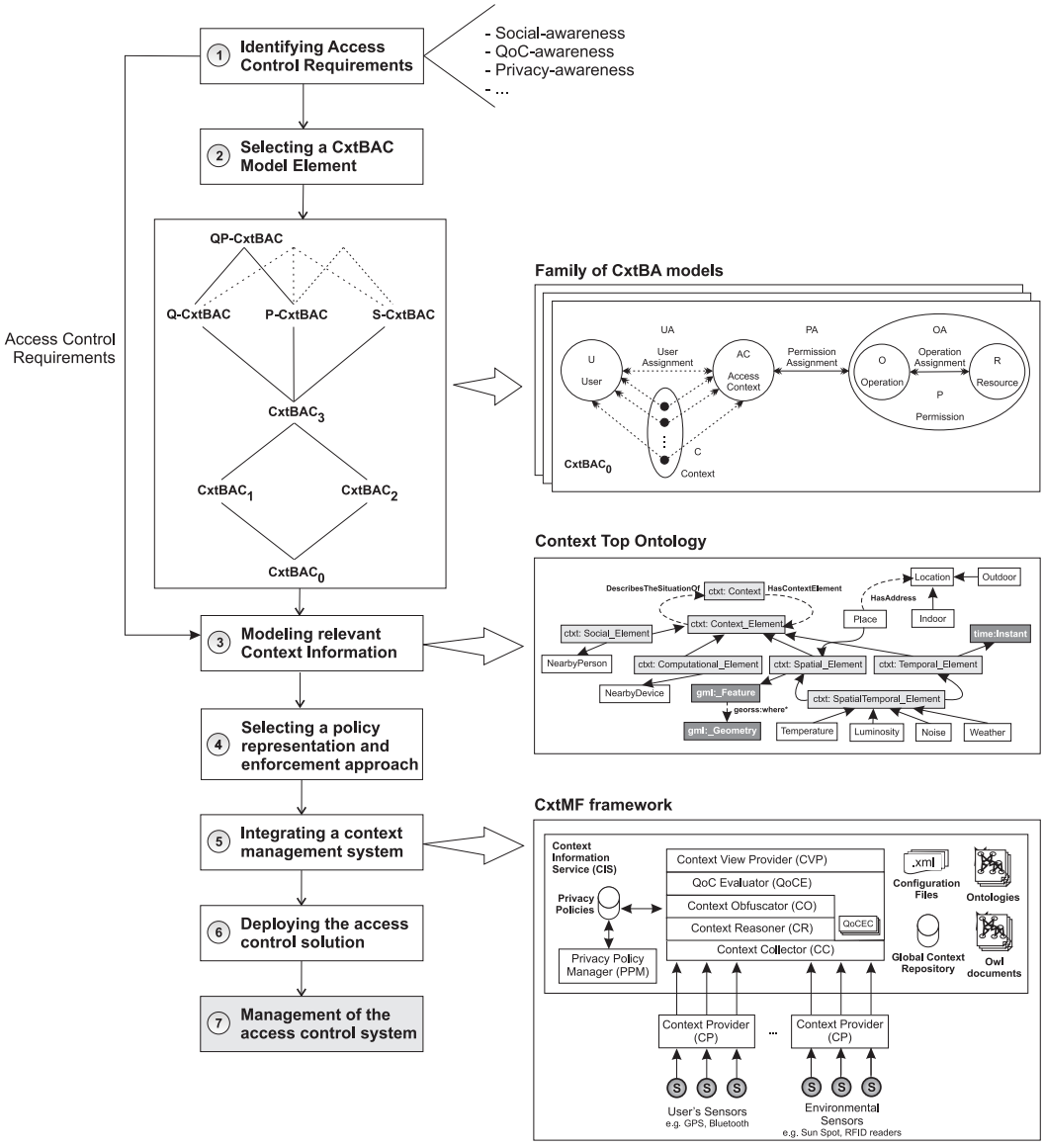


Figure 5.1: Using CxtBAC and CxtMF to implement access control solution.

Part II

Proposition

Chapter 6

CxtBAC - a Family of Context-Based Access Control Models

Résumé: *Ce chapitre présente la famille des modèles de contrôle d'accès proposé pour la protection des ressources dans les environnements pervasifs. Cette famille est composée par 8 (huit) modèles différentes qui pourrons être utilisé comme point de départ de développement des mécanismes de sécurité pour la protection de ressources selon les besoins de chaque scenario considéré. Ainsi, un langage générique est proposé pour la définition de politiques de sécurité basée sur des règles sensibles au contexte. Nous présentons également différentes propositions pour la vérification de politiques de sécurités sensible au contexte et discutons les besoins d'implémentation de ces modèles.*

Contents

6.1 Introduction

In our life, normally we grant access to other people based on situations. It means we grant access on our resources according to our situation, the situation of other people that may get access to our resources, the situation of resources itself and the situation of the environment that around us. For example, we could lend (i.e., granting access) our summer house (resource) to a select group of friends (group of requestor) when the house is available (i.e., we are not staying in the house). Like in real life, in spontaneous pervasive environments the short-lived interactions between people and resources occur often in a dynamic, distributed, and transparent manner. In these scenarios, it is desirable to grant/get access permission on protected resources taking into account the current situation of the entities that interact with the pervasive environment.

Moreover, it should be possible users to get access on distributed resources that do not have a well-defined ownership relation. For example, a user may wish to print his/her documents using a printer currently available in the pervasive environment. In this case, we do not know clearly who is the printer's owner (normally this kind of resource belongs to the owner of the environment, such as a company or organization). From our point of view, smart usage of context information provides a powerful approach for controlling access to resources that, in many situations, is more suitable than conventional identity-based or role-based access control solutions.

In traditional access control models described in Chapter 2, we observed that users and objects must be known *a priori* to define access policies. These policies introduces unnecessary administrative complexities by forcing rigid rules. In reality, users and protected objects possess certain properties, such as location, that change rapidly, thus making traditional policies inflexible and ineffective in dynamic pervasive environments.

we state that, the same as with role, the concept of context can provide an indirection level between users and permission. Instead of managing subjects and their permissions individually, a system administrator or the resource owner defines for each context the set of applicable permissions. When a subject operates in a specific context granting special permission, he/she instantaneously acquires that set of activated permission. When he/she changes his/her operating context, his/her previous permission is automatically revoked and new permission may will be acquired according his/her new context.

With this in mind, this Chapter presents our work that proposes a family of reference models for Context-Based Access Control, named (CxtBAC). *CxtBAC* is a less intrusive and more flexible access control model that mimics our natural way of access authorization in the physical world. *CxtBAC* exploits the ability to sense and use contextual information to augment or replace traditional user attributes such as username/password for the purpose of authentication and access control by making them less intrusive and adaptable to situational or contextual changes.

In this perspective, instead of assigning permissions directly to the users/roles, resource owners/administrators may define for each protected resource the context conditions that enable someone to access it, i.e., the access control policies are completely based on and dependent of context information. When a request on a protected resource is made, the access control mechanism should identify the current context in order to enforce the associated set of access control policies. *CxtBAC* considers user identities and roles as specific types of context information. This allows *CxtBAC* to handle also subject-based access control if needed. Therefore, unlike traditional access control models permission is directly associated with context instead of user identities/roles. A mobile user/device acquires a set of permission by entering a specific context.

Our main idea is to propose a family of reference models. Therefore, no particular implementation mechanism is imposed when describing formally the family of model. We have defined this family taking into account the requirements of pervasive environments for making control access decisions. In order to implement a element of the proposed *CxtBAC* family, firstly we need to identify the set of specific security requirements of the pervasive environment. For example, a pervasive environment that uses context information gathered from distributed context providers belonging to many different domains, might verify the quality of that context information used for making access control decisions.

For each element of *CxtBAC* family, permissions are associated with situations, called by us of *access context*, and users are part of these *access contexts*. CxtBAC reference models provide a systematic approach for understanding *CxtBAC*, categorizing its implementation in different context-aware access control services. *CxtBAC* includes capabilities to establish relationships between *access contexts* and *permission*, as well as between *users* and *access contexts*.

For example, two *access contexts* can be established as mutually exclusive, them the same user is not allowed to take on both *access context*. *Access contexts* can also take on inheritance relations, whereby one *access context* inherits permissions assigned to another *access context*. With *CxtBAC* it is possible to predefine (*access context*) × (*permission*) associations, which the system should be able to identify the set of activated *access context* according to the current context and makes it simple to dynamically assign users to the predefined *access contexts*.

Access contexts are created for describing various situations in pervasive environment and users are dynamically assigned to these *access contexts* based on current situation. Users can be easily reassigned from one *access context* to another. *Access contexts* can be granted new *permissions*, and *permissions* can be revoked from *access contexts* as needed.

An *access context* can be properly viewed as a semantic construct around which access control policies are described. *Users* and *permissions* brought together by an *access context* that is more stable in pervasive environments because we are able to describe the situations in which we grant access on our resources but we cannot knows previously the users that will interact with us possibly consuming and sharing resources.

Access control policy is embodied in various components of CxtBAC such as (*access context*) × (*permission*), (*user*) × (*access context*), and (*access context*) × (*access context*) relationships. These components collectively determine whether a particular user will be allowed to access a particular piece of data in the environment. *CxtBAC* components may be configured directly by the system owner, by the resource owner, or indirectly by an administrator as delegated by the system owner. Moreover, the access control policy can evolve incrementally over the system life cycle. The ability to modify policy to meet the changing needs of resource owners is an important benefit of *CxtBAC*. For example, when an access policy is no longer applicable in future situations (i.e., the validity of policy is outdated) it should be removed from the activated list of access policies.

The family of *CxtBAC* models proposed is neutral to security policies. It means that the definitions presented in this Chapter are independent of implementation and security policy language used to describe the rules and access permissions on the protected resources. Moreover, *CxtBAC* supports two well-known security principles described in following: *least privilege* and *data abstraction*. Least privilege is supported because access control system implementing *CxtBAC* should be configured for granting only the set of permissions required for the current situation of users. This is accomplished dynamically through (*user*) × (*access context*) associations. Data abstraction is supported by means of abstract permission, such as share an multimedia object (e.g., a photo, a video) rather than read, write, execute, and delete permission typically provided by operating systems.

CxtBAC does not take into account the control of operation sequences, such as

context-aware workflows [Georgakopoulos 1995]. In fact, this control is outside the scope of *CxtBAC* models, although it can be a foundation on which to build such controls. In the next section we briefly describe a comparison between the proposed family of access control models with the RBAC model.

6.2 CxtBAC definitions

In order to make clear the basis concept of *CxtBAC*, we need to answer a question before presenting the family of *CxtBAC* reference models: *what are the differences between roles, groups, and access contexts?*

A major difference between roles and groups is that groups are typically treated as a collection of users and not as a collection of permission [Sandhu 1996]. A role represents both a collection of users on one side and a collection of permissions on the other. Roles serve as intermediary to bring these two collections together. Moreover, roles have two characteristics: it should be equally easy to determine role memberships and role permission associations [Sandhu 1996]; the control of role memberships and role permissions should be relatively centralized in a few users (e.g., administrator, owner of system).

The main difference between role and *access context* is that this last is created for describing a situation in which users could be dynamically assigned in order to have access permission on resources. From our point of view, roles could be interpreted by *CxtBAC* models as an information describing the current situation of users, i.e., the current role assigned to the users is part of their situations (i.e., *access context*). Therefore, the idea behind the concept *access context* is broader than *role*. A user may perform a function (a role) when requesting access permission on a protected resources, but this request is accomplished in a given situation.

Making a more detailed comparison between RBAC and the proposed *CxtBAC* models, we can observe the differences bellow:

- **(Role) x (access context):** roles describe functions performed by users, which generally follow the hierarchy of organizations. In the *CxtBAC*, we have replaced this concept by *access context*. *Access context* concept do not represents the function of users but rather the situation in which them are inserted at request time of a particular protected resource. Moreover, *access context* do not refers only to the situation of *resource requestors*, but also the situation of *resource owners*, resources themself, and pervasive environments;
- **(Session) x (context):** in RBAC-based models, session is an important mechanism for determining the roles assigned to the users [Ferraiolo 1992, Sandhu 1996]. However, sessions are not part of CxtBAC models. From our point of view, a session corresponds to a particular occasion when a user signs on the system to carry out some activity, which can vary widely from system to

system, i.e., sessions represent the period of time in which users are recognized by the access control systems after passing through an authentication process. Thus, we do not formalize how sessions should be established in the systems implementing *CxtBAC* models. Therefore, unlike RBAC-based approaches sessions in *CxtBAC*-based solutions do not grant immediately permissions on any resources to the users through determining role memberships. In fact, users only get access to the resources when there exist one or more activated *access context*, according the current context, granting permissions. During a session the *CxtBAC*-based solution should be able to identify the current context of users in order to determine the set of activated access control policies. Once one or more *access context* are activated, users may get permission associated with an activated *access context*. In this case, when a matching occurs the concerned users will have the permissions described in the activated access control policies. At this moment, we need to add these policies in the list of activated access control policies in order to verify their validity to each change on the context. Therefore, we need an entity in charge of verifying the list of activated access control policies in order to revoke the granted permissions when the context of access is no longer valid;

- **Dynamic and static access context**: unlike roles in RBAC models, *CxtBAC* supports two types of *access context*: *static* and *dynamic*. The *static access context* have a well-defined lifetime. However, a *dynamic access context* has an indefinite lifetime. For instance, the access context named $Reunion_X$ is static because it will be active only during the reunion and could be disabled just after finishing it, revoking all the granted permission. Disabling *static access contexts* will improve the performance of policy enforcing process. By contrast, a *dynamic access context* may be activated in various situations. For example, the access context named $Working$ will be activated in all business days during the work time for employees located in the corporative building. Therefore, the definition and activation processes of access contexts in *CxtBAC* models are more flexible than roles in RBAC models;

- **Uncertainty about the set of assigned permissions**: in RBAC models, the set of permissions assigned to each role and the $(user) \times (role)$ associations are known previously. However, in CxtBAC the users are not able to know at long-term what permission are assigned to them. In fact, situations change constantly and, consequently, the permissions assigned to the users. Therefore, in dynamic collaboration scenarios that we considered to deploy a *CxtBAC*-based solution it is impossible to define in advance all necessary policies for all possible situations.

We have identified in [Filho 2009] three entities that should be observed for gathering relevant context information for defining context-based access control policies: context of *resource owners*, context of *resource requestors*, and context of *resources* themself. In addition, we have also defined the concept of *Access Entity* that *refers*

to any set of implicated entity in an access control systems, i.e., resource owners, resource requestors, and resources themself.

However, there is yet another entity that should be observed in *CxtBAC*-based solutions: the *environment*. We named any information characterizing that entity as context of *environment*, and it can also be used for defining context-based access control policies. Context of environment describes only the situation with regard the pervasive environments, i.e., it is not assigned to any *access entity*. For instance, the period of time, the temperature of a room, etc, is context information of environment that can be used for defining context-based access control policies, independently of the entity that is requiring access on the protected resources. Bellow we present the concepts of each type of context that we have defined based on Dey's definition [Dey 2001]:

- **Context of Resource Owners (CxtOwn):** it refers to any information that can be used to characterize the situation of resource owners in the access control framework, which is considered relevant for making context-based access control decisions. For instance, the location, activity, body temperature, blood pressure, etc, could be used by a context-based access control framework that protects health care applications for making access control decisions. For instance, a patient would like to grant read permission on her medical records to any doctor located in the hospital if she is in a life-threatening situation characterized by a sudden drop in her blood pressure or in her heart rate. In this example, the access decisions on the patient medical records will be made taking into account the context of resource owner (i.e., blood pressure and heart rate of patient);

- **Context of Resource Requestors (CxtReq):** it refers to any information that can be used to characterize the situation of entities that are trying to access resources protected by the context-based access control framework. For each access request on the protected resources treated by a context-based access control framework, the context of resource requestor should be identified in order to determine the set of affected access control policies. For example, a user would like to grant read access on her presentation file for everyone located in the meeting room during a reunion (e.g., January 29^{th}, 2010, from 10:00AM to 12:00AM). In this case, the context of resource requestor (i.e., her location) is essential for making access control decisions;

- **Context of Resources (CxtRes):** it refers to any information that characterizes the situation of protected objects, which is considered relevant for making access control decisions. The set of context dimensions more relevant for describing the situation of resources is directly dependent of the type of protected object. We classify the resources in two sets: *static* and *dynamic* resources. Static resources are not subject to constant changes in their internal features over time. For this type of resource, the context information

describes its situation at creation time. In some cases, it could be interesting also to record the historic of context information for describing the situation of resources at modification time. However, in most cases the context information gathered at creation time of resources will be sufficient for the purpose of access control. For instance, photo, video, and audio contents are generally characterized as static resources and context information describing their situations at creation time are sufficient for defining context-based access control policies. A user would like, for example, to grant read access on their photos to his/her friends that were located nearby him at photo shoot time. To make this possible, each photo should be annotated with information describing the situation (e.g., Bluetooth address of nearby mobile devices, location of user's device) at photo shoot time (i.e., on creation time). Unlike static resources, dynamic resources are subject to constant changes on their status over time. For example, distributed services or even physical resources (e.g., printers, video cameras) change constantly their status over time (e.g., availability, processing load, cost, battery charge). For this type of resource we are more interested in context dimensions that characterize its current situation;

- **Context of Environment (CxtEnv):** it refers to any information that characterizes the pervasive environments and is considered relevant for making access control decisions. This type of information do not need to be associated with any access entity, however it can be used alone or in combination with other types of context information for making access control decisions. For example, we could have the following access policy defined by the administrator of a pervasive domain: the printers are available to any employee at work time. In this case, the time is a context information that characterize the pervasive environment and the access control framework will grant access to the printers to any employee that send documents for printing at work time. In fact, this type of context information refers to the physical environment where the access control system is deployed in order to control the access on resources.

The last but not least important term is the *Access Context* concept, which is defined as following: *"Access Context refers to any information that characterizes the situation of any access entities and the environment around them, which is considered relevant for making access control decisions. Access Contexts are used for enforcing access control policies in order to grant permissions to users."*

Access_context is the central concept of $CxtBAC$ models. It capture any relevant information about the access entities (i.e., resource requestor, resource owner, and resource) and the environment.

Figure 6.1 illustrates each context concept and the existing relationships among them. Let C represents the concept of context defined by Dey *et al.* [Dey 2001]. C is a set of context information that contains all other subsets that we defined

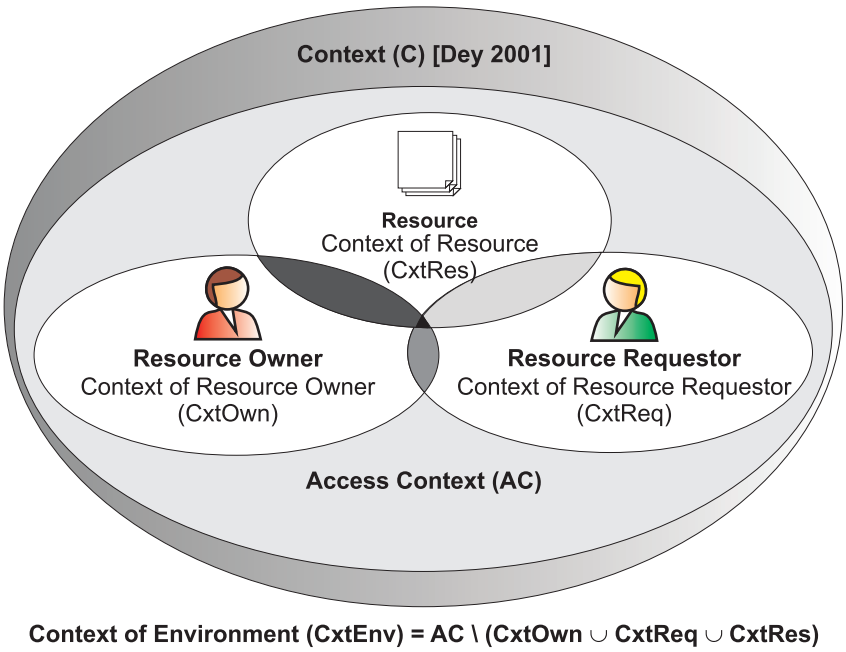


Figure 6.1: Relationships among context concepts.

previously, i.e., $C \supseteq AC$, $C \supseteq CxtOwn$, $C \supseteq CxtReq$, $C \supseteq CxtRes$, *and* $C \supseteq CxtEnv$.

The *access context (AC)* is a subset of context that is obtained from the union of four sets: context of resource owner (CxtOwn), context of resource requestor (CxtReq), context of resource (CxtRes), and context of environment (CxtEnv), i.e., $AC = CxtOwn \cup CxtReq \cup CxtRes \cup CxtEnv$. The context of environment is not illustrated in Figure 6.1. In fact, the $CxtEnv$ is the set difference of AC and the set resulting of the union of three sets: CxtOwn, CxtReq, and CxtRes, i.e., $CxtEnv = AC \setminus (CxtOwn \cup CxtReq \cup CxtRes)$.

The context of resource owner, resource requestor, and resource are different sets of information. However, it is possible that a context information belongs to one or more access entities, simultaneously (see in Figure 6.1 the intersection among these sets). For example, the resource owner, the resource requestor, and the resource itself could be located in the same place at request time. The next sections present the proposed family of CxtBAC models.

6.3 A Family of CxtBAC Reference Models

In order to describe clearly the various dimensions of $CxtBAC$ we have defined a family of conceptual models. We are using the same terminology used by the authors of RBAC96 [Sandhu 1996] for describing the proposed $CxtBAC$ family.

$CxtBAC_0$, the base model, defines the minimum requirement for any access control system implementing $CxtBAC$. $CxtBAC_1$ and $CxtBAC_2$ include $CxtBAC_0$, but add independent features to it and they are different from each other. On the one hand, $CxtBAC_1$ adds the concept of *access context* hierarchies (i.e., an *access context* can inherit permission from other(s) *access context(s) previously defined in the access control system*). On the other hand, $CxtBAC_2$ adds constraints that impose restrictions on different components of $CxtBAC$.

The $CxtBAC_3$ model is obtained from the fusion of $CxtBAC_1$ and $CxtBAC_2$, by transitivity, including the $CxtBAC_0$. From our point of view, $CxtBAC_3$ will be the model most commonly implemented following the specification of $CxtBAC$.

However, as discussed in Chapter 5 and Chapter 4, it is very important for ensuring the correctness of access decisions to take into account also the quality of context information (QoC) used for enforcing context-based access control policies. With this in mind, we have defined a new conceptual model from the $CxtBAC_3$, named *Q-CxtBAC* (Quality-Aware CxtBAC) for taking into account QoC constraints on *access contexts* when enforcing access control policies. We have identified also that in some situations it is necessary to enforce some privacy constraints when granting access permissions on the protected resources. In order to take into account the social relationship among users we proposed the $S-CxtBAC$ using as basis the $CxtBAC_3$ (Social-Aware CxtBAC). Therefore, we have also proposed another $CxtBAC$ model, named *P-CxtBAC* (Privacy-Aware CxtBAC) from the $CxtBAC_3$ in order to take into account privacy constraints when enforcing access control policies. Finally, we have defined the *QP-CxtBAC* (Quality and Privacy-Aware CxtBAC) from the fusion of *Q-CxtBAC* and *P-CxtBAC* models, completing our proposed family of $CxtBAC$ models. It is yet possible to define other models from the fusion between Q-CxtBAC, S-CxtBAC, and P-CxtBAC. However, these models will not be described here since any additional formalism is carried out in relation to the models used as the basis for the definition.

The family of $CxtBAC$ models are intended to be reference points of comparison with existing access control systems and models in the literature, such as the solutions described in Chapter 3. They can also serve as a guideline for development of new context-based access control systems. In the next sections we describe in details each element of the proposed family of $CxtBAC$ models.

6.4 CxtBAC$_0$ - Base model

$CxtBAC_0$ is illustrated in Figure 6.3. This model is composed by five sets of entities called $user(U)$, $Access\ Context(AC)$, $Resource(R)$, $Operation(O)$, and $Permission(P)$, respectively. There is also a collection of $Context(C)$ that is not an entity of our model, but it is important to show how users and access context entities relate to each other.

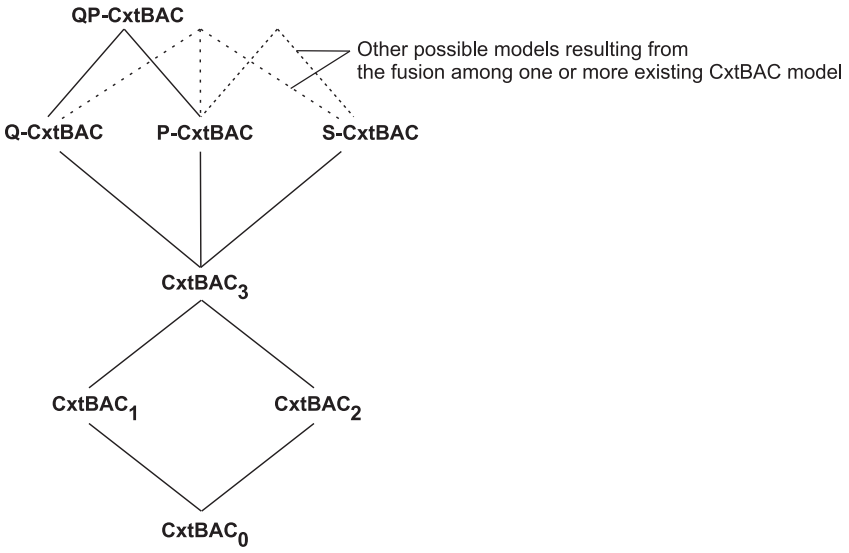


Figure 6.2: The family of CxtBAC models.

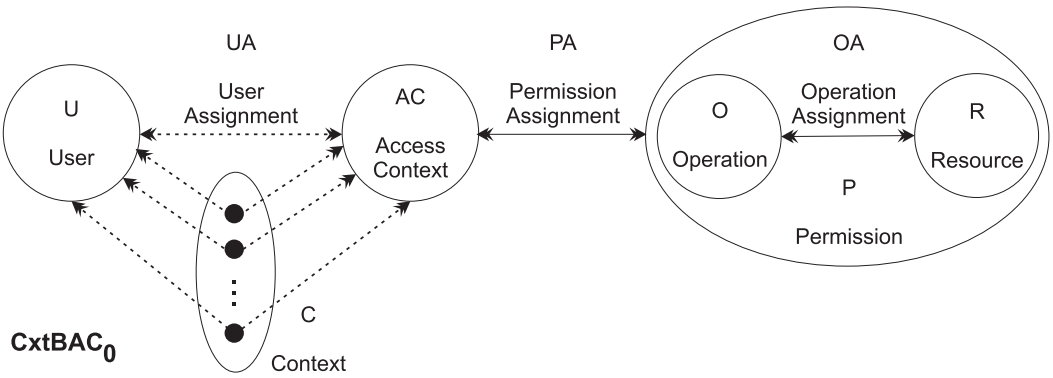


Figure 6.3: $CxtBAC_0$ - CxtBAC base model.

We consider a *user(U)* in this model as a human being, however this concept can be generalized to include other types of entities that are able to require access on protected resources, such as intelligent agents and web services.

An *access context* is defined in our model as a set of context constraints regarding to the situation of *access entities* and the pervasive environment around them at request time. It means that a user (i.e., the resource requestor) will get access permission on protected resources if and only if (iff) at request time the current situation meet the context constraints associated with one or more predefined *access contexts*. If the context constraints associated with an access context are satisfied, then the permission associated with that access context will be granted to users.

In the *CxtBAC* reference model, a *permission(P)* is an approval of a particular mode of access to one or more objects protected by the access control systems, i.e., it is an approval to perform an *operation(O)* on one or more *resource(R)* in a given

Access Context(AC). *CxtBAC* supports various interpretations for permission, from very coarse grain (e.g., access on a folder of files) to very one grain (e.g., a particular instance of classes defined in ontologies[1]).

Moreover, in the family of *CxtBAC* models permission is always positive, thus the system confer to user(s) only operation(s) for executing on the protected resource(s) granted by the resource owner(s) or the system administrator(s), in well defined situation(s). However, there are access control approaches, such as Access Control Lists (ACL) [Ferraiolo 2003], that support *negative permission* for denying access rather than confer permission. *CxtBAC* supports denial of access as constraints rather than a negative permission (see $CxtBAC_2$ for more details).

The nature of permission will depend directly on the implementation details of the access control system that follows our reference model, and the type of protected resources. For instance, an operating system protects resources such as files, directories, devices, ports, etc., with operations such as READ, WRITE, and EXECUTE. Therefore, with the intention to define the proposed family of models as generic as possible, we treat permission as abstract symbols that are independents of implementation. Moreover, the manner in which individual permission are joined into a generic permission in order to be assigned as a single unit is highly dependent of implementation.

Figure 6.3 shows *User Assignment (UA)*, *Permission Assignment (PA)*, and *Operation Assignment (OA)* relations, both are many-to-many relations. The double-head dashed arrows represent dynamic associations among users and access contexts. In order to be established, these associations depend on the activation of access context.

Therefore, a user (i.e., the resource requestor) can be dynamically associated with many access contexts, and an access context can have many users. An access context can have many permission, and the same permission can be assigned to many access contexts. Similarly, a resource can have many operations, and the same operation can be assigned to many resources.

The key to *CxtBAC* lies in UA and PA relations. Treating access contexts as intermediary for enabling users to exert permissions provide much greater control over access configuration than directly relating them to permission.

Access control systems that implement the proposed reference model should be able to identify, at request time, the subset of activated access context taking into account the current situation (context). If one or more access contexts are activated in that situation, the system will grant the assigned permissions to the users, i.e., the situation at request time will be verified in order to identify some subset of activated access context that users are member of.

The double-headed dashed arrow from the context to AC in Figure 6.3 indicates

[1] http://www.w3.org/TR/owl-ref/

that multiple access contexts could be simultaneously activated in a determined situation. The set of permission granted to the user is the union of permission from all access context activated in that context.

Each context is associated with various users, as indicated by the double-headed dashed arrow from the *context* to U in Figure 6.3. This association remains constant for the time life of a context, i.e., while there is no change in the context. A user who is a member of several activated access contexts can exercise any subset of permission that is suitable for her situation. This feature of $CxtBAC_0$ supports the principle of *least privilege*.

The following definition formalizes the above discussion:

Definition 9. *The $CxtBAC_0$ Model is composed of the following components:*

- *A set U of users, a set AC of access contexts, a set R of resources, a set O of operations, a set P of permission, and a generic context condition language $GCCL$;*
- ***User Assignment (UA):*** *$UA \subseteq U \times AC$, a many-to-many dynamically relationship mapping user to access context assignment relation. UA $(UA = 2^{U \times AC}) = \{(u, ac) | u \in U, ac \in AC\}$;*
- ***Operation Assignment (OA):*** *$OA \subseteq O \times R$, a many-to-many operation to resource assignment relation;*
- ***Permission Assignment (PA):*** *$PA \subseteq P \times AC$, a many-to-many permission to access context assignment relation. This set of permission assignments is defined as following: PA $(PA = 2^{P \times AC}) = \{(p, ac) |\, p \in P,\, ac \in AC\}$;*
- *$AC = \{(ac,\ e) \mid ac$ is a label and e is a context constraint expression defined using the Generic Context Condition Language $(GCCL)\}$;*
- *$P(P = 2^{R \times O}) = \{(r, o) | r \in R, o \in O\}$, which each permission is a approval to perform an operation on one resource in a given access context;*
- ***Assigned users to an access context:***
  assigned_ U(u, ac) *$= \{u \in U, ac \in AC | (u, ac) \in UA\}$, the mapping of an access context onto a set of users;*
- ***Assigned permission to an access context: assigned_ P(p, ac)*** *$= \{p \in P, ac \in AC | (p, ac) \in PA\}$, the mapping of an access context onto a set of permission;*
- ***Operations associated with a permission:*** *$(p : P) \rightarrow o \subseteq O$, the permission to operation mapping. It is the set of operations associated with a permission p;*

- ***Resources associated with a permissions:*** $(p : P) \rightarrow r \subseteq R$, *the permission to resource mapping. It is the set of resources associated with a permission* p;
- $users : C \rightarrow U$, *a function mapping each context* c_i *to a set of users* $U(c_i)$;
- $access\ contexts : C \rightarrow 2^{AC}$, *a function mapping each context* c_i *to a set of access contexts* $AC(c_i) \subseteq \{ac\,|(U(c_i),\ ac) \in UA\}$ *and context* c_i *has the permissions* $\bigcup_{ac \in AC(c_i)}\{p\,|\,(p,\ ac) \in PA\}$.

We expect each access context to be assigned to at least one permission, however our reference model does not requires this explicitly. Moreover, the resource owners or administrators should define access control policies predicting the possible scenarios in which they grant permissions on protected resources, since the association between access contexts and users are performed dynamically.

As discussed previously, $CxtBAC_0$ treats permissions as abstract symbols because the nature of a permission is dependent of access control implementation and access control system requirements. We expect permission is applied on resource objects and not to the components of $CxtBAC$ models.

However, there is a set of special permission to modify the sets U, AC, P, O, R, and relations UA, PA, and OA, called administrative permissions. These permissions will be discussed later (Section 6.12) in the management operations of $CxtBAC$. We assume that only a single user (i.e., resource owner) or administrator can change these components.

Ideally, an access control system that implements this reference model has to be built on a context management framework. These frameworks could offer to the users and administrators the possibility of describing privacy policies on context information in order to protect the privacy of users (see Chapter 7 for more details). In this case, the enforcement of privacy policies will impact directly on the set of permission that users will be enable to exercise based on her disclosed context. In fact, the access control system will be unable to identify some context information of users which, therefore, will limit the set of activated context-based access control policies.

We provide additional detail of the Generic Context Condition Language (GCCL) defined in order to describe condition context expressions associated with *access contexts*.

6.4.1 Generic Context Condition Language (GCCL)

$CxtBAC_O$ model includes a simple generic language for expressing context constraints (GCCL) associated with *access contexts*. We need to define this language to offer means of defining context constraints independently of policy implementation, such as XACML. A context constraint is defined as a dynamic constraint

that checks the actual values of one or more contextual attributes for predefined conditions. If these conditions are satisfied, the corresponding access request can be permitted. This language is independent of context model and access control system implementations.

The context constraints are defined by means of expressions that should be evaluated when enforcing context-based access control policies. A context constraint expression is defined through the terms *observed entity*, *context object*, *context function*, and *atomic context condition*.

- *Observed entity:* it represents any entity that could be observed by the system, which is considered relevant for making access control decisions. We have identified four types of observed entities: *resource requestor*, *resource owner*, *resource* itself, and the *environment* around them;
- *Context object:* it represents a type of context information that characterizes a observed entity, such as *location*, *nearbyDevice*, and *activity*. Thus, context object is a means of making context information explicit to the policy enforcing process of an context-based access control system. For example, the context object *location* associated with the requestor entity is referenced by *requestor.location*;
- *Context function:* it is a mechanism to obtain the current value of a specific context object property that characterize a observed entity. For instance, the function *getGPSCoordinates()* returns the current GPS coordinates of a context object (e.g., *location*) associated with a observed entity (e.g., *Requestor.location.outdoor*). The context functions are encapsulated into the context objects (e.g., *Requestor.location.outdoor.getGPSCoordinates()*) managed by entities in charge of context management operations, such as the CxtMF described in Chapter 7;
- *Atomic context condition:* it is a predicate that consists of an operator and two or more operands. At least one operand represents a property of a certain context object (e.g., *Requestor.location.outdoor.GPSCoordinates*), while the other operands may be either a context object property (associated with the same or another context object) or a constant value. The values of context object properties are gotten by using corresponding context functions. The operator can be a prefix operator that accepts two or more input parameters or a binary infix operator that compares two values;
- *Context constraint expression:* it is a clause containing one or more atomic context conditions.

In the following, the definition formalizes how context constraint expressions could be described by using the GCCL:

Definition 10. *Let $CxtObj_X$, $CxtObj_Y$, ..., $CxtObj_N$ be context objects that characterize the observed entities (e_W, e_Z, ..., e_M) associated with a context c ($c \in C$). Each property p of a $CxtObj \in c$ has a domain of possible values, denoted as $Dcxt$. An atomic context condition (acc) defined over c has the form ($e_W.CxtObj_X.p$ op $vCxt$), where $e_W.CxtObj_X, e_Z.CxtObj_Y, ... \in c$, $vCxt \in Dcxt$, $op \in OP = \{>, \geq, <, \leq, \neq, =\}$. The set of op can be extended in order to accommodate user-defined and administrator-defined operators as well. For example, we can add spatial operators such as inside, disjoint, or the set operator "in" to verify the pertinence of elements. The context constraint expressions of GCCL are defined as following:*

- *An atomic context condition (acc) is, itself, a context constraint expression of GCCL;*
- *Let acc_i and acc_j be context constraint expression of GCLL, then $acc_i \wedge acc_j$ is also a context constraint expression of GCCL;*
- *Let acc_i and acc_j be atomic context conditions of GCLL, then $acc_i \vee acc_j$ is also a context constraint expression of GCCL.*

Based on this generic language, we are able to specify complex context constraints associated with *access contexts* in order to describe any type of context-based access control policies supported by the proposed model. *CxtBAC* supports exactly $2^4 - 1$ types of access control policies resulting from the combination of four context information sets: context of resource owner, context of resource requestor, context of resource, and context of environment (see Section 6.13 for more detail).

As the proposed model is independent of access policy implementation, we have also defined a high-level format to offer a means of describing generic context-based access policies. We present this generic representation format in the next subsection.

6.4.2 Generic Context-Based Access Control Policies

In *CxtBAC*, the access policies are used to mediate context-based access control decisions. As we have discussed previously, *CxtBAC* is a policy neutral model, meaning that the language to be used to represent access control policies is not included in the model. However, it is necessary to propose a generic representation of access policies in order to guide the developers when implementing any element of the family of *CxtBAC* models.

Therefore, we propose an abstract and generic format based on tuples to describe policies for mediating context-based access control decisions. In fact, the tuples define relationships between the entities of $CxtBAC_0$ illustrated in Figure 6.3, i.e., each tuple defines associations between *User*, *Access Context*, and *Permission*. The generic format is defined as below:

$$generic\_policy(p_i) = [u, p_{set}, (ac, e), bit] \tag{6.1}$$

- u(user) is a identity assigned to resource requestor (e.g., login, identifier, name, group, role) in the current access context. The identity of resource requestors can change because we suppose they are able to reduce dynamically the disclosure level of their identity in order to protect their privacy. When the identity of requestor is omitted or it is assigned the value *everyone*, only the resource requestors assigned to the access context ac that meets the context constraint described in the expression e will get access permissions on the protected resource;
- p_{set} is a set of one or more permission, where $p_{set} \subseteq P$. Let p be a permission in the set p_{set}. p is a tuple that defines the relationship between a resource and an operation ($p_i = (r, o) \in P = 2^{R \times O}$);
- ac is an *access context* ($ac \in AC$) that restricts the set of permission p_{set} to the users. Only the users that are part of the context activating that ac will get the set of permission p_{set};
- e is a context constraint expression defined using the $GCCL$. This expression will be enforced by attributing the current values of access context objects;
- *Bit* indicates if the associated policy is enabled or disabled. *Bit* has the value 1 if the policy is enabled and 0 if the policy is disabled. By using this bit it will be possible to maintain a policy registered on the access control policy repository, however this policy will not be considered by the enforcing process of access control policies.

For each enabled access control policy in the policy repository, we need to verify the ac by replacing the current values of context objects on the context constraint expression e. If the expression e is true then the associated set of permission will be granted to affected users. In another case, the associated set of permission will be denied.

Resource owners and administrators can define a set of policies that is represented formally as follows:

$$policy\_set(pol_{set}) = \{p_i \mid p_i \, is \, a \, policy, i \geq 0, \, and \, i \in \mathbb{N}\} \tag{6.2}$$

The resources and who will be able to access them can be indeterminate at the moment of defining access control policies. For example, a user may grant read access on his/her photos taken in Paris to his/her friends. However, at definition time of this policy the user does not known who are your friends (this group is growing) and the resources that will be accessible (he/she is still taking the photos).

To implement an element of $CxtBAC$ model family we need to translate this generic representation format to a concrete access control language, such as XACML[2]

[2]Extensible Access Control Markup Language: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

language and SWRL[3] rules. Chapter 9 presents a semantic approach that implements this generic context-based access control policies, using as basis OWL ontologies for protecting multimedia resources.

6.5 CxtBAC$_1$ - Access Context Hierarchies

$CxtBAC_1$ is defined using as basis the $CxtBAC_0$ by introducing the support to access control hierarchies (ACH), as illustrated in Figure 6.4. The inheritance relationship among access context is essential to describe *access context* more specific than its senior *access context*.

Unlike RBAC-Based models, a junior access context of CxtBAC models inherits the set of permission associated with its senior *access context*. In fact, context constraints associated with a junior *access context* will be more restrictive than those associated with its senior *access context*. Therefore, if a junior *access context* is activated in a given situation then the set of permission associated with its senior also will be granted to the users.

The set of permission associated with a junior *access context* (ac_j) is equal to the set of permission resulting from the union of the set of permission directly associated with it and the set(s) of permission associated with each one of its senior(s) (ac_{s_1}, $ac_{s_2}, \ldots,$ ac_{s_n}), which is defined as follows:

$$assigned\_P(p, ac_j) = assigned\_P(p, ac_j) \cup assigned\_P(p, ac_{s_1}) \cup assigned\_P(p, ac_{s_2}) \cup \ldots assigned\_P(p, ac_{s_n}),\ n\ is\ the\ number\ of\ seniors,\ n \in \mathbb{N}$$

Moreover, the context constraint expression (e_j) associated with a junior *access context* is equal to context constraint expression resulting from the (and) among its expression and the expression (e_{s_i}) associated with each senior *access context*:

$$e_j = e_j \wedge e_{s_1} \wedge e_{s_2} \wedge \ldots e_{s_n},\ n\ is\ the\ number\ of\ seniors,\ n \in \mathbb{N}$$

An example of access context hierarchy (ACH) is shown in Figure 6.5. Like role hierarchies in RBAC-based models, by convention the more powerful access context (i.e., the juniors) are shown toward the top of hierarchy, and the less powerful access context (i.e., the seniors) toward the bottom.

In Figure 6.5, the junior-most access context is ac_4 (ReunionX) and the senior-most is ac_1 (Working). Each access context is associated with a context constraint

[3] A Semantic Web Rule Language Combining OWL and RuleML: http://www.w3.org/Submission/SWRL/

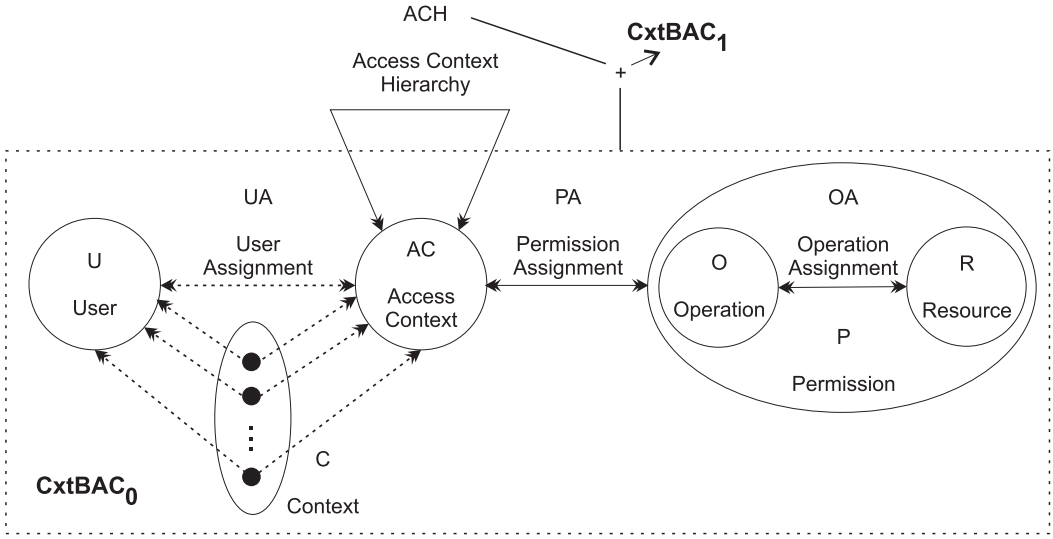


Figure 6.4: $CxtBAC_1$ - CxtBAC supporting access control hierarchies.

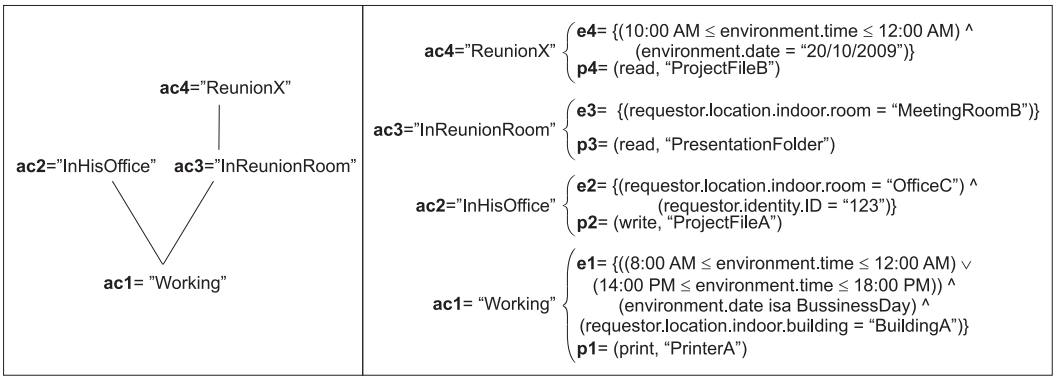


Figure 6.5: Example of access control hierarchies.

expression e_i defined using $GCCL$. Moreover, they have one or more permission associated with them (for reasons of simplicity, we have illustrated only one permission associated with each access context).

InReunionRoom access context is junior to *Working* and thereby inherits the set of permission from *Working access context* when *InReunionRoom* is activated. The *InReunionRoom* can have permissions in addition to those inherited from the *Working* access context. For example, it has the permission (p_4) to read the *PresentationFolder*.

Inheritance of permission is transitive so, for example, the *ReunionX* inherits set of permission from the *InReunionRoom* and *Working* access contexts (i.e., when *ReunionX* is activated, the users how are characterized by it will have the set of permissions $P = \{p_1, p_3, p_4\}$). *InHisOffice* and *InReunionRoom* both inherit permission from the *Working* access context, but each one of these will have different permission directly assigned to it (e.g., *inHisOffice* grants p_1 and p_2 to the users when it is activated).

Access context hierarchies are partial orders ($\geq$) [Pemmaraju 2003], which are characterized by reflexive, transitive, and anti-symmetric relationships between access contexts. This definition is described as following:

Definition 11. *Let ac_1, ac_2, and ac_3 be access contexts (AC).*

- ***Reflexivity:*** *$ac_1 \geq ac_1$ for all $ac \in AC$. Inheritance is reflexive because an access context inherits its own set of permission;*
- ***Antisymmetry:*** *$ac_1 \geq ac_2$ and $ac_2 \geq ac_1$ implies $ac_1 = ac_2$. Antisymmetry access context out access contexts that inherit from one another and would therefore be redundant;*
- ***Transitivity:*** *$ac_1 \geq ac_2$ and $ac_2 \geq ac_3$ implies $ac_1 \geq ac_3$. It is a requirement of access context hierarchy.*

The formal definition of $CxtBAC_1$ is given below:

Definition 12. *The $CxtBAC_1$ model has the following components:*

- *U, AC, R, O, P, $GCCL$, UA, OA, and PA are unchanged from $CxtBAC_0$, $ACH \subseteq AC \times AC$ is a partial order on AC called the access context hierarchy, also written as $\geq$, and*
- *$P(P = 2^{R \times O}) = \{(r, o, s) | r \in R, o \in O, s\,(scope\ of\ permission) \in \{0, 1\}\}$, which each permission is a approval to perform an operation on one resource in a given access context. The scope of a permission can be private (s is equal to 0) or public (s is equal to 1);*

- *access contexts* : $C \rightarrow 2^{AC}$ *was modified from* $CxtBAC_0$ *in order to identify access contexts* $AC(c_i) \subseteq \{ac \mid (\exists\, ac' \geq ac)[(U(c_i),\, ac') \in UA]\}$ *and context* c_i *has the permissions* $\bigcup_{ac \in AC(c_i)} \{p \mid (\exists\, ac' \geq ac[(p,\, ac') \in PA]\}$.

Note that a user is allowed to be part of a situation (c_i) that can activate any combination of access contexts that meets the current context. Also, the set of permission granted to the users in a give situation are those directly assigned to the set of activated access contexts as well as those assigned to its senior access context.

It is sometimes useful in access context hierarchies to limit the scope of inheritance. We could define, for example, private permissions that will be not inherited by junior access contexts. In order to offer users the possibility of limiting the scope of inheritance, the model should differentiate these two types of permissions: *private* and *public* permission.

Permission defined with private scope will be not inherited by junior access context. By contrast, permission with public scope will be automatically inherited by junior access context.

Note that there are two approaches for implementing the support for private and public permission. The first approach favors the hierarchical relationship between access contexts, where permission is public by default, since resource owner/administrator do not have explicitly defined a permission with a private scope. The second approach prioritizes the security of the access control system, where permission is private by default requiring an explicit indication by resource owners/administrator that a given permission has a public scope.

From our point of view, the first approach is simpler to be implemented. In this case, when an *access context* is defined from another access context, only public permissions will be inherited by it.

6.6 CxtBAC$_2$ - Constraints

$CxtBAC_2$ model introduces the concept of constraints, as shown in Figure 6.6. For example, constraints can be used for defining *mutually disjoint* access contexts, such as $Working$ and $Vacation$. This concept is different from the principle called *separation of duties (SoD)* [Gligor 1998] supported by RBAC-based models. In fact, in *CxtBAC* models the users are able to be part of any access context defined in the set of AC, taking into account only the current context. Separation of duties requires that for particular sets of transactions or operations, no single individual be allowed to execute all transactions within the set. For instance, in a company a user could not be able to initiate a payment operation and to authorize it.

Constraints can be applied to C, UA, PA, and OA relations. Constraints are predicates which, enforced to these entities, return a value of *acceptable* or *unacceptable*. The formal definition is described below:

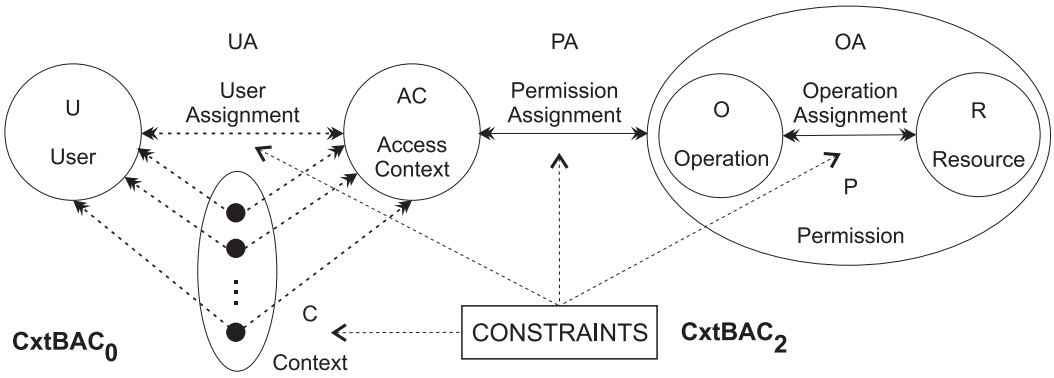


Figure 6.6: $CxtBAC_2$ - $CxtBAC_0$ with constraints.

Definition 13. *$CxtBAC_2$ is unchanged from $CxtBAC_0$ except for requiring that there exist a set of constraints that determine whether or not values of various $CxtBAC_0$ components are acceptable. Therefore, only acceptable values will be permitted.*

Constraints are better viewed according to their classification, then we describe the main set of *CxtBAC* constraints in the following:

- **Mutual exclusion:** mutual exclusion in terms of context C specifies that one user cannot be characterized by both context (e.g., a user cannot be located in his/her room and at home, simultaneously). Mutual exclusion in terms of UA specifies that one user cannot be a member of both access contexts at a given moment. Mutual exclusion in terms of PA specifies that the same permission cannot be assigned to both access contexts and in terms of OA specifies that the same operation cannot be assigned to both resources. Mutual exclusion constraints on PA would prevent the permission from being inadvertently, or maliciously, assigned to a determined access context. The same occurs with mutual exclusion constraints on OA, which prevent the operation from being inadequately assigned to a determined resource;

- **User assignment constraint**: users assigned to various access contexts can be deemed to be acceptable or not. For example, it may be acceptable for a user to be part of *ReunionX* access context and *ReunionY* access context at different moments, but unacceptable to take on both access context at same time. Another type of user assignment constraint, called cardinality constraints, is that an access context can have a maximum or minimum number of members. For instance, the maximum number of person in the *ReunionX* access context could be five. Similarly, the number of access contexts to which an individual user can belong could also be limited. Unlike RBAC-based models, *CxtBAC* supports minimum cardinality constraints. For example, if in a given context there is not the minimum number of users being part of

it, the system will not grant users the associated set of permission (i.e., the access context will not be activated);

- **Prerequisite access contexts**: a user can be assigned to access context X only if she is already assigned to access context Y. For example, only those users who are already assigned to access context *inReunionRoom* can be assigned to access context *ReunionX*. In this example, the prerequisite access context is senior to access context being assigned.

Mutual exclusion constraints can also be applied to contexts. For instance, it could be acceptable a user to be dynamically assigned to two access contexts but the user cannot be active in both access context at the same time. An other constraint on context could limit the number of access context that a user can have active at the same time.

From our point of view, an access context hierarchy can be considered as a constraint, i.e., the constraint in the hierarchy is that a permission assigned to a senior access context must also be assigned to all junior access contexts. In other words, the constraint is that a user assigned to a junior role must also be assigned to its senior roles.

6.7 CxtBAC$_3$ - The core

$CxtBAC_3$ combines $CxtBAC_1$ and $CxtBAC_2$ in order to provide both role hierarchies and constraints. However, there are several issues that arise by bringing these two access control models together. For instance, constraints can be applied to access context hierarchy itself and access context hierarchy is required to be a partial order. Moreover, additional constraints could limit the number of senior/junior access context that a given access context may have, or access contexts could also be constrained to have no common senior/junior access context.

Figure 6.7 shows the $CxtBAC_3$ model. $CxtBAC_3$ is the basic reference model that can be used for defining new access control models and approaches based on context. $CxtBAC_3$ is formally defined in the following:

Definition 14. *$CxtBAC_3$ is unchanged from $CxtBAC_2$ except for requiring that there exist the support for access control hierarchies (ACH) and a new set of constraints applied to that components (ACH).*

Constraints applied to access control hierarchies (ACH) can be of the following types:

- *Mutual exclusion constraint:* an access context cannot be junior of two access context mutually exclusive;

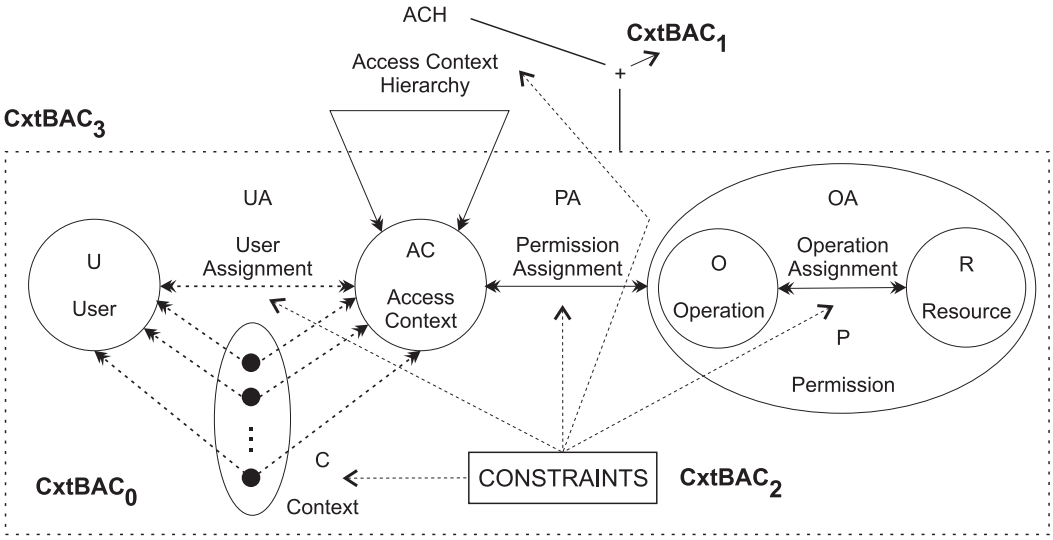


Figure 6.7: $CxtBAC_3$ - The core model.

- *Cardinality:* there is a limited number of senior access context that a junior access context can inherit permission;
- *Prerequisite:* an access context cannot inherit permission from another access context, without a predefined hierarchy relationship among them. Furthermore, a junior access context should inherit the set of permission associated with all its senior access context.

In the following sections, we present the other $CxtBAC$ models defined from the $CxtBAC_3$. Each new model adds a new feature defined in order to reinforce the context-based access control policies. For instance, we have identified the need to support constraints on contexts and access contexts based on the quality of context information used for making access decisions.

6.8 *Q-CxtBAC* - Quality-Aware CxtBAC

$Q - CxtBAC$ model introduces QoC constraints on the $CxtBAC_3$ in order to improve the correctness of context-based access control decisions. QoC constraints could be defined on the *context* and also directly on the *access context* entities of the $CxtBAC_3$ model. To take into account QoC requirements when making context-based access control decisions, might fortify the enforcement mechanism. Thus, QoC-awareness can reduce the probability of making a faulty decision.

For example, if a context information is incomplete the $CxtBAC$-based access control system will be unable to make access decisions. If the context information is incorrect or inaccurate, context-based access decisions made using this information

might grant permission to unauthorized users. Moreover, it may result in security gaps if the quality of context information used by the system is not verified. For instance, the system may allow the context injection accomplished by malicious users in order to get access permission on protected resources.

Figure 6.8 shows this model. We incorporate the support to QoC in the $CxtBAC_3$ by means of *QoC constraints*. QoC constraints can be defined on the context C and access context AC entities. There are two types of QoC constraints supported by the $Q-CxtBAC$: *QoC global constraints*, and *QoC local constraints*. The basic difference between these two types of QoC constraints is the scope of application, changing from global to local scope, respectively. We define these constraints below:

- QoC global constraints (QoC_{gc}): this kind of QoC constraint is defined on context entities (i.e., $c \in C$). The main objectives of using QoC global constraints are the following: i) to avoid unnecessary enforcement operations of access control policy; ii) to increase the security of $CxtBAC$-based access control systems, since context information that does not meet QoC_{gc} will not be used for making access control decisions. QoC_{gc} are enforced on all context information that is considered relevant for making access control decisions. Therefore, QoC_{gc} acts as a filter to prevent $CxtBAC$-based access context systems of using context information with low-quality. As a direct result of using QoC global restrictions, we have: i) reduced the processing cost, because any policies will be enforced when the context does not meet the QoC requirements; ii) certified that the context used for making decision meet QoC global restrictions before assigning permission to users. To summarize, access control policies associated with *access contexts* will be enforced if and only if (iff) the current context meets the QoC_{qc};
- *QoC local constraints (QoC_{lc}):* this kind of QoC constraint is verified on access context entities (i.e., $ac \in AC$). The main idea behind QoC_{lc} is to offer means of defining specific QoC requirements associated with each *access context*. QoC_{lc} may be different from the QoC_{gc}, imposing QoC requirements more/less restrictive than QoC_{qc}. Unlike QoC_{gc}, the set QoC_{lc} can be defined on one or various context objects that characterize an *access context*. Moreover, it is desirable to be able to define policies that grant different sets of permission to users, according to different QoC_{lc} levels associated with the same *access context*.

$Q-CxtBAC$ is defined as follows:

Definition 15. *$Q-CxtBAC$ extends the $CxtBAC_3$ model by adding a set of QoC constraints that determine whether or not QoC associated with context and access context are acceptable. Therefore, only acceptable values will be permitted.*

- *QoC_{gc} and QoC_{lc} are set of QoC thresholds on context and access context, respectively;*

- *Let $CxtObj$ be a context object representing an instance of context information that characterizes an observed entity e. Each $CxtObj_i$ describes an information of the current context c ($c \in C$). $CxtObj_i$ can be associated with one or various QoC indicators ($QoCI$), which describe well-defined quality aspects of that context information.*

The QoC thresholds (QoC_{gc} and QoC_{lc}) can be assessed following one of the following solutions:

- Each context information is individually associated with a set of QoC thresholds.

  Let $QoCI_{set}$ be the set of $QoCI$ that the system is able to evaluate, i.e., $QoCI_{set} = \{QoCI_1, QoCI_2, \ldots, QoCI_n\}$, where n is the number of $QoCI$ supported by the system.

  Therefore, $QoC_{gc} = \{(CxtObj_i, \{QoCI_{1,t}, QoCI_{2,t}, \ldots, QoCI_{n,t}\}) \mid CxtObj_i \in c,\ QoC_{j,t}$ is the threshold value corresponding to the $QoCI_j, j = 1, 2, \ldots, n\}$, and $QoC_{lc} = \{(CxtObj_i, \{QoCI_{1,t}, QoCI_{2,t}, \ldots, QoCI_{n,t}\}) \mid CxtObj_i \in ac,$ $QoC_{j,t}$ is the threshold value corresponding to the $QoCI_j, j = 1, 2, \ldots, n\}$;

- the set of QoC thresholds are applied for any kind of context information. It is defined as following: QoC_{gc} and QoC_{lc} is a set $\{(QoCI_{1,t}, QoCI_{2,t}, \ldots, QoCI_{n,t}) \mid QoC_{j,t}$ is the threshold value corresponding to the $QoCI_j, j = 1, 2, \ldots, n\}$;

- The most practical and recommended solution is to define an average value calculated by using weights associated with each threshold. It is defined formally in the following: Let QoC_{ac} be the average value calculated by using the QoC thresholds. $QoC_{ac} = \frac{\sum_{n=1}^{N} QoCI_{n,t} \times W_n}{\sum_{n=1}^{N} W_n}$, where $W_n (n = 1, 2, ..., n)$ represents the set of weights associated with the set of QoC thresholds. Thus, QoC_{gc} and QoC_{lc} uses the value of QoC_{ac} for enforcing QoC requirements on *context* and *access context*, respectively.

In order to offer means of defining different set of permission P_set according to the quality of context information used in the context constraint expression e associated with access contexts ac, we have modified the format of generic context-based access control policy described in Section 6.4.2. The generic format for defining $Q - CxtBAC$ policies is defined as below:

$$generic\_policy(p_i) = [u, p_{set}, (p_{set_x}, QoC_{lc_x}), (ac, e), bit] \tag{6.3}$$

where x represents zero or various QoC_{lc} associated with the same access context. We fixed in four possible values of x: *zero*, which any QoC_{lc} will be evaluated. In this case, the original format of the policy will be maintained; *low*, which $p_{set_{low}}$ will

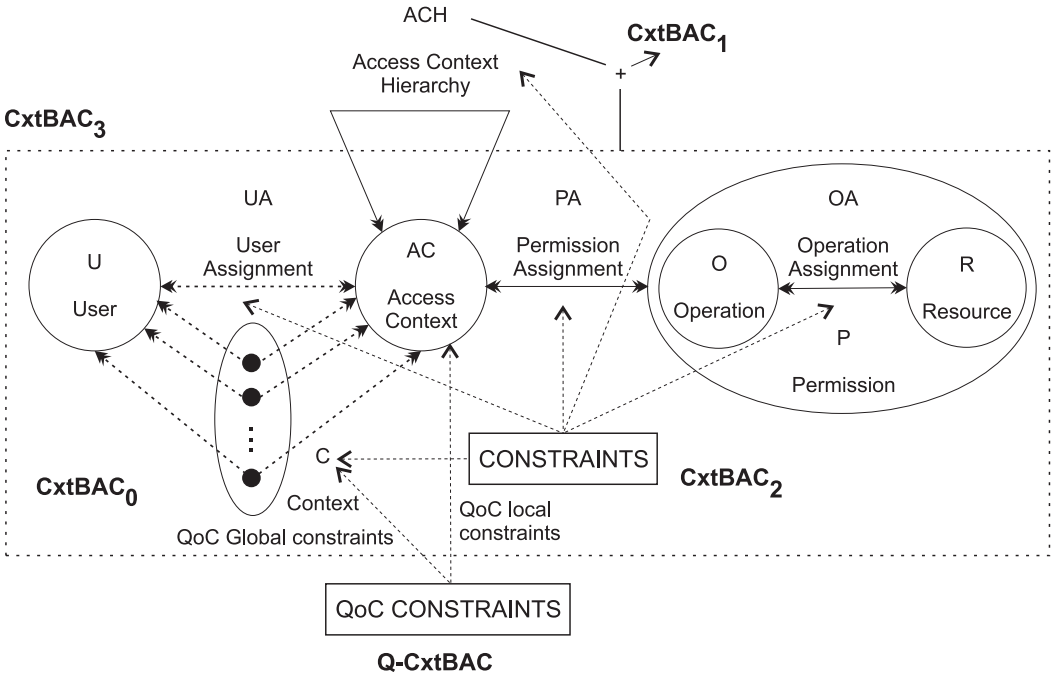


Figure 6.8: $Q - CxtBAC$ - Quality-Aware CxtBAC.

be granted if the access context meets the QoC_{lc} values defined in [0,0.33]; *medium*, which $p_{set_{medium}}$ will be granted if the access context meets the QoC_{lc} values defined in [0.33, 0.66]); and *high*, which $p_{set_{high}}$ will be granted if the access context meets the QoC_{lc} values defined in [0.33, 0.66]). Therefore, (p_{set_x}, QoC_{lc_x}) is the set of permission associated with QoC_{lc} for each different value of x.

Thus, $p_{set} \neq p_{set_{low}} \neq p_{set_{medium}} \neq p_{set_{high}}$, in which p_{set} is the set of permission granted if the access context is activated, and the other sets will be granted if only if (iff) the access context meets their corresponding QoC_{lc}.

6.9 *S-CxtBAC* - Social-Aware CxtBAC

In the real life, the interactions between people occur spontaneously. Generally, we classify the known persons by the social relationship established among us. For example, we can classify the known persons as friend, family, best friend, coworker, etc. From the access control point of view, we can use this social classification to grant different set of permission.

Therefore, we extend $CxtBAC_3$ proposing the $S - CxtBAC$ (Social-Aware CxtBAC) in order to support the definition of social relationships among users. Social relationships can be combined with contextual information for improving the context-based access control policies. In this case, the support to social relationships for defining access policies strengthens the constraints associated with access

contexts.

Figure 6.10 shows this model. In $S-CxtBAC$, users are able to annotate other users with zero or more terms that describe their social relationship. Formally, the $S-CxtBAC$ is defined as in the following:

Definition 16. *$S-CxtBAC$ extends the $CxtBAC_3$ model by adding the support of defining social relationship among users, i.e., resource owners and resource requestor. Thus, the policies can be defined as a function of social relations and restrictions associated with access context.*

- *Social relationship are defined by means of annotations, named Social Annotations (SA). Social annotations can be classified into two types: personal and professional social annotation, describing the social relation among users from personal and professional perspectives, respectively;*
- *Each user has a social network that describes his/her social relationships established with other users;*
- *Let u_i and u_j be two hypothetical users from the set U ($u_i, u_j \in U$). u_i can annotate u_j with zero or various social annotations sa. Moreover, u_i can be annotated by other users with zero or various social annotations;*
- *By default, social annotations are asymmetric and intransitive. Let sa be a social annotation defined by u_i in relation of u_j, which is represented by $u_i \overset{sa}{\rightarrow} u_j$. Considering that exist the $u_i \overset{sa}{\rightarrow} u_j$ and $u_j \overset{sa}{\rightarrow} u_z$ annotations, then we cannot assume that $u_j \overset{sa}{\rightarrow} u_i$ (symmetry) and $u_i \overset{sa}{\rightarrow} u_z$ (transitivity). However, a system that implements $S-CxtBAC$ can support symmetric and transitive social annotations, i.e., $u_i \overset{sa}{\leftrightarrow} u_j$ (symmetry), and if $u_i \overset{sa}{\leftrightarrow} u_j$, $u_j \overset{sa}{\leftrightarrow} u_z$, then $u_i \overset{sa}{\leftrightarrow} u_z$ (transitivity);*
- *Propagation: $S-CxtBAC$ supports the concept of propagation of permission. Propagation is a numeric value indicating the number of hops in the social network between users getting access and other from their social network, following the same type of relationship classification. For instance, if a user defines a policy that grants read access on a $video_file$ to her friends with a Propagation value equal to 2, then the read operation will be granted also to the friends of the her friends. This concept is illustrated in Figure 6.9. Let us note that the set of propagated permission will be granted if only if (iff) the context constraints associated with the access context meets for these other users.*

In order to offer means of defining social-aware context-based access control policies, we have modified the format of generic context-based access control policy described in Section 6.4.2. The generic format for defining $S-CxtBAC$ policies is

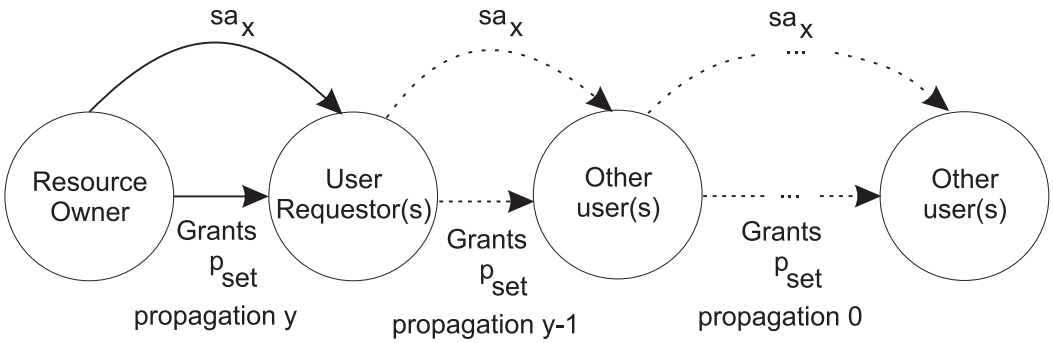


Figure 6.9: Social-aware propagation of permission.

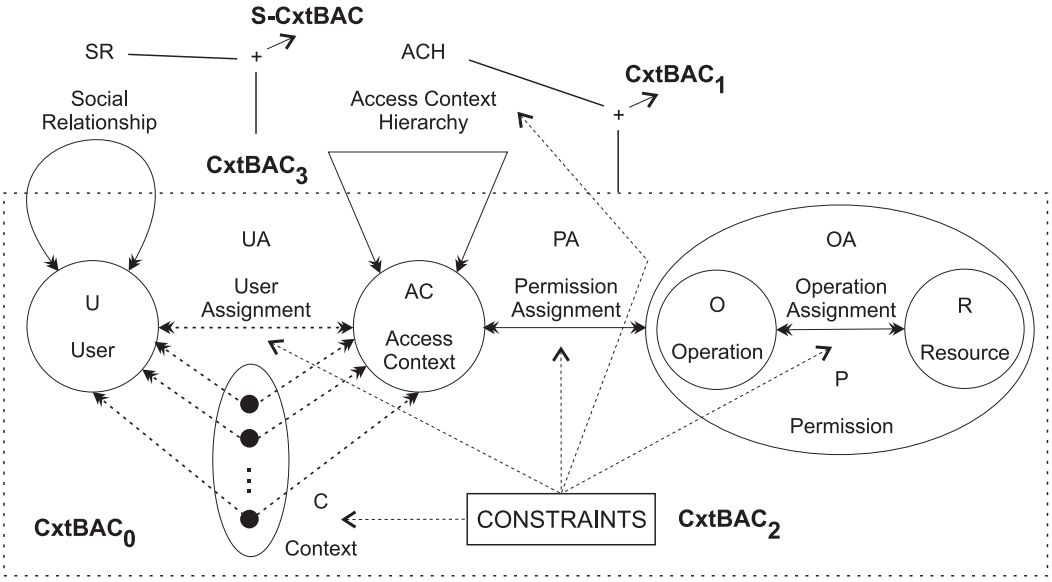


Figure 6.10: $S - CxtBAC$ - Social-Aware CxtBAC.

defined as below:

$$generic\_policy(p_i) = [sbj_{set}, p_{set}, (ac, e), bit] \tag{6.4}$$

where sbj_{set} is a set of subject that gets access permission p_{set} in the access context ac. A subject sbj can be any kind of user's identity supported by $CxtBAC_3$ (i.e., name, pseudonym, group, everyone) and any social annotation sa defined by the user (resource owner).

6.10 *P-CxtBAC* - Privacy-Aware CxtBAC

As described previously, $CxtBAC$ models uses context information for enforcing access control polices. However, context information can describe private informa-

tion about the current situation of users. In this case, users might do not want to disclose this information used for making access control decisions. Moreover, if users disclose his/her context information for a $CxtBAC$-based system, might they want that the enforcement of policies made using his/her context information preserve his/her privacy. The main idea behind this model is to increase the confidence of users in the system.

With this in mind, we propose the $P-CxtBAC$ from the $CxtBAC_3$ by defining functional requirements with regard the privacy of users. In the following, we describe the $P-CxtBAC$ definitions:

Definition 17. *$P-CxtBAC$ extends the $CxtBAC_3$ model by defining functional aspects with regard to the support to privacy requirements of users, such as anonymous enforcement of access control policies.*

- *Privacy Functions (PrivFunc): $P-CxtBAC$ supports the following functions defined on context and access context entities: anonymity, selection, and obfuscation;*
- *Anonymity: by supporting anonymity, the access control system should enforce access control policies preserving the privacy of users, i.e., it should not be possible to associated users with the permission granted by the activated access context. For implementing this functionally, the system should offers means of anonymizing users and policies, but at the same time ensuring the normal execution of enforcing policy process;*
- *Selection of context information: $P-CxtBAC$-based system should use only the set of context information necessary for making access control decisions;*
- *Obfuscation: by supporting obfuscation, $P-CxtBAC$-based should apply obfuscation rules on context information of users in order to use only the information described in the disclosure level necessary to make access control decisions;*
- *Complete control on policies, resources, and context information: users should be able to define access control policies for protecting their resources at anytime and anywhere. They has the complete control on their set of policies, being able to enable/disable them when they deem necessary. The same occurs with their resources and context information. Users should be able to decide when, who, and in what situation they want kept in private or share with other users their resources and context information by means of policies.*

Figure 6.11 illustrates the $P-CxtBAC$ model, which is derived from the $CxtBAC_3$ by adding the *PrivFunc* on context and access context entities.

In order to support anonymous enforcement of access control policies, we have modified the format of generic context-based access control policy described in Section 6.4.2. The generic format for defining $P-CxtBAC$ policies is defined as below:

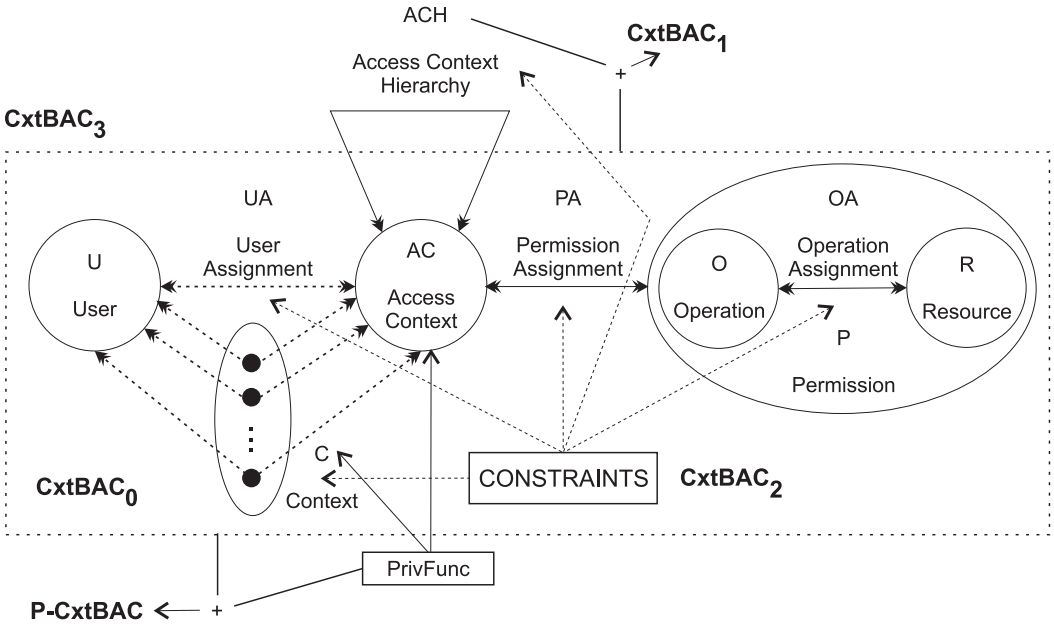


Figure 6.11: $P - CxtBAC$ - Privacy-Aware CxtBAC.

$$generic\_policy(p_i) = [PrivFunc(u), p_{set}, PrivFunc((ac, e)), bit] \tag{6.5}$$

where $PrivFunc$ is a set of operations performed for protecting the privacy of users when enforcing policies associated with access context ac. Therefore, one or various privacy operations (e.g., anonymity, selection, and obfuscation) can be performed on the identity of users and on the context information used by enforcing context constraint expression e associated with the access context.

6.11 *QP-CxtBAC* - Quality and Privacy-Aware CxtBAC

$QP-CxtBAC$ is a model derived from the union of $Q-CxtBAC$ and $P-CxtBAC$. Therefore, $QP - CxtBAC$ enforces access control policies taking into account the quality and privacy requirements of context information. In the following, we describe the $QP - CxtBAC$ definitions:

Definition 18. *$QP-CxtBAC$ is derived from the union of $P-CxtBAC$ and $Q-CxtBAC$ model, by supporting the enforcement of privacy and quality requirements on context and access context entities.*

Figure 6.12 illustrates the $QP - CxtBAC$ model, which is derived from the union between $P - CxtBAC$ and $Q - CxtBAC$. *PrivFunc* and QoC constraints are applied on context and access context entities. There is an execution order for

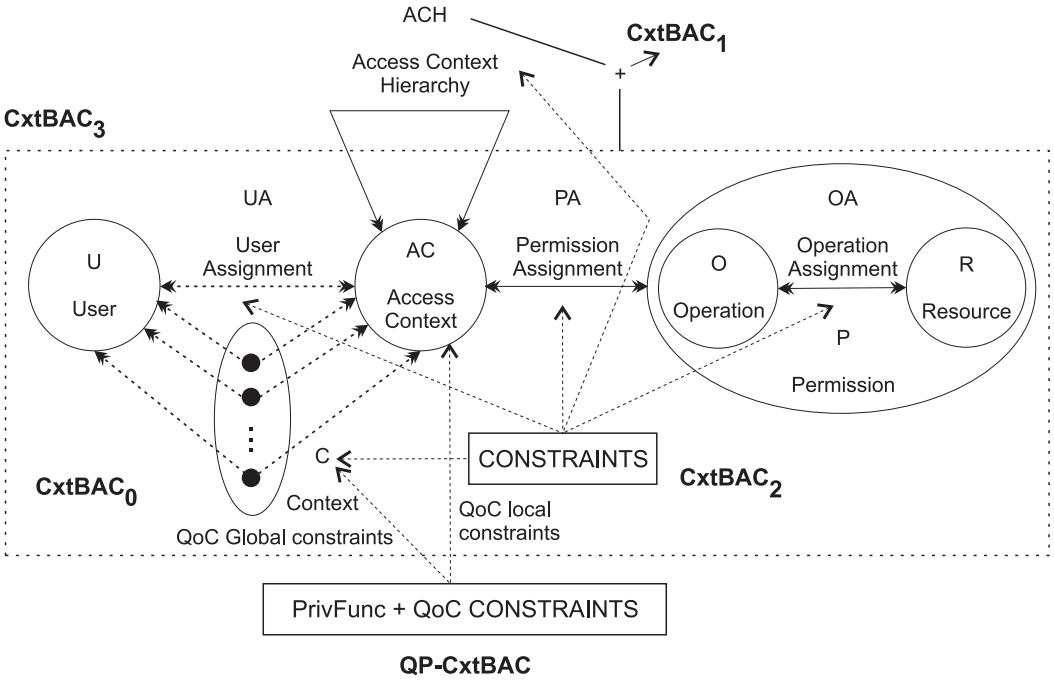


Figure 6.12: $QP - CxtBAC$ - Quality and Privacy-Aware CxtBAC.

these operations that should be respected. First, privacy functions (PrivFunc) are applied, since context information may be modified in order to protect the privacy of users. Then, the quality of context information resulting from the execution of $PrivFunc$ will be verified.

In order to support quality and privacy-aware context-based access control policies, we have modified the format of generic context-based access control policy described in Section 6.4.2. The generic format for defining $QP - CxtBAC$ policies is defined as below:

$$generic\_policy(p_i) = [PrivFunc(u), p_{set}, (p_{set_x}, QoC_{lc_x})PrivFunc((ac, e)), bit] \tag{6.6}$$

$PrivFunc$ and (p_{set_x}, QoC_{lc_x}) were defined in Section, respectively, therefore we will omit here these definitions.

6.12 Administration of $CxtBAC$ models

Management of access control systems that implements the $CxtBAC$ core model ($CxtBAC_3$), consist of performing the following set of activities:

1. Identifying the context information that the system is able to gather for char-

acterizing the situation;

2. Defining access context and its hierarchies;
3. Defining the constraints;
4. Defining context-based access context policies;
5. Reviewing policies, access context, access context hierarchies and constraints during the entire life cycle of the system.

If an access control system uses as basis $S - CxtBAC$, $P - CxtBAC$, $Q - CxtBAC$, and $QP - CxtBAC$, other administration operations should be performed by the user or administrator. Such operations are described as follows:

- $S - CxtBAC$: users should annotate the persons in their social network in order to be able of defining social-aware context-based access control policies;
- $P - CxtBAC$: users should define policies with regard to their privacy requirements for protecting their context information. Moreover, the user or administrator should configure the $PrivFunc$ to be executed on context, access context, and policies;
- $Q - CxtBAC$: the user or administrator should define QoC global threshold and QoC local threshold on context and access context, respectively. When defining a policy, the user or administrator is able to use QoC local threshold for verify the quality of context information;
- $QP - CxtBAC$: it should performs the operations described previously for the $P - CxtBAC$ and $Q - CxtBAC$, following this order.

6.13 Examples of *CxtBAC* Policies

In this section we describe some examples of $CxtBAC$ policies. For demonstration purposes, we consider only access policies based on $CxtBAC$ core model.

Based on the concept of *access context* that characterize the situation of *access entities* and the environment around them, we are able of defining $2^4 - 1$ different types of context-based access policies. It results from the combination of context information associated with each observed entity (i.e., context of resource owners (CxtOwn), context of resource requestor (CxtReq), context of resource (CxtRes), and context of environment (CxtEnv)).

The $CxtBAC$ policy examples are defined using the generic representation format, described previously in Section 6.4.2, and the generic context constraint language, described in Section 6.4.1. Therefore, in the examples we demonstrate the expressiveness of access control policies supported by $CxtBAC$ models.

- **CxtOwn-based access policy:** this type of access control policy takes into account only context information that characterizes the situation of resource owners.

  Example: a patient (owner) may grant read permission on her medical records to any doctor if she is in a life-threatening situation characterized by a sudden drop in her blood pressure ($blood\_p$) or in her heart rate ($heart\_r$).

  $ac1 =$ "$life\_threatening$"
  $e1 = \{(owner.blood\_p < 85) \vee (owner.heart\_r < 60)\}$;
  $p1 = (doctor, (read, medicalrecords), ac1, e1, true)$;

  In this example, the access control system takes into account only context information of the resource owner (her health conditions) for making the access control decision.

- **CxtReq-based access policy:** this type of access control policy takes into account only context information that characterizes the resource requestor.

  Example: a user grants read access on his $presentation\_file$ to everyone located in the meeting room X.

  $ac2 =$ "$inMeetingRoomX$"
  $e2 = \{(requestor.location.indoor.room = \text{"}meetingRoomX\text{"})\}$;
  $p2 = ((read, presentation\_file), ac2, e2, true)$;

  Let us note that in this example the access control policy is dynamic, i.e., it will be activated in different situations, since the context constraint expression *e2* do not have a period time of validity. Moreover, it is not explicitly described the person to whom access is being allowed, so anyone who meets the context constraint conditions imposed by *e2* may get permissions.

- **CxtRes-based access policy:** this type of access control policy takes into account only context information that characterizes the resource.

  Example: a user grants read access on her photos taken in Paris to everyone.

  $ac3 =$ "$PhotosTakenInParis$"
  $e3 = \{(resource.location.outdoor.country =$ "$Paris$"$) \wedge$
  $(resource.type =$ "jpg"$\}$;
  $p3 = (everyone, (read, resource), ac3, e3, true)$;

  In this case, all protected photos should be annotated with contextual information that describes the location where they were taken.

In the following, we present examples of access control policies based on context information that characterizes simultaneously one or more *access entities*. We generalize the use of context information for defining context-based access control policies, showing the expressive power of *CxtBAC*-based policies. We do not intend to present a exhaustive list of policy examples resulting from the combination of context information associated with each access entities. However, these examples can be used as basis to define new types of context-based access policies.

- **CxtOwn and CxtRes-based access policy:** a user grants read access on documents created at her office room (context of resource) to project team members if she is at her desk (context of resource owner).

  $ac4 =$ "$atDesk$"
  $e4 = \{(owner.location.indoor.room = \text{“}office\_322\text{”}) \wedge$
  $(resource.location.indoor.room = \text{“}office\_322\text{”}) \wedge$
  $(resource.type = \text{“}doc\text{”}\};$
  $p4 = (ProjectTeam, (read, resource), ac4, e4, true);$

- **CxtOwn and CxtReq-based access policy:** a user grants read access on $photo\_collectionX$ to everyone located near him.

  $ac5 =$ "$InProximity$"
  $e5 = \{(requestor.device.bluetoothAddr\ in$
  $owner.nearbyDevice)\};$
  $p5 = (everyone, (read, photo\_collectionX), ac5, e5, true);$

- **CxtReq and CxtRes-based access policy:** a user grants read access on photos taken in Paris (context of resource) to everyone located in this city (context of resource requestor).

  $ac6 =$ "$PhotosInParis$"
  $e6 = \{(requestor.location.outdoor.country = \text{“}Paris\text{”}) \wedge$
  $(requestor.location.outdoor.country = resource.location.outdoor.country)\} \wedge$

  $(resource.type = \text{“}jpg\text{”}\};$
  $p6 = (everyone, (read, resource), ac6, e6, true);$

- **A generalized context-based access policy:** a user grants write access on photos taken in Paris (CxtRes) to everyone located in this city (CxtRes), but only when she is located also in Paris (CxtOwn).

  $ac7 =$ "$PhotosInParisForVisitors$"
  $e7 = \{(requestor.location.outdour.country = \text{"}Paris\text{"}) \wedge$
  $(requestor.location.outdour.country = resource.location.outdour.country) \wedge$
  $(requestor.location.outdour.country = owner.location.outdour.country) \wedge$
  $(resource.type = \text{"}jpg\text{"}\};$
  $p7 : (everyone, (write, resource), ac7, e7, true).$

In this policy, we use context information associated with each access entity (resource owner, resource requestor, and resource) in order to grant access permission on the photos taken in Paris.

6.14 Implementation approaches

We have identified three implementation approaches for *CxtBAC*-based access control services: *peer-to-peer* approach, *server-based* approach, and *server-client* approach. Basically, they differ among themselves according to the location of entities in charge of requesting access permission on protected resources and enforcing access control policies. These entities are named PEP (Policy Enforcement Point) and PDP (Policy Decision Point) components, which are in charge of querying and enforcing process of access control policies, respectively.

In the peer-to-peer approach, each pervasive device (e.g., smartphones, notebook, netbook, etc) has an instance of PEP and PDP entities running on the device (client side). Consider the scenario described in Figure 6.13. The pervasive device 1 request access to a resource (req(r1)) by using the PEP. As the requested resource is located in the pervasive device 2, the PEP of pervasive device 1 requests the resource to PDP of pervasive device 2, which makes an access control decision by enforcing the access policies. The same process occurs when the pervasive device 2 requests access on the resource 2.

In the server-based approach, pervasive devices do not have any instance of PEP and PDP entities running on the device, i.e., both PEP and PDP are running in the server side. Figure 6.14 illustrates this approach. Users, by means of user-friendly application interfaces, request access on protected resources. Then, this application rewrite the request and send it to the PEP running in the server. Upon receiving this request, PEP requests the PDP to enforce it. Once the request is handled, PDP answers the PEP that resends the answer to the application, granting/denying access on the protected resource.

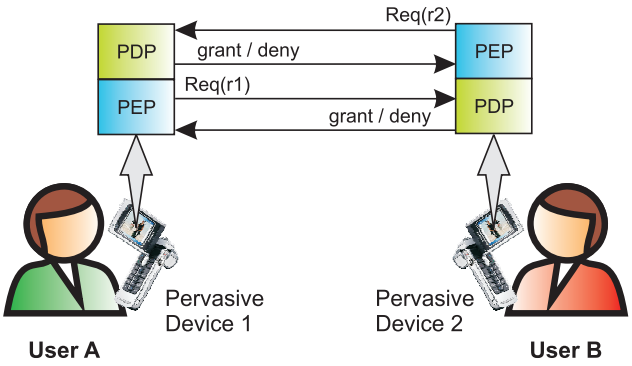


Figure 6.13: Peer-to-peer approach: Pervasive devices request/enforce policies.

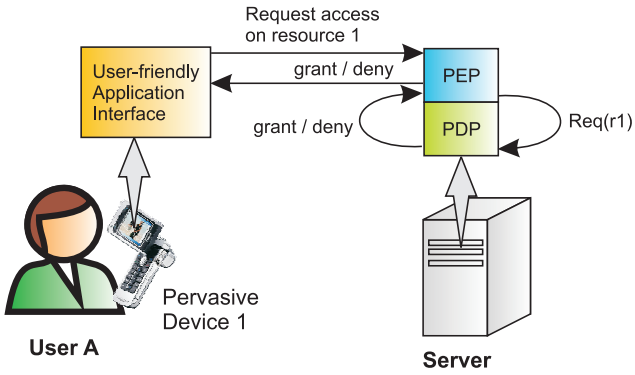


Figure 6.14: Server-based approach: PEP and PDP running on the server side.

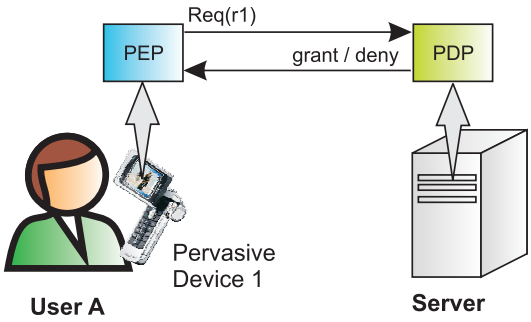


Figure 6.15: Client-server approach: PEP on mobile devices and PDP on the server.

In the client-server approach, pervasive devices has an instance of PEP running on the device (client side), while the PDP is running in the server side. Figure 6.15 illustrates this approach. Pervasive devices, by means of the PEP running in the device, request directly the PDP running in the server. Upon receiving this request, PDP answers the PEP that resends the answer to the application, granting/denying access on the protected resource.

In Chapter 8 we describe an instance of $S - CxtBAC$ developed for supporting multimedia application, which is based on the server-based approach described previously. We opted by this approach to reduce the processing load on the client side, since the entities in charge of enforcing access control policies is running in the server side.

6.15 Context-based access control policies

CxtBAC can be implemented to support one of the three following types of context-based access control policies: *mandatory context-based access control policies*, *discretionary context-based access control*, and a hybrid approach that supports both *mandatory and discretionary context-based access control policies*. They differ according to the actor(s) in charge of performing the policy administration operations, as described below:

- *Context-based mandatory access control policies:* by implementing the $CxtBAC$ to support this kind of policies, the system administrator will be in charge of defining access control policies. These policies are named system-level context-based access control policies. Usually, the user authentication is performed based on the situation, i.e., situations that should grant access permission to users are prefixed by the administrator. For example, an administrator can define a policy to grant access permission on a service to the users located in the train during their trip, or yet to consumers located in a fast-food. In this case, the location of users and a receipt of these services (e.g., the train ticket

and the payment receipt) could be used as contextual information for authenticating users in order to grant access permission on the protected service;

- *Context-based discretionary access control policies:* by implementing this kind of policy, resource owners are in charge of defining access control policies to protect their own resources. For example, a user would grant read access on his/her videos to his/her friends who were present at creation time of these videos. This is a *user-level context-based access control policy*;
- *Mandatory and discretionary context-based access control policies:* by supporting this kind of policy, the access control model should support policies defined by administrators and users, simultaneously (i.e., user-level and system-level access control policies). In this case, resource owners are able to define user-level access control policies to protect their resources, and the administrator is able to impose constraints upon these policies. Moreover, the administrator can define system-level access control policies in order to protect resources belonging to an organization (i.e., it do not belong to a specific user), which are available in the environment.

6.16 Enforcing context-based access control policies

In order to implement a *CxtBAC-based access control system (CxtBACS)*, we need to deploy a enforcement process of context-based access control policies. We have identified three possible enforcement approaches that could be used by *CxtBACS*: *passive*, *active*, and *hybrid* enforcing mechanisms.

To explain clearly the differences between these approaches, we present sequence diagrams that describe the messages exchanged by the entities in charge of enforcing access control policies. We have defined three set of entities: *requestor*, *CxtBACSystem*, and *CxtManagementFramework*.

Requestor represents a user requesting access permission on protected resource by means of a *Access Control Client Application (ACCA)*. *CxtBACSystem* represents the implementation of a *CxtBAC* model, and *CxtManagementFramework* the entities in charge of managing context information.

Before presenting these approaches, we need to present the format of an access request and a pseudo algorithm in charge of evaluating access requests.

6.16.1 Evaluating access requests

CxtBACSystem makes access decisions by processing access requests. We define an access request (Req_A) as a triple *(req, perm, cv)*, where:

- *req* is a requestor entity who issues this access request;

- *perm* is the permission that this requestor wants to acquire;

- *cv* is a set of values for every context information c_i characterizing the observed entities at request time. That is, $cv = \{v_1\ of\ c_1,\ v_2\ of\ c_2, ...,\ v_n\ of\ c_n\}$, where $\{c_1,\ c_2, ...,\ c_n\}$ is the set of context information described by the *access context*.

A request of access $Req_A(req, perm, cv)$ is granted if only if (iff) there exists an access control policy $Pol_i = (u, p_{set}, (ac, e), bit)$ from the policy set (Pol_{set}), such as $bit = 1$ (i.e., the policy is enabled), $req \in u$, $perm \in p$, and e evaluates true under cv (i.e., the e returns true when all values of c_i are replaced in the context constraint expression e associated with ac).

From this definition, we designed algorithms to determine whether access requests are authorized or not, according to the current context values of *access context*. We propose a solution divided into two algorithms: i) the first algorithm (see Figure 6.16) is in charge of identifying the set of candidate policies (Pol_{setC}) from the policy set defined by resource owners/administrators (Pol_{set}); ii) the second algorithm (see Figure 6.17) verifies if the context constraint expression e associated with each candidate policy from the set Pol_{setC} evaluates true, according to the current *access context*.

Algorithm 1 verifies for each Pol_i of Pol_{setC} if *req* of Req_A is in the u of Pol_i. In addition, it verifies if the *perm* of Req_A is in p_{set} of Pol_i. If these two conditions are true, the Pol_i is a candidate policy. After running the algorithm for identifying the candidate policy set (Pol_{setC}), this set will be evaluated by the *Algorithm 2 (EvaluateContextConstraint)* in order to verify if the expression e is true for the current *access context*, granting/denying access permission to the requestor entity.

6.16.2 Passive approach

In the passive approach, the enforcing process of access control policies is executed only at request time of a protected resource. For each access request received on a protected resource, the access control system should identify the current access context in order to enforce the affected access control policies.

Therefore, the access control system identifies, at request time, the current context of resource owner, resource requestor, resource, and the environment in order to evaluate the context constraint expressions associated with the affected access control policies. Once the policies are enforced, the access control system grants/denies the assigned permissions to the resource requestor.

Passive approach is the simpler and lightweight solution for enforcing context-based access control policies, since the system does not enforce continually the access control policies. However, this approach has a security breach on the resource discovery process. In order to offer requestors the possibility of requesting access

Algorithm 1: EnforceRequest(Req_A)

Require: Req_A
Ensure: Permit/Deny
Pol_{setC} ← Null *{Initializing the set of candidate policies}*
Result ← Deny *{Initializing the result}*
{Identifing the set of candidate policies}
for each $Pol_i \in Pol_{set}$
  if (enable_bit of Pol_i is true)
    if (req of $Req_A \in$ re of Pol_i) **and**
      (perm of $Req_A \in p_{set}$ of Pol_i)
       $Pol_{setC} \leftarrow Pol_{setC} + Pol_i$
    end if
  end if
end for

{Evaluating context constraints of candidate policies}
for each Pol_j in Pol_{setC}
  if (EvaluateContextConstraints(e of Pol_i) is true)
    result ← Permit
    Break
  else
    result ← Deny
  end if
end for
{Granting/denying access}
return result

Figure 6.16: Algorithm 1: Enforcing request of access.

Algorithm 2: EvaluateContextConstraint(e)

Require: e, cv
Ensure: true/false
result ← false
{Verifing each access context condition}
for each $acc_i \in$ e
  v ← value of c_i described in cv
  if (acc_i is false with the value v)
    return false
    break
  end if
end for
return true

Figure 6.17: Algorithm 2: Enforcing context constraint expression e.

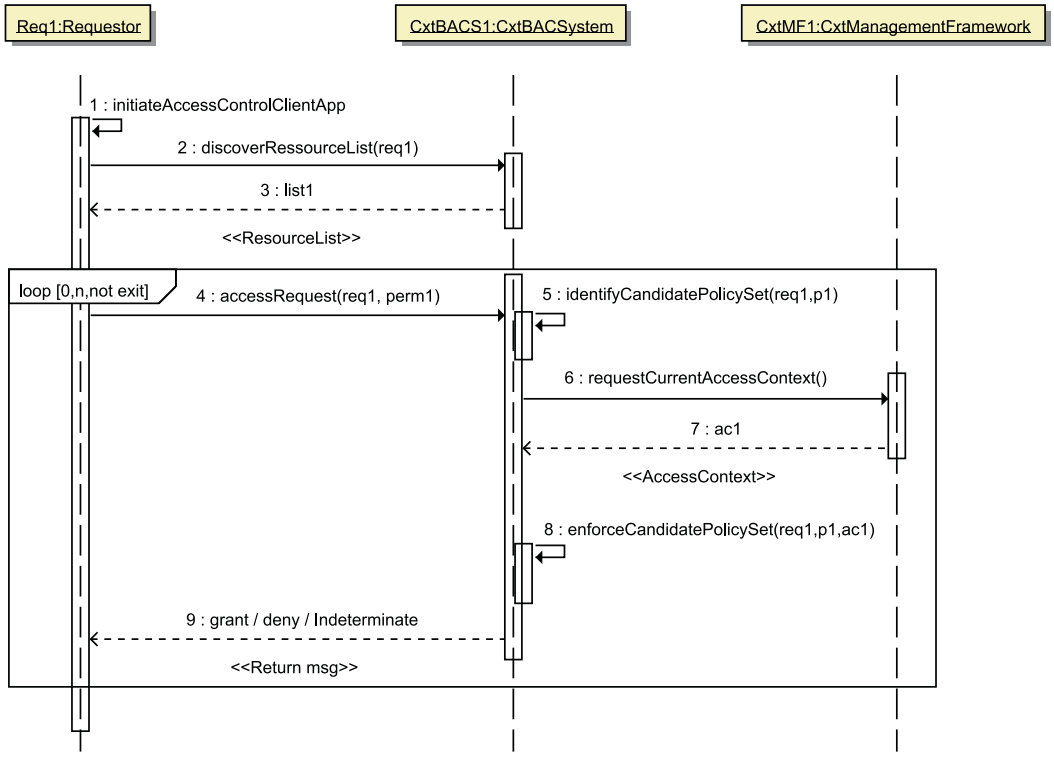


Figure 6.18: Passive approach for enforcing context-based access control policies.

permission on any protected resource, *CxtBACSystem* needs to disclosure the complete list of protected resources to them.

In this case, requestors are able to know the complete list of protected resource even if they do not have access to them at this moment. Thus, access control system grants/denies permission to requestors only after receiving a request on a protected resource. *CxtBACSystem* identifies the current access context in order to enforce the access control policies associated with the requested resource.

Figure 6.18 shows the sequence diagram of the passive approach for enforcing context-based access control policies. In the step 1 (*initiateAccessControlClientApp*), the requestor (req1) initiates the *ACCA* in order to request access on a protected resources. We consider requestors are authenticated before executing this step (or they has been authenticated by the *initiateAccessControlClientApp*). We do not describe the authentication process of users here because it is out of the scope of this work.

The requestor, by using the *ACCA*, executes automatically the discover resource action (*message 2: discoverRessourceList(req1)*) by sending the identity of requestor to *CxtBACSystem* (CxtBACS1). Then, *CxtBACS1* identifies the set of resources that the requestor possibly could access by using only her identity, i.e., *CxtBAC*1 did not take into account the current access context. In fact, the returned list

contains only the resources protected by policies assigned implicitly to the requestor (i.e., to her identity, her group, etc) and those that do not make reference to any user (i.e., granting permissions to everyone in a given situation).

CxtBACS1 sends the list of resources (e.g., a XML file describing the URI addresses of protected resources) to the ACCA (*message 3 : list1*). This list is presented to the requestor in a legible way. These first three steps (messages 1, 2, and 3) are performed only one time in the all cycle life of ACCA. After executing these steps, ACCA should wait for explicit access requests performed by the requestor.

From the list of resources presented by ACCA, the requestor can try to get access on any protected resource. When requestor tries to get access on a protected resource, ACCA sends a message that contains the identity of the requestor and the requested permission (i.e., perm1 = (operation1, resource1) to the *CxtBACS1* (*message 4: accessRequest(req1, perm1)*).

Then, *CxTBACS1* uses the identity of requestor (req1) and the requested permission ($operation1, resource1$) to select from the existing access control policy set a subset named *candidate policy set*. In fact, CxtBACS1 verifies each policy Pol_i from the existing Pol_{set} if *req1* is in the u or $\in User(Pol_i)$. When this verification returns a true value, then it verifies if *perm1* is in the $Permission(Pol_i)$. If these two conditions return a true value, Pol_i is considered as a candidate policy and it is inserted into the Pol_{setC}. This process for selecting candidate policies is realized by the *IdentifyCandidatePolicySet* algorithm presented in Figure 6.16 (*message 5: identifyCandidatePolicySet(req1, p1)*).

After finishing this process, *CxtBACS1* requests the current access context to the *CxtMF1* (*message 6 : requestCurrentAccessContext()*) in order to enforce the candidate policy set. *CxtMF1* sends the current access context (*message 7 : ac1*) to the *CxtBACS1*. Then, this last one enforces the Pol_{setC} (*message 8 : enforceCandidatePolicySet(req1,p1,ac1)*) by executing the *enforceCandidatePolicySet* algorithm presented in Figure 6.17.

There exist a loop in the ACCA, in which for each access request the messages 4, 5, 6, 7, and 8 are executed in order to verify the access request. This loop will be finished when the requestor quit the ACCA.

6.16.3 Active approach

The active approach is more complex than the passive approach for enforcing context-based access control policies. In order to implement a $CxtBACS$ that supports that active approach, we need to continuously verify the current access context in order to dynamically enforce the set of candidate access control policies.

When $CxtBACS$ identifies access control policy activated according to the current context, it should notify the affected users. In this case, ACCA shows in the activated resource list only the resource in which there exists one or various granted

permission assigned to the user. Therefore, the task of assigning permission to users is dynamically context-dependent.

The active approach requires a notification service that constantly updates the list of resources available to the affected users, taking into account the permissions assigned to the context conditions (i.e. access policies) that match the current context.

6.16.4 Hybrid approach

In the hybrid approach, the enforcement process is conducted in two phases. Firstly, the user should request a list of activated resources, according to the current access context. Posteriorly, users request access to any resource from this list. *CxtBACS* requests the current context to the *CxtMF* in order to verify again the granted permissions to the requestor.

This approach eliminates the security problems present in the passive approaches, beyond reducing the processing load and the difficulty for implementing the active approach.

6.17 Implementation Requirements

In order to fully exploit the expressiveness of CxtBAC Models, *CxtBAC*-based access control systems (CxtBACS) need to support each type of context information (i.e., context of resource, resource requestor, resource owner, and environment) that is relevant for making context-based access control decisions. It can be met by the following requirements:

1. It is required to define an access context model taking into account the access control requirements of an application scenario. This model should describe the context information dimensions needed to support the context-based access control policies;

2. It is required an entity in charge of context management associated with resources, resource requestors, resource owners, and environment. This entity should be able to provide context information with quality and security to CxtBACS, preserving the privacy of users;

3. It is required the support to context sensing and annotation operations in order to annotate resources with context information at creation and request time. Static resources should be annotated at creation time, while dynamic resources at request time. Moreover, users should be able to annotate other users with terms describing their social relationship. Therefore, these mechanisms should assist *CxtBACS* in the contextual annotation task of resources;

4. It is required to assure the security, privacy, and quality of context information used for making context-based access control decisions. Using compromised context information may result in incorrect access control decisions;

5. *CxtBACS* needs to identify the current access context at request time of any protected resources, in order to decide whether or not it should grant users the assigned permission. Moreover, it should be possible to suspend the access control policies activated by an access context when the situation changes to a state where its context constraint expression is not more true [Yokoyama 2006].

6.18 Conclusion

We have described in this Chapter a family of context-based access control models for pervasive environments. *CxtBAC* models can be instantiate for implementing context-based access control systems for pervasive environments. According to the specific requirements, such as QoC, privacy, and social-aware support, access control systems could implement the correspondent *CxtBAC* model. *CxtBAC* do not support separation of duties, however, *CxtBAC* supports the principle of least privilege. Moreover, *CxtBAC* models can be instantiate to support discretionary and mandatory access control policies. Compared with traditional access control models such as *RBAC*, *CxtBAC* introduces several features:

- The set of access permission are dynamically variable for a *CxtBAC* system, while for a conventional system this set is constant. An access permission to a resource is based on certain thresholds that depend on context information, such as users behavior. CxtBAC-based access control systems alter dynamically the access permission to prevent potential abuse of privileges, identifying deviation from usual behavior or falling outside some context;

- *CxtBAC* allows making decision access based on multiple situational information (e.g., location of users, time, velocity) instead of a single one for conventional systems (identity). This offers flexibility but also complexity that should be dealt with in an appropriate way;

- Multiple administrative entities (e.g., each user could be an administrator) will be involved in a *CxtBAC* system, whereas only a single entity is involved in a conventional access control model. This requires an infrastructure that facilitates trusted cross-domain context exchange and peer-to-peer interactions;

- Although the *CxtBAC* model is more flexible and increases the expressiveness of access control policies, the set of access policies for granting permissions must be defined in advance, i.e., they are not generated on the fly by an access control engine that explores context information, e.g., concerning resource usage patterns, learns and derives the most appropriate access control policies.

In order to make the access control process truly dynamic and transparent, context-based access control policies need to be generated on the fly as well. We have introduced a generic representation of context-based access control policies that supports the definition of context constraints as part of the policy. However, it is necessary to use the access control behavior learned over time to functionally adapt those policies, resulting into a more robust, flexible, and scalable access control solution.

Chapter 7

CxtMF: Context Management Framework

Résumé: *Ce chapitre décrit l'architecture de gestion d'information contextuelle proposée, ainsi que les modèles contextuelles sémantiques et les méthodes d'estimation des indicateurs de qualité associés à cette information. Nous décrivons les besoins et l'impact du support liés au traitement de l'information de qualité du contexte dans chaque couche de gestion. Ainsi, ce chapitre discute des résultats d'implémentation et d'évaluation de l'architecture proposé pour la gestion contextuelle avec le support aux modèles sémantiques d'information (qualité et contexte) et les méthodes d'estimation des indicateurs de qualités.*

Contents

7.1 Introduction

In order to deploy context-based access control services that implements elements of the family of CxtBAC models, we need to integrate a context management service to

provide QoC-enriched context information. In this chapter we describe the proposed Context Management Framework, named *CxtMF*. *CxtMF* is composed by services in charge of gathering, deriving, inferring, protecting, and providing context enriched with quality information to context consumers, such as context-based services (e.g., CxtBAC-based system) and context-aware applications.

As discussed in Chapter 4, QoC has a real impact on the behavior of any context-aware application and service. Using context information with unexpecting quality increases the risk of unsuitable context-based actions. Let us note that it is very important to consider QoC in the various layers of context provisioning processes. In our framework, QoC is used for:

1. *Supporting Global and Local QoC thresholds:* the context management framework supports Global and Local QoC thresholds in order to provide context information that meets specified quality requirements. Global QoC thresholds are defined in the level of management and are applied for all context consumers. Local QoC thresholds are defined individually to meet QoC specific requirements of each context consumer;

2. *Selecting of sensor and context providers:* QoC is used to select sensors and/or context providers in order to discard raw context data from sensors/context providers that does not reach the minimum quality level fixed by QoC thresholds. Moreover, context providers and sensors that do not reach the predefined QoC thresholds are added into the black list of registered context providers/sensors;

3. *Improving context-based applications:* by taking into account QoC information, applications and services can improve its context-aware reasoning and decision making by reducing the probability of incorrect adaptation. In our case, QoC is used to improve the enforcement of context-based access control policies.

Despite the existence of others context management frameworks and middlewares proposed by the scientific community, such as the *Context Toolkit*[1], CASS [Fahy 2004], Hydrogen [Hofer 2002], and SOCAM [Gu 2004], we decided to define and develop a new framework in order to have flexibility to easily integrate the management of the proposed context and QoC models, as well the proposed QoC measurement approaches.

However, nothing prevents the ideas and QoC measurement mechanisms proposed here of being embedded in other existing solutions. The proposed QoC assessment mechanisms are quite general and can be implemented in accordance with specific requirements of other context management frameworks.

Therefore, our intention here is not to improve features of context management

[1] http://www.cc.gatech.edu/fce/contexttoolkit/

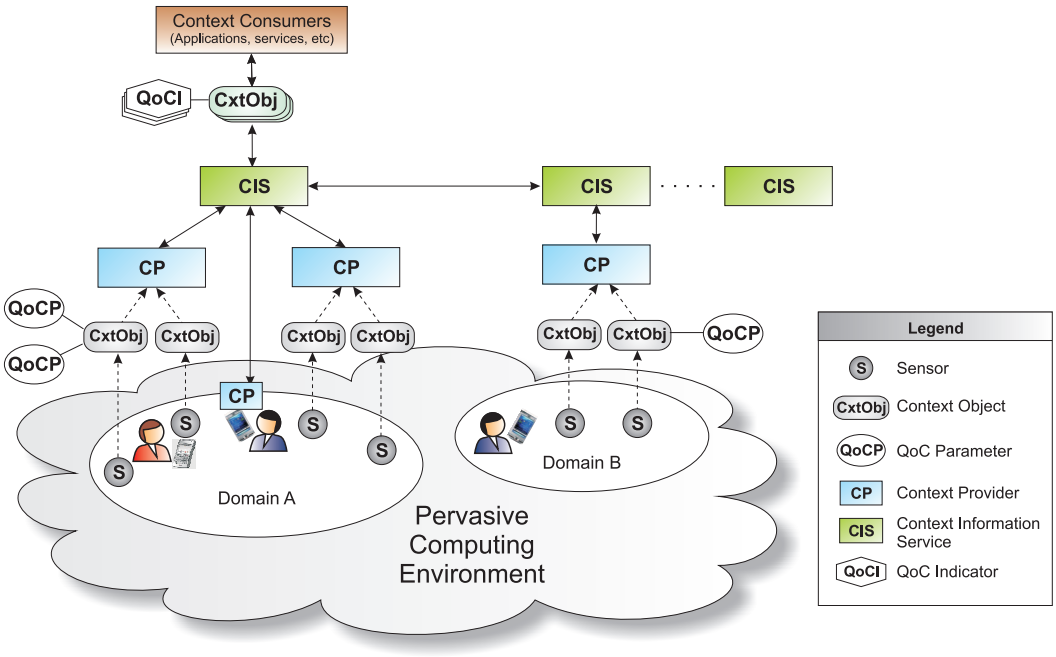


Figure 7.1: Pervasive Computing Environment Overview.

process itself, but simply to exploit the context management layers to enrich context with quality in order to verify if QoC information meets the QoC thresholds.

The reminder of this Chapter is organized as follows: first, we describe the $CxtMF$ reference architecture and the functional aspects of its components. Then, we present the context and QoC models proposed to represent semantically context information and its associated quality dimensions, respectively. Finally, we present preliminary evaluation results of our implementation.

7.2 Reference Architecture of Context Management Framework (CxtMF)

Figure 7.1 illustrates a hypothetical pervasive computing environment used as basis for defining our context management architecture. In this scenario, we have various sensors S distributed on the environment and on embedded mobile devices (e.g., smartphone equipped with GPS sensor, Bluetooth, Wifi, etc) producing context information. Context information is provided to context consumers, such as context-aware applications and services.

Let $CxtObj$ be an object that represents a context information c and its value (e.g., location) about a real world entity E (e.g., user). $CxtObj$ are collected by sensors S that can be classified as *physical* (e.g., smart-phone equipped with GPS sensor, Bluetooth, Wifi, Sun Spot) or *logical*: *physical sensors* (i.e., pieces of hard-

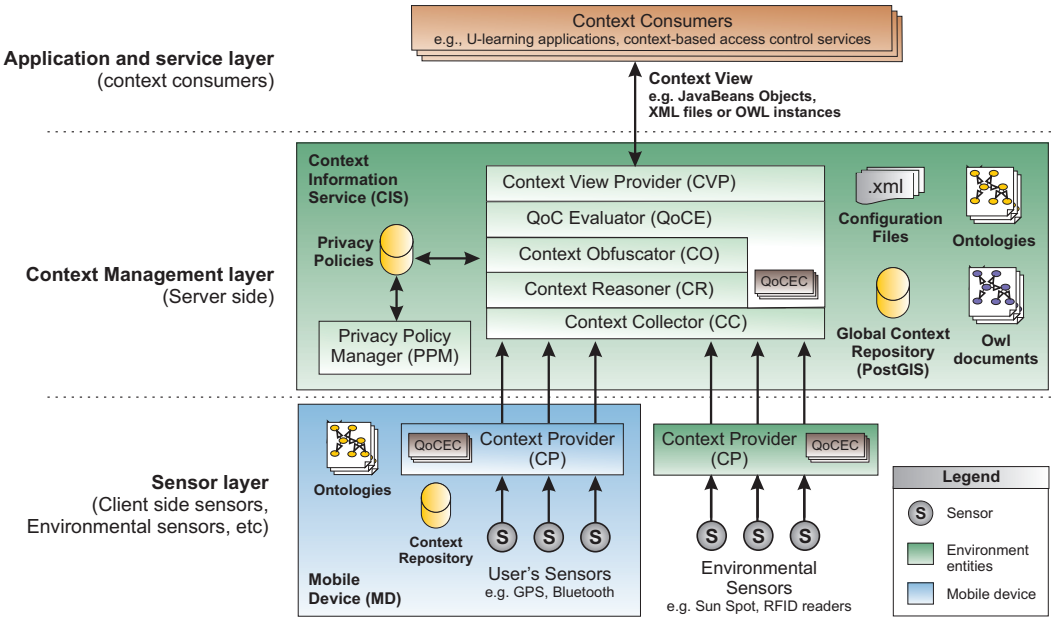


Figure 7.2: Reference Architecture of CxtMF.

ware) capture information from a host located in the environment (e.g., location, temperature, noise-level, light, proximity of another device); *logical sensors* consist only of software components. Such logical sensors are used to gather information that can be obtained from users (e.g., an application that ask the users about her/his current activity) or system internal sources (e.g., log-files, status of applications and services, network information).

Sensors belong to different domains in the environment (see in Figure 7.1 sensors belonging to Domain A, and Domain B). A domain represents a pervasive sub-environment associated with an organization, such as a company, a university, etc. *CxtObj* are collected, aggregated and stored by Context Providers (CP) distributed in such domains. *CxtObj* are associated with some QoC information (QoCP) sensed from the environment, named QoC parameters, which is used to evaluate the corresponding QoC indicator (QoCI). For instance, the *captureTime* (information sensed from the environment) is a *QoCP* used to measure the *QoCI up-to-dateness* of the context information represented by a *CxtObj*, such as location.

Figure 7.2 illustrates the reference architecture of our context management framework, named *CxtMF*. *CxtMF* is defined to support context-aware applications and services in PCE, such as instances of *CxtBAC* model. The main idea behind the *CxtMF* is providing context information to context consumers, taking into account the quality aspects of context information in all steps of context management process.

The main entities of *CxtMF* are the *Context Providers (CP)* and *Context Information Service (CIS)*. Context Providers (CP) are brokers that send *CxtObj*

associated with some QoCP to the Context Information Service (CIS) belonging to the same domain, i.e., its primary CIS. Each CP is registered at only one primary CIS. CIS is composed of various modules in charge of context management functions: Context Collector (CC), Context Reasoner (CR), Context Obfuscator (CO), QoC Evaluator (QoCE), and Context View Provider (CVP). CIS can still communicate with other CIS of different domains with which maintain trust relationships.

The separation of $CxtMF$ into two main entities (i.e., CP and CIS) is only functional. It means that $CxtMF$ can implement these entities running together on a single processing unit (e.g., an application server running the CP and CIS) or on various distributed processing units (e.g., CP running on smartphones and CIS on an application server). In the following, we give some definitions used throughout this chapter:

- *Context Information (CI):* CI represents a set of context information supported by the system. Each element of CI ($ci \in CI$) has a domain of possible values, denoted as ci_{dv}. For instance, ci could be location, time, date, etc;
- *Entity (E):* E represents a set of real world entity that can be observed by the pervasive environment. By observed we mean the environment ability of collecting CI about such entities. For example, $e \in E$ could be a user, a room, etc;
- *Sensor (S):* S represents a set of sensors that can be used to gather information about the observed entities E in the environment. A sensor can be classified as *physical* or *logical*;
- *Context Object (CxtObj):* it is a set of objects that represents a set of context information CI and its value gathered by sensors S about entities E;
- *Quality of Context (QoC):* QoC represents a set of information that describes the quality of a context information ci. QoC can be classified yet as QoC parameter (QoCP) or QoC indicator (QoCI).

Therefore, a $ci \in CI$ associated with a entity $e \in E$ is sensed from the environment by using one sensor $s \in S$, which is represented in the framework by a $co \in CxtObj$. Moreover, a $CxtObj$ can still be associated with some QoC information gathered and generated throughout the management process.

Before describing in detail $CxtMF$, we present the Context and QoC models used as basis to represent semantically context and QoC information in the $CxtMF$, respectively.

7.2.1 Modeling Context and Quality of Context

Various modeling approaches can be used to represent context and QoC information in pervasive systems, as discussed in Chapter 4. In [Strang 2004], Strang *et*

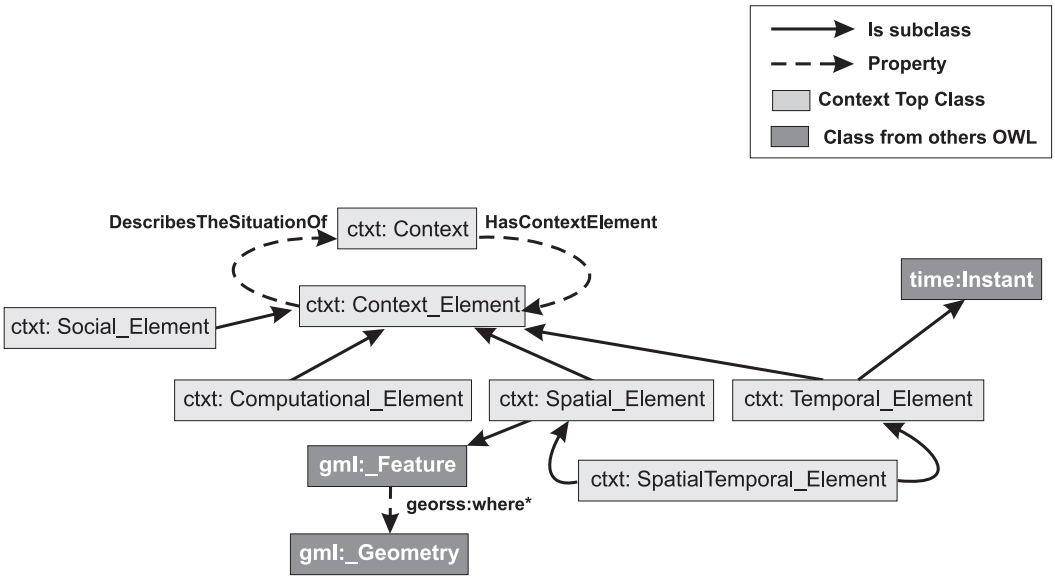


Figure 7.3: Context Top Ontology [Viana 2008].

al. present a survey of existing context modeling solutions and provides a taxonomy to classify them. They identified the following categories of technologies to modeling context: key-value models, markup scheme models, graphical models, object-oriented models, logic-based models, and ontology-based models.

Our experience with Semantic Web and Web 2.0/3.0 technologies shows that using ontologies for modeling context information is well suited for PCE [Viana 2008]. In fact, ontologies are often used in order to achieve a shared semantic understanding of concepts and the relationships that hold among them. Besides that, ontologies allow semantic enrichment of context information through inference and/or derivation processes.

Therefore, we have defined two ontologies for modeling Context and QoC information in order to facilitate the context and QoC representation, sharing, and semantic interoperability in the *CxtFM*. We used the OWL Web Ontology Language[2] to represent the proposed ontologies: *Context* and *QoC* ontologies.

7.2.1.1 Context Modelling

We proposed in [Viana 2008] a *Context Top* Ontology that classifies context information according to five different dimensions (see Figure 7.3) : spatial (e.g., location), temporal (e.g., date, instant, interval), spatio-temporal (e.g., weather conditions), social (e.g., nearby persons and friends), and computational (e.g., Bluetooth address of nearby devices). These dimensions are defined in the following:

[2] http://www.w3.org/TR/owl-ref/

- *Spatial dimension:* this contextual dimension characterizes the situation of observed entities from spatial aspects. For instance, indoor and outdoor location, GPS coordinates, address;

- *Temporal dimension:* a context information belongs to this dimension if it characterizes the situation from time aspects. For example, instant, period of day, month, year, day, etc;

- *Spatio-temporal dimension:* this dimension characterizes the situation of observed entities from both spatial and temporal aspects. Each piece of context information is associated with a particular location at a particular time. For instance, weather conditions, temperature, noise, luminosity, etc;

- *Social dimension:* this dimension characterizes the situation from social relationships. For example, a context management framework could identify the persons in the environment, user's friends around when she/he uses a context-aware application, etc;

- *Computational dimension:* this dimension characterizes the situation from computational characteristics. We still classify this information in two different types: *invariable* and *variable*. An invariable context information is constant over the useful life of the sensor. For example, the capabilities of the user's mobile device is an invariable context information. A variable information is just the opposite, where it may change during the useful life of the sensor. For example, it can be the pervasive devices around the user (e.g., other mobile devices and printers), the consumption of memory and processing of a mobile device, etc;

Observed entities can be classified as: *user*, *environment*, and *resource*. We are reusing the *Context top Ontology* that we have defined in [Viana 2008] as a basis to define news ontologies to represent context of user (CxtUser ontology), context of resource (CxtRes ontology), context of environment (CxtEnv ontology), and access context (AccessCxt Ontology, which is used to represent the context of access entities at request time).

Moreover, we are reusing GeoRSS[3] concepts to describe GPS coordinates, OWL-Time[4] ontology in order to express temporal information, and the RDF FOAF[5] ontology for describing social context dimensions.

Figure 7.4 illustrates the *CxtUser* model, i.e., the context of user. The main context concepts related to users are the following: Location (Indoor and Outdoor), FOAF profile, Activity (Personal and Professional), and Time (Instant, period of day). User's identity can be semantically described by a *ID*, *fname* (first name),

[3] http://www.georss.org/
[4] http://www.w3.org/TR/owl-time
[5] http://xmlns.com/foaf/spec/

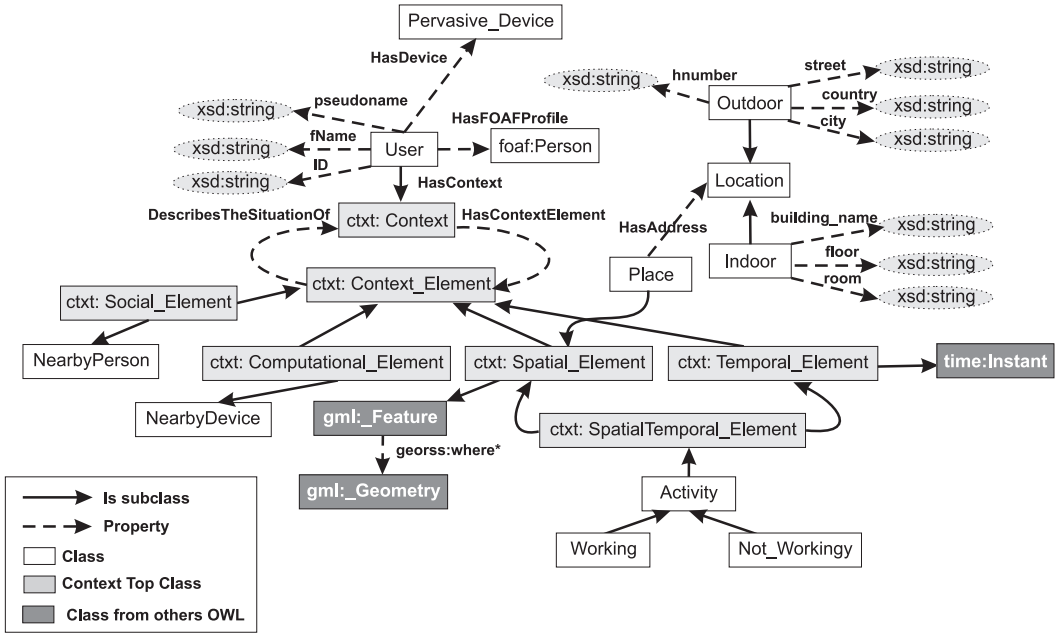


Figure 7.4: CxtUser Ontology: Context of users.

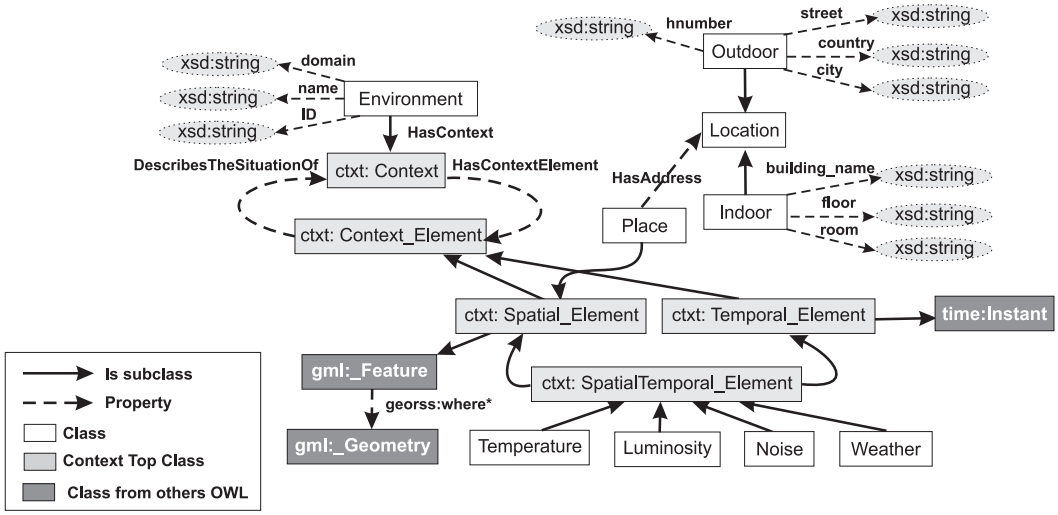


Figure 7.5: CxtEnv Ontology: Context of environment.

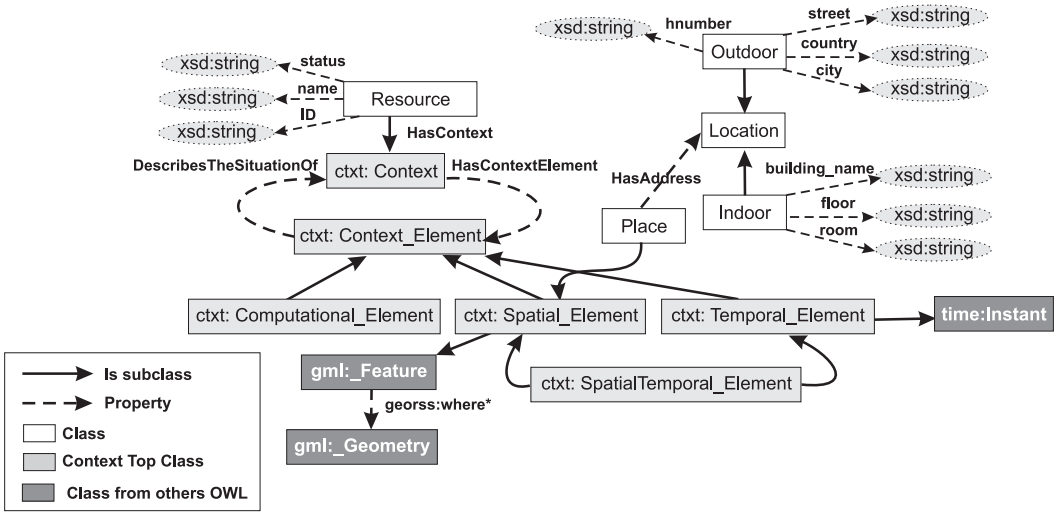


Figure 7.6: CxtRes Ontology: Context of resource.

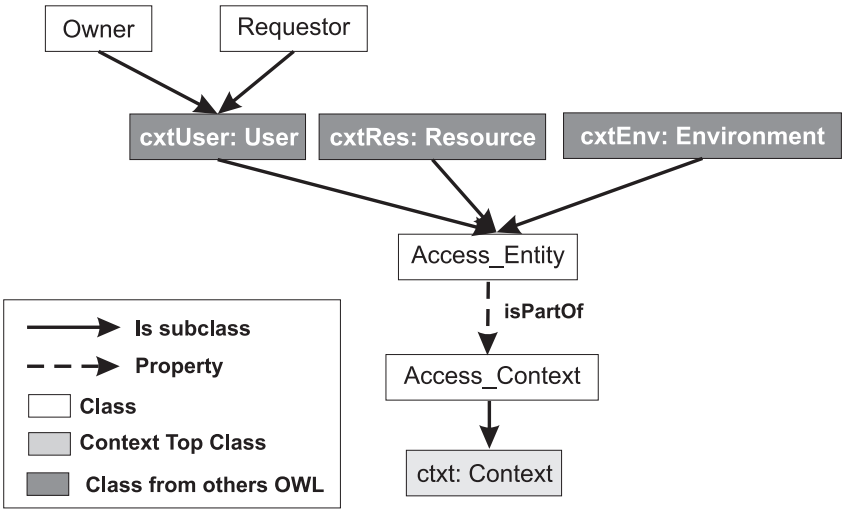


Figure 7.7: Access Context Ontology: Observed entities and the environment.

lname (last name), *pseudonym*, and *role_group*, which are datatype properties of the *Identity* concept.

We are using IETF RFC 4119[6] as basis to represent semantically indoor and outdoor locations. Indoor location can be described using any of the following formats: *building_name* (LMK); *building_name* and *floor* (LMK, FLR); *building_name*, *floor*, and *room* (LMK, FLR, LOC). Outdoor location can be stated using four standard notations: *country* (country); *country*, and *city* (country, A3); *country*, *city*, and *street* (country, A3, A6-STS); *country*, *city*, *street*, and *house_number with suffix* (country, A3, A6-STS, HNO-HNS).

Figure 7.5 illustrates the context of environmental entities. The lowest level of granularity to define a pervasive environment is a room for a indoor situation and a GPS coordinate for a outdoor situation. A environment belongs to a domain (data property *domain*) that has various distributed sensors, which are represented as *SpatialTemporal_Element* (e.g., temperature, noise, luminosity). Like users, the environment is also associated with a location and a time. Figure 7.6 illustrates the context of resource entities, such as protected files, printers, services, etc. The status of dynamic resources (e.g., printers, services) is represented by the concept Status.

Figure 7.7 illustrates the access context concept described in Chapter 6. This ontology represents the relevant context information for making access control decisions about the observed entities (users and resource) and the environment around them. It is defined using as basis the CxtUser, CxtEnv, CxtRes ontologies. Each context-aware application/service registered in the *CxtFM* is able to define their own context model.

Relationships between the context of observed entities and QoC information are defined by two object properties defined in the QoC ontology: *hasQoCP* and *hasQoCI*. Before presenting the QoC Ontology, we need to introduce some QoC concepts that were used to guide our definition.

7.2.1.2 QoC definitions

Modeling QoC is not always straightforward and easy due to the subjective nature of the term quality. Unlike existing works [Buchholz 2003, Razzaque 2005, Preuveneers 2006, Kim 2006b, Sheikh 2008, Sheikh 2007] that identify only QoC dimensions for describing the quality of context, we are classifying QoC in two different types: *QoC indicators* (QoCI) and *QoC parameters* (QoCP). We define **QoC indicators (QoCI)** as *any well-defined quality aspect that can be evaluated by the system and used for describing the quality of context information.* For instance, we can evaluate the *QoCI precision* for describing the quality of location information used by a location-based service (LBS).

[6] http://www.ietf.org/rfc/rfc4119.txt?number=4119

We define **QoC parameters (QoCP)** as *any information sensed from the environment that can be used for measuring QoC indicators.* For instance, *captureTime*, *currentTime*, and *lifeTime* are QoCP used for measuring the *QoCI up-to-dateness*. In fact, QoCI represent high-level interpretations of QoCP according to a well-limited aspect (e.g., precision, resolution). Therefore, QoCI values are more likely to be used by services and applications than *QoCP*.

We classify *QoCI* according to the moment that they should be measured: *real-time* QoCI and *transformation-time* QoCI. A *real-time* QoCI should be measured at the very moment when a *CxtObj* will be used/verified by a context consumer. It means that its value will not be valid if this QoCI has been measured for a non immediate use. For instance, *QoCI up-to-dateness* is a *real-time* QoCI. A *transformation-time* QoCI should be evaluated each time that its associated *CxtObj* is transformed/generated by a derivation/inference process in order to get a new high-level context information. For example, QoCI precision is a *transformation-time* QoCI. These concepts are important to guide us for defining the QoC measuring methods proposed in this work. Moreover, they allow us to identify at which time and layer(s) of the context management framework each *QoCI* must be evaluated.

Buchholz *et al.* [Buchholz 2003] claim that *precision*, *probability of correctness*, *trust-worthiness*, *resolution*, and *up-to-dateness* are the most important QoCI for PCE. Kim *et al.* [Kim 2006b] have proposed a different set of QoCI: *accuracy*, *completeness*, *representation consistency*, and *access security*. Later on, Sheikh *et al.* [Sheikh 2007] have considered the following QoCI: *precision*, *freshness*, *temporal resolution*, *spatial resolution*, and *probability of correctness*. From our point of view, the relevant set of QoCI is directly dependent from the context consumer (i.e., application and service using that information) and/or the situation. For example, *QoCI up-to-dateness* is a very important QoCI for real-time context-aware applications, such as health care application. However, up-to-dateness is not so relevant for context-based annotation systems such as multimedia management applications [Viana 2008].

Therefore, we have defined the *QoC* model using as basis the set of QoCI identified in the existing work [Buchholz 2003, Kim 2006b, Sheikh 2007, Sheikh 2008, Manzoor 2008]. This set was extended with the following new QoC concept: *sensitiveness*. Moreover, we have redefined the following QoCI concepts: *precision*, *completeness*, *resolution*, *access-security*, and *up-to-dateness* (see Section 7.2.4). Thus, the set of QoCI described in the *QoC* ontology is composed by the following elements: precision, correctness, consistency, completeness, resolution, accuracy, trustworthiness, access-security, sensitiveness, significance, and up-to-dateness. *QoC* model is extensible, allowing the addition of new QoC concepts (QoCI and its associated QoCP). Unlike the model proposed in [Razzaque 2005], *QoCOnt* allows us also to define similarity relationships between *QoC* concepts and to represent *QoCP*.

In order to measure a *QoCI*, it is used one or more associated *QoCP*. To measure the QoCI *precision*, *completeness*, *resolution*, *access-security*, *up-to-dateness*, and

Table 7.1: Relationships between QoCI and QoCP

QoC Indicator	QoC Parameter
Up-to-dateness	captureTime, currentTime, lifeTime
Sensitiveness	numberOfDisclosureLevel, currentDisclosureLevel
Access Security	CurrentSecurityLevel, NumberOfSecurityLevel
Completeness	NumberOfAnsweredRequest, NumberOfRequest
Precision	NumberOfPrecisionLevel, CurrentPrecisionLevel, ProcessAccuracy
Resolution	NumberOfGranularityLevel, CurrentGranularityLevel, EntityLocation

sensitiviness we defined the set of *QoCP* described in Table 7.1. For instance, *captureTime*, *currentTime*, and *lifeTime* are QoCP used to measure the *QoCI up-to-dateness*.

QoCP can be captured at two different moments: at context sensing time, or at evaluating time of the associated QoCI. In the next sections, we present the *CxtOnt* and *QoCOnt* models in detail.

7.2.1.3 QoC Ontology

Figure 7.8 illustrates the QoC ontology defined to represent QoC associated with context concepts of Context Top Ontology. QoC model is constructed around two main classes: `QoCP` and `QoCI`. The `QoCP` class has fifteen pairwise disjoint sub-classes which define the set of QoCP that we are taking into account.

ElementaryElement class represents the raw context data. The link between *ElementaryElement* concept (subclass of the *Context_ Element* class defined in the CxtOnt) and the *QoC* concepts is established using the object property `hasQoCP`. New QoCP and QoCI can be defined if needed, as new specializations of the `QoCP` and `QoCI` class, respectively.

The `QoCI` class models the QoCI. For associating context elements with QoCI we use the `hasQoCI` object property. A context element can be linked to every defined `QoCP` and to every defined `QoCI`. One can also use QoCI defined in other ontologies by specifying alignments or correspondences with our *QoCOnt* ontology.

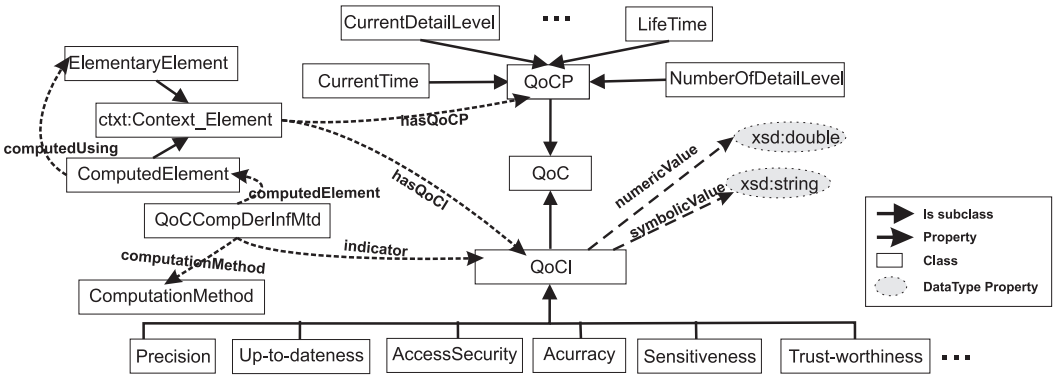


Figure 7.8: QoC Ontology.

An inferred or derived context element, named CE, (i.e., instance of the `ComputedElement` class) is calculated using one or several raw context information represented by *ElementaryElement* class, $E_1, \ldots, E_k$, designated by the `computedUsing` object property. In order to calculate the $QoCI$ attached to the computed context element CE, computation methods are applied on the $QoCI$ defined for the elementary context elements, $E_1, \ldots, E_k$.

We use reification in order to specify the ternary relation (`QoCCompDerInfMtd`) between the computed context element (`ComputedElement`), the `QoCI` to be computed, and the computation method (`ComputationMethod`) to be used.

7.2.2 Context Providers (CP)

Context Providers (CP) are the $CxtMF$ entities in charge of gathering context information measured from the environment. CP are brokers on the environment or on the mobile devices that are able of capturing and sending *CxtObj* to the *Context Information Service (CIS)* (see 7.2). A CP can be configured to run in one of these two operating modes: *push* and *pull*. In the push operating mode, CP send automatically the gathered information to the context information service (CIS) as alerts. However, in the pull mode the CIS should request the CP every time that they need any contextual information managed by that CP.

A CP can manage one or more sensors, which can be of type cxtsensors (this sensor is in charge of gathering the raw context data) and QoCsensors (this sensor is in charge of gathering QoC parameters when the QoC is activated in the framework). Therefore, a CP can be in charge of gathering one or more kind of context information. To support this functionality, CP keep a dynamic list of registered sensors S, controlling the synchronous (push operating mode) and asynchronous (pull operating mode) notifications from them. Moreover, CP keeps a dynamic queue of receiving information from each registered sensor.

A CP can have one or more QoC evaluating components (QoCEC) registered in

charge of gathering and evaluating QoC information associated with $CxtObj$. CP deploy dynamically the QoCEC in order to evaluate the quality of raw context data. Each $QoCEC$ is in charge of evaluating a well-defined QoC dimension (QoCEC is described in details in Section 7.2.4). By using $CxtObj$ enriched with QoC information, a CP is able to select sensors based on QoC thresholds defined by the context management administrator. CP support four aggregation methods for evaluating QoC thresholds, as described in the following:

- *Default:* it evaluates individually all QoC thresholds, verifying if each current evaluated QoC indicator associated with the $CxtObj$ reaches its correspondent QoC threshold. Formally, let qi_j be a $QoCI$ and $QoCI_{t,j}$ be its correspondent QoC threshold, $j = 1, 2, \ldots, n$, and n is the number of elements in $QoCI$. For each $qi_j, if\ qi_j \geq QoCI_{t,j}\ then\ qi_j$ reaches $QoCI_{t,j}$. If all qi_j reaches its correspondent $QoCI_{t,j}$, then the raw context information represented by the $CxtObj$ meets the predefined quality requirements and will be forwarded to the CIS. In another case, the information will be discarded by the CP;

- *Pessimistic:* this approach takes the highest QoC threshold value as the global QoC threshold. Formally, let $QoCI_t$ be a QoC threshold, $QoCIT$ be the set of QoC thresholds, and $hQoC_t$ be the highest QoC threshold of $QoCIT$. Formally, $\forall QoCI_t \in QoCIT, QoCI_t \leq hQoC_t$. $hQoC_t$ will be defined as the Global QoC threshold $QoCI_g$. In this case, each current evaluated QoC indicator associated with the $CxtObj$ should reaches that global QoC threshold. Formally, let qi_j be a $QoCI$, $j = 1, 2, \ldots, n$, and n is the number of elements in $QoCI$. For each $qi_j, if\ qi_j \geq QoCI_g\ then\ qi_j$ reaches $QoCI_g$. If all qi_j reaches the $QoCI_g$, then $CxtObj$ will be forwarded to the CIS. In another case, the information will be discarded by the CP;

- *Optimistic:* this approach is similar to pessimistic but it takes the lowest QoC threshold value as the global QoC threshold. Formally, let $lQoC_t$ be the lowest QoC threshold of $QoCIT$. $\forall QoCI_t \in QoCIT, QoCI_t \geq lQoC_t$;

  Average and weighed average: these methods calculate the average value of QoC thresholds and use it as Global QoC thresholds. The weighed average takes into account the weight of each QoCI indicator when calculating the average.

CP supports yet three sensing modes described below:

- *Default:* if a CP is configured to operate in this sensing mode, only raw context information will be gathered from the sensors controlled by it. In this case, the $CxtFM$ will run like a QoC-unaware context management system;

- *Supporting QoC:* by using this sensing mode, raw context information will be enriched with some QoC parameters. QoC parameters will be used for

measuring QoC indicators associated with *CxtObj* by the upper layers of context management;

- *Measuring QoC:* if a *CP* is configured to operate in this sensing model, raw context information *CxtObj* will be enriched with QoC Parameters and QoC Indicators. In this case, *CP* should run the QoCE components in order to measure the QoCI values.

Each *CP*sensing mode has advantages and drawbacks. Default sensing mode requires less processing time and memory. However, if it is used the framework and applications will not be able to take advantage of supporting QoC in the various steps of management and use of contextual information, respectively.

The second sensing mode offers means of evaluating QoC in the upper layers of the framework. However, *CP* will not be able to select the sensor with the highest QoC indicators from a set of redundant sensors, i.e., *CP* will be QoC-unaware. In this case, the quality verification of redundant context information should be carried out by the upper layers of *CxtFM*. Finally, *CP* sent the collected information to the *Context Collector (CC)* entity, i.e., a set of *CxtObj* that may be associated with some *QoC* information.

In the *CxtMF*, context information is semantically represented by using CxtUser, CxtEnv, CxtRes, and *QoC* Ontologies. *CxtMF* supports also Java Beans and XML files to represent *CxtObj*. All configuration parameters described previously (e.g., QoC threshold values, the aggregation method type) must be described in a configuration file. We have described it in a XML file named *CP_ config.xml*.

The gathered context information by *CP* is sent to the Context Information Service (CIS), more specifically to the component *Context Collector (CC)*. The next section presents the *CIS* in details.

7.2.3 Context Information Service (CIS)

Context Information Service (CIS) is the main component of our context management framework (see Figure 7.2). It is composed by the following sub-services: *Context Collector (CC)*, *Context Reasoner (CR)*, *Context Obfuscator (CO)*, *QoC Evaluator (QoCE)*, *Context View Provider (CVP)*. *CxtMF* is configurable, allowing activate/deactivate the sub-services CR, CO, and QoCE when the administrator deems it necessary. Deactivating these services do not affect the core functionality of the context management framework: to capture and provide information to context consumers. In this case, CVP communicates directly with the CC.

- *Context collector (CC)* is in charge of receiving context information from (CP) and environment sensors;

- *Context Reasoner (CR)* is in charge of inferencing and deriving operations on raw sensed data;

- *Context Obfuscator (CO)* applies privacy rules on semantic high-level context information;

- *QoC Evaluator(QoCE)* measures the QoC associated with each context information;

- *Context View Provider (CVP)* is in charge of making interface with context consumers, providing QoC-enriched context information.

CIS keeps a list of registered *Context Provider (CP)*, a list of registered environment sensors (S), and a list of registered context consumers. Registered *context consumers* are allowed to configure QoC thresholds and QoC aggregation methods in order to filter context information according QoC requirements. This set of XML configuration files, named *Consumer's QoC policies*, are enforced only when the QoC support in the framework is activated. In the following, we present in detail each sub-service that composes the *CxtMF*.

7.2.3.1 Context Collector (CC)

Context Collector (CC) collects and aggregates context and QoC information sent by *CP*. Moreover, *CC* can collect context information sent directly by sensors distributed on the environment (see Figure 7.2). In this case, *CC* incorporates some features of *CP*. The context and QoC information collected is used to construct the *global context* of the observed entities. *Global context* consists of all context information that the system is able to collect about an observed entity, such as a user or a room (environment).

CC communicates directly with *QoC Evaluator (QoCE)* service in order to evaluate the QoC dimensions of raw context information. This process is executed only if the *CP* and the *CxtMF* are configured to support QoC management functionalities.

Global context of each observed entity is stored in the *Global Context Repository (GCR)* (i.e., a SGDB). Global context represents all context information that the framework is able to gather for charactering a observed entity and the environment. At this moment, an instance of the correspondent context model of the observed entity is generated with the most recent *global context* (i.e., *CxtUser* or *CxtEnv* or *CxtRes*). This OWL document will be used by the *Context Reasoner (CR)* and *Context Obfuscator (CO)* services for inferring/deriving high-level context information from raw sensed data.

7.2.3.2 Context Reasoner (CR)

Context Reasoner (CR) runs inference and derivation process on raw sensed data in order to obtain semantic high-level context information. This process is carried out on OWL documents that describe the Global context of observed entities. Our implementation of *CxtMF* uses Pellet[7] for evaluating SWRL[8] rules predefined by the context management administrator in order to infer new higher level context information from raw context data.

For instance, a SWRL rule can be defined for inferring the name of nearby user's friends from Bluetooth address of nearby devices (computational context) and the FOAF[9] user's profile. This new information (nearby user's friends) will compose the social context dimension of the Global context of users (i.e., an instance of CxtUser).

To infer new high-level context information from raw sensed data, *CR* supports the dynamic deployment of *Context Reasoner Components (CRC)*. Each *CRC* is in charge of deriving/generating a new context information from the existing raw sensed data. For example, a *CRC* can be implemented to use the georeverse Web Service (Geonames[10]) in order to set the address from GPS coordinates.

After executing the set of inference rules and the set of *CRC* on raw context data described by the global context, the *CR* must evaluate the *QoC* associated with each new high-level context information. This operation is executed only if the QoC support is activated in the *CxtMF* (*CIS_ config.xml*).

7.2.3.3 Context Obfuscator (CO)

Context Obfuscator (CO) enforces privacy policies on context information by running obfuscation and anonymization operations based on ontologies. Privacy policies are divided in two set of rules: personal privacy policies and global privacy policies. Personal privacy policies are defined by the owner of context information, i.e., the user.

However, the global privacy policies are defined by the *CxtMF* and are applied on the context information of all observed entities. There exists two operating modes that define the enforcing priority of privacy policies: *mandatory* and *discretionary* mode. In the mandatory mode, global privacy policies take precedence over personal privacy policies in the case of having two or more conflicting policies. However, by operating in the discretionary mode the personal privacy policies take precedence over global privacy policies.

We are using an approach similar to the solution proposed by Wishart *et al.* [Wishart 2007] for obfuscating context information based on ontologies. In our

[7] http://clarkparsia.com/pellet
[8] http://www.w3.org/Submission/SWRL/
[9] http://www.foaf-project.org/
[10] http://www.geonames.org/

approach, privacy policies are represented by using SWRL rules. Privacy policies are evaluated by the *CO* in order to limit and/or to generalize the disclosure level of context information. As result, only the disclosed context concepts, properties, and datatype properties described on the global context document will be described on the resulting OWL document, named *Context View*.

Privacy policies are stored by the *Privacy Policy Management (PPM)*, which provides a web interface to ease the writing of SWRL rules. As occurs with *CR*, after applying the SWRL privacy rules if the QoC support is activated in the framework, *CR* requires the *QoCEC* for re-evaluating the QoC of modified context concepts described in the Context View. By performing this operation, the consistence of QoC values assigned to the context concepts will be preserved.

7.2.3.4 QoC Evaluator (QoCE)

QoC Evaluator (QoCE) is the main service of QoC evaluating process. It allows to dynamically deploy *QoCEC* in order to evaluate the QoC dimensions supported by the *CxtMF*. There exists a *QoCEC* for each well-defined QoC aspect (e.g., precision, up-to-dateness).

Let us note that in the *CxtMF*, QoC measuring process is performed in two steps: *first step* evaluates QoC on raw sensed data, which is realized by CP/CC as was described in previous sections; In the *second step*, QoC of high-level context information resulting of deriving, inferring, protecting operations executed on raw sensed data is re-evaluated.

We describe in detail the QoCE components proposed for evaluating the quality of context in section 7.2.4.

7.2.3.5 Context View Provider (CVP)

Context View Provider (CVP) answers the context information queries, providing *Context Views* to context consumers. They can operate in two modes: *push* and *pull*. Moreover, they support two types of queries: *full query:* CVP answers the context request by sending the full context view associated with an observed entity; *personalized query:* CVP answers the context request by sending the context view containing only the information requested. Before answering the request, *CVP* calls the $ContextObfuscator(CO)$ in order to evaluate the privacy rules.

If the QoC support is activated in the *CxtMF*, CVP is able to provide QoC-enriched *Context Views* to the context consumers. Context View can be sent to context consumers as JavaBeans objects, XML files or a OWL document. *CVC* keeps a list of registered context consumers, which can be context-based applications/services or other *CIS* in which it maintains a trust relationship.

7.2.4 Measuring Quality of Context

It is necessary to use a uniform representation of QoCI
values in order to provide useful quality information for context consumers. Moreover, the QoCI representation must be understandable for any entity in the pervasive environment. Aiming to meet these requirements, we have assumed that for any tuple of *(CxtObj, QoCI)* supported by the context management framework, there exists one sequence of deterministic steps (*Alg*) whose result is a real number in the interval [0,1], where 0 and 1 represent the minimum and maximum quality degrees of *QoCI* related with the context information *CxtObj*, respectively. This definition is described below:

$$\forall\,(CxtObj, QoCI)\;\exists\;Alg(QoCP_{set}) : x,\; x \in \mathbb{R},\; 0 \leq x \leq 1$$

where $QoCP_{set}$ is the set of QoCP values used for measuring the associated $QoCI$, and x is the $QoCI$ value obtained by the QoC measuring method Alg. QoCI value can be also represented in percentage or by using the following symbolic descriptions, which is more understandable by human beings: *low*, **medium**, and *high*. *Low* corresponds to the values in the interval [0,0.33], *medium* to the interval [0.33, 0.66], and *high* to the interval [0,66, 1]. In our context management framework, these QoC measuring methods are implemented as *QoCE components (QoCEC)*. These QoCEC extend the abstract class *QoCEComponent* bellow:

```
public abstract class QoCEComponent {
    public Double nValue  = 0.0;
    // This method receives the CxtObj that will be evaluated by th QoCE.
    public void measureQoCI(ContextElement cxtObj);
       alg(cxtObj);
    // Evaluating QoCI associated with the context element
    public abstract void alg(ContextElement cxtObj);
}
```

The *nValue* is a variable that represents the numerical value of *QoCI*. The method *measureQoCI(ContextElement cxtObjt)* is used by the other components of the *CxtMF* for requiring QoCI evaluation of the contextual information (*CxtObj*). The method *alg(ContextElement cxtObj)* is abstract, which means that this method must be encoded by the classes that extend the class *QoCEComponent*, i.e., the QoCE component implementations in charge of measuring QoCI.

We present in the next subsections the proposed QoC measuring methods for evaluating the QoCI *sensitiveness*, *access-security*, *completeness*, *resolution*, and *precision*. For the other QoCI described in the QoCOnt, we are using the existing approaches described in Chapter 4.

7.2.4.1 Sensitiveness

We define the QoCI sensitiveness *as the disclosure level of context information at a given time. The disclosure level can be changed by the context owners in order to enforce their privacy requirements.* The sensitiveness is a transformation-time QoCI.

Let us take the location of users (outdoor and indoor) as example to illustrate how the QoCI *sensitiveness* can be changed in accordance with the situation. For example, a user defines a privacy rule disclosing his outdoor location for his boss when he is on vacation in the first disclosure level. Table 7.2 shows the location of user associated with the *disclosure levels* supported by the *CxtOnt* Model.

Table 7.2: Indoor and outdoor locations associated with disclosure levels

Disclosure	**Location**	
level	*Indoor*	*Outdoor*
0	Undisclosed	Undisclosed
1	(LMK)	(C)
2	(LMK, FLR)	(C,A3)
3	(LMK, FLR, LOC)	(C,A3,A6-STS)
4	-	(C,A3,A6-STS,HNO-HNS)
5	-	GPS coordinates

In order to measure the *QoCI sensitiveness*, let QoCP $numberOfDisclosureLevel$ be the maximum disclosure level for the *CxtObj*. This information is obtained from a XML configuration file in our architecture, which describes the disclosure levels for each concept in the CxtOnt Ontology. Let QoCP $CurrentDisclosureLevel$ be the current disclosure level of the *CxtObj*, according the privacy rules of users. The QoCI sensitiveness of *CxtObj*, *S(CxtObj)*, is measured by the equation bellow:

$$S(CxtObj) = \frac{CurrentDisclosureLevel}{numberOfDisclosureLevel} \tag{7.1}$$

where QoCP $numberOfDisclosureLevel \neq 0$. The value 0 means that the context information is undisclosed and the value 1 means that the context information is being provided in the highest disclosure level. In the example described previously, the $S(location.outdoor) = \frac{1}{5}$, then $S(location.outdoor) = 0.2$, which can be interpreted as a *low* level of disclosure. This QoCI informs the context management layers and context consumers about the level of sensitivity of that context information, with regard to the privacy requirements of users. By using that QoCI, context management frameworks and context consumers will be able to apply security mechanisms to protecting the context information accordantly.

We present bellow the code of a CoCE component
(class *QoCECsensitiveness*) that implements this QoCI measuring method:

```
public class QoCECsensitiveness extends QoCEComponent {
public void alg(ContextElement cxtObj){
// Verifing if the used QoCP values are not equal to zero.
if (cxtObj.getQoC().getQoCPcurrentDisclosureLevel() != 0 &&
    cxtObj.getQoC().getQoCPnumberOfDisclosureLevel() != 0) {
    // Measuring sensitiveness
     this.nValue= (double)
          (cxtObj.getQoC().getQoCPcurrentDisclosureLevel() /
           cxtObj.getQoC().getQoCPnumberOfDisclosureLevel());
     cxtObj.getQoC().setQoCISensitiveness(this.nValue);
  } else cxtObj.getQoC().setQoCISensitiveness(0.0);
 }
}
```

QoCECsensitiveness extends the abstract *QoCEComponent* class. The command if verifies if each QoCP used to measure the QoCI is not equal to zero. If this condition evaluates to true, then $nValue$ gets the resulting value of QoCI measuring method, which will be assigned to the QoCI associated with contextual information ($cxtObj.getQoC().setQoCISensitiveness(this.nValue)$). In any other case, it will be assigned zero to the QoCI associated with the evaluated context information ($cxtObj.getQoC().setQoCISensitiveness(0.0)$).

7.2.4.2 Access-Security

We define the QoCI access-security as *the probability with which the context information is delivered in security to the context consumers.* This real-time QoCI is useful to know the probability with which the context information has been maintained in security, from its capture by sensors to its use by context consumers.

We consider that the context management frameworks are able to adapt the security mechanisms used to protect the communication channels between the S and CIS, and between CIS belonging to different domains. For instance, in the $CxtMF$ a CP (running on the environment or on a smartphone) is able to adopt any FRAMESEC security strategy [Filho 2005] in order to protect its communication channels. In this case, the CP verifies constantly its battery life in order to adapt their security mechanism accordingly, i.e., by using another instance of FRAMESEC that requires less consumption of resources and, consequently, offers a lower security level.

It is based on a configuration file describing all security mechanisms supported by the $CxtMF$ for protecting the communication channels, sorted by the security level provided for each solution. For instance, the AES[11] symmetric algorithm provides a higher level of security than the 3DES, which provides a greater level of security than the DES. Thus, security mechanisms constructed using these algorithms for

[11] http://www.csrc.nist.gov/

providing confidentiality of context information will follow the same classification of security level. Let QoCP *CurrentSecurityLevel* be the security level of the current mechanisms used and QoCP *NumberOfSecurityLevel* be the maximum security level described in that configuration file. The QoCI access-security related to *CxtObj*, *AS(CxtObj)*, is measured by the equation bellow:

$$AS(CxtObj) = \frac{CurrentSecurityLevel}{NumberOfSecurityLevel} \tag{7.2}$$

where $NumberOfSecurityLevel \neq 0$. The value 0 means that any security mechanism was used to protect the communication channel, and 1 means that was used the mechanism with the highest security level supported. This QoCI is useful for CIS and context consumers to select CP and CIS that provides context information, respectively. For example, a CIS can be configured to accept context information from *CP* only if the associated *QoCI access-security* reaches the minimum of required security.

7.2.4.3 Completeness

We define the QoCI completeness as *the degree of disponibility with which the context information is provided to the context consumers.* In [Kim 2006b], completeness has been computed as the ratio of the number of context information available to the total number of context gatherings. Manzoor *et al.* [Manzoor 2008] have enhanced this concept by using weights for different context attributes, once they do not have the same significance for the context management framework or the context consumer. Then, the completeness of a context object is computed as the ratio between the sum of the weights of available attributes of a context object, and the sum of the weights of all the attributes of that context object.

In our opinion, by evaluating the completeness as the ratio of context information available and to the total number of sensor readings [Kim 2006b] is a costly and inefficient approach because, at each new sensor reading, it is necessary to re-evaluate that QoCI even if the information is not used. Furthermore, the measuring methods proposed in [Kim 2006b] and [Manzoor 2008] do not indicate if the context information is available and current. For instance, the outdoor location of a user could have a high level of completeness but when a context consumer request this information it is not available (e.g., the GPS sensor cannot locate the satellite signals at this moment) or yet it is available but not current (i.e., the *QoCI up-to-dateness* is equal to 0). Therefore, we propose a measuring method for the completeness that describes how the context information is complete, available, and up-to-date.

Let $CO(CxtObj)$ and $U(CxtObj)$ be the values of completeness and up-to-dateness related with the context object, respectively, which are evaluated using the methods proposed by Manzoor *et al.*[Manzoor 2008]. Moreover, let QoCP $NumberOfAnsweredRequest$ be the number of requests answered with a valid

context information (i.e., $U(CxtObj) \neq 0$ and $CxtObj \neq null$), and QoCP $NumberOfRequest$ be the total number of requests performed on $CxtObj$, both obtained from log files. This real-time QoCI associated with the $CxtObj$, $C(CxtObj)$, is measured by the equations bellow:

$$C(CxtObj) = \begin{cases} CO(CxtObj) \times \frac{NumberOfAnsweredRequest+1}{NumberOfRequest+1} \\ : if\ U(CxtObj)\ \neq\ 0\ and CxtObj\ \neq\ null \\ \\ CO(CxtObj) \times \frac{NumberOfAnsweredRequest}{NumberOfRequest+1} \\ : otherwise \end{cases} \tag{7.3}$$

where $NumberOfRequest \geq 0$. The value 0 means that all requests were answered with a context information out of date and/or the context information was unavailable, and 1 means that all requests were answered with a current context information.

7.2.4.4 Precision

We define the QoCI precision as *the level of details in which the context information is describing an entity of the real world.* For example, the identity of users described in function of their name has a higher precision level than their pseudonym or their role_group. For a numeric context information, the value described with three significant figures (e.g., 32.2^0 celsius) is more precise than with two significant figures (i.e., 32^0 celsius). This QoCI is a *transformation-time QoCI*. To measure the QoCI precision, let QoCP $NumberOfPrecisionLevel$ be the maximum level of precision for the $CxtObj$ obtained from a configuration file, QoCP $ProcessAccuracy$ be the accuracy of the process executed in order to obtain the $CxtObj$ (i.e., sensing, inferring, or deriving operation), and QoCP $CurrentPrecisionLevel$ be the current precision level of that $CxtObj$. The precision of $CxtObj$ is measured by the equation below:

$$P(CxtObj) = \frac{CurrentPrecisionLevel}{NumberOfPrecisionLevel} \times ProcessAccuracy \tag{7.4}$$

where $NumberOfPrecisionLevel > 0$. The value 0 means that the accuracy used to obtain that information is 0 ($ProcessAccuracy = 0$) or the precision of this information has not yet been measured. The value 1 means that this information is described in the higher precision level and the process used to obtain it has accuracy equal to 1.

7.2.4.5 Resolution

We define the QoCI resolution as *the spatial granularity with which the context information is being described or sensed from the environment.* For instance, the temperature of a building can be described in the following spatial granularity levels: building, floor, and room. Then the temperature described on room level has higher level of resolution that the temperature described on building level. The measurement of this transformation-time QoCI is highly dependent on how the physical area is expressed in the context management framework.

Let QoCP $NumberOfGranularityLevel$ be the maximum level of spatial granularity associated with the $CxtObj$ obtained from a configuration file. Let QoCP $CurrentGranularityLevel$ be the current spatial granularity level used to describe the location of the observed entity (i.e., $EntityLocation$). The resolution of $CxtObj$ is measured by the equation bellow:

$$R(CxtObj) = \frac{CurrentGranularityLevel}{NumberOfGranularityLevel} \tag{7.5}$$

where the $NumberOfGranularityLevel \neq 0$. The value 0 means that the context information is sensed/described at the lowest resolution level and 1 means at the highest resolution level supported by the context management framework.

7.2.4.6 Measuring QoC of inferred and derived context information

In order to measure QoC of inferred and derived information, we have implemented a special QoCEComponent ($QoCECInfDev$). See below the implementation code of that component:

```
public void algInferred(ContextElement[] cxtObjS,
                ContextElement cxtObj, int method) {
  // If the cxtObj is inferred from only one context concept
  if (cxtObjS.length == 1) {
    cxtObj.setQoC(cxtObjS[1].getQoC());
  }
  else // If the cxtObj is inferred from a set of cxtObj
    if (cxtObjS.length > 1) {
      switch(method){

       case 1: // If the method is pessimistic
         cxtObj.getQoC().setAllQoCI(lowestValues(cxtObjS));
       break;

       case 2: // If the method is optimistic
          cxtObj.getQoC().setAllQoCI(highestValues(cxtObjS));
       break;

       case 3: // If the average method is used
         cxtObj.getQoC().setAllQoCI(averageValues(cxtObjS));
       break;
```

```
      case 4: // If the weighted average method is used
        cxtObj.getQoC().setAllQoCI(weightedAverageValues(cxtObjS));
      break;
     }
   }
}
```

This method receives as parameters a set of CxtObj (*cxtObjS*) that were used for inferring that new context information; the new context concept *cxtObj*; and the method of aggregation to be used: (1) pessimist, (2) optimist, (3) average value, and (4) weighted average value. The first *if* verify if the *cxtObj* has been inferred from only one existing context information (i.e., the set *cxtObjS* has only one element). In this case, we consider that the inferred information has the same set of QoCI values as the context information used for inferring it. If *cxtObj* was inferred using two or more context concepts, we must use a QoC aggregation method to calculate the resulting set of QoCI values. We have implemented five functions that help us to measure the QoC of inferred/derived processes: *lowestValues*, *highestValues*, *averageValues*, *weightedAverageValues*, and *rawValues*. These functions are in charge of identifying the set of resulting QoCI values from the *cxtObjS* that will be assigned to the inferred context information *cxtObj*.

For example, the *lowestValues* function returns an array of lowest QoCI values from the *cxtobjS*. These values are assigned to *cxtObj* by the method *setAllQoCI*. The QoC measuring method for derived *cxtObj* is very similar. The difference is that it must take into account the *accuracy of the process* used for deriving that *cxtObj*. Thus, the set of QoCI values associated with the context information used for deriving that new information will be multiplied by the (*ProcessAccuracy*) in order to get a valid QoCI data. See the implemented method below:

```
public void algDerived(ContextElement[] cxtObjS,
      ContextElement cxtObj, Double pAccur, int method) {
 // If the cxtObj is derived from only one context concept
 if (cxtObjS.length == 1) {
    cxtObj.getQoC().setAllQoCI(rawValues(cxtObjS), ProcessAccuracy);
 } else // If the cxtObj is derived from a set of cxtObj
  if (cxtObjS.length > 1) {
  switch(method){

   case 1: // If the method is pessimistic
    cxtObj.getQoC().setAllQoCI(lowestValues(cxtObjS),ProcessAccuracy);
   break;

   case 2: // If the method is optimistic
    cxtObj.getQoC().setAllQoCI(highestValues(cxtObjS),ProcessAccuracy);
   break;

   case 3: // If the average method is used
    cxtObj.getQoC().setAllQoCI(averageValues(cxtObjS),ProcessAccuracy);
   break;

   case 4: // If the weighted average method is used
    cxtObj.getQoC().setAllQoCI(weightedAverageValues(cxtObjS),ProcessAccuracy);
   break;
```

```
  }
 }
}
```

7.3 Implementation and Evaluation

CxtMF is developed following a component-based architecture. Thus, the steps of context management process are placed into separate components, which can be coupled and reconfigured at runtime. Context information and functions inside each component are semantically related, what makes the components *modular* and *cohesive*. We opted for such kind of architecture in order to facilitate the integration of new QoC evaluating components (QoCEC) as well as new sensors and context management layers. The component-based framework is implemented using Java Technologies: J2EE to the server side (i.e., CIS and CS) and J2ME to the client side (i.e., CP).

As this is a prototype, we do not take into account issues related to user authentication, but it can be easily integrated with an existing authentication service, such as Kerberos[12] or a service based on X.509[13] certificates. Therefore, *CxtMF* does not demand on a particular authentication mechanism, it simply requires that a means exists to authenticate users and services within the system.

7.3.1 Evaluation

We deployed the proposed framework into university building in order to provide context-aware services to the users, such as context-based access control and context-aware control of heating and lighting, among others. In this case study, we evaluated the QoC associated with temperature and luminosity information, which was provided by four Sun Spots[14] installed in two different rooms (D322 and D318).

We deployed a *CP* on each Sun Spot for gathering context information. Afterwards, the gathered information was transmitted to a Sun Spot base station that was connected to a server running a *CIS* (Intel Core Duo 2.GHz, 4 GB, Windows Vista 32 bits, MySql 5.0.45).

The sensing was carried out during 24 hours, with intervals of 5 seconds. When QoC is activated in the *CxtMF*, the QoC indicators will be evaluated using the measuring methods proposed in section 7.2.4. The evaluation consisted of two verifications: (i) a study of performance in order to verify the time overhead added by the quality support in the *CxtMF*, and (ii) an analysis of the use of quality information for selecting context providers by the CIS.

[12] http://web.mit.edu/Kerberos/
[13] http://www.ietf.org/rfc/rfc2459.txt
[14] http://www.sunspotworld.com/

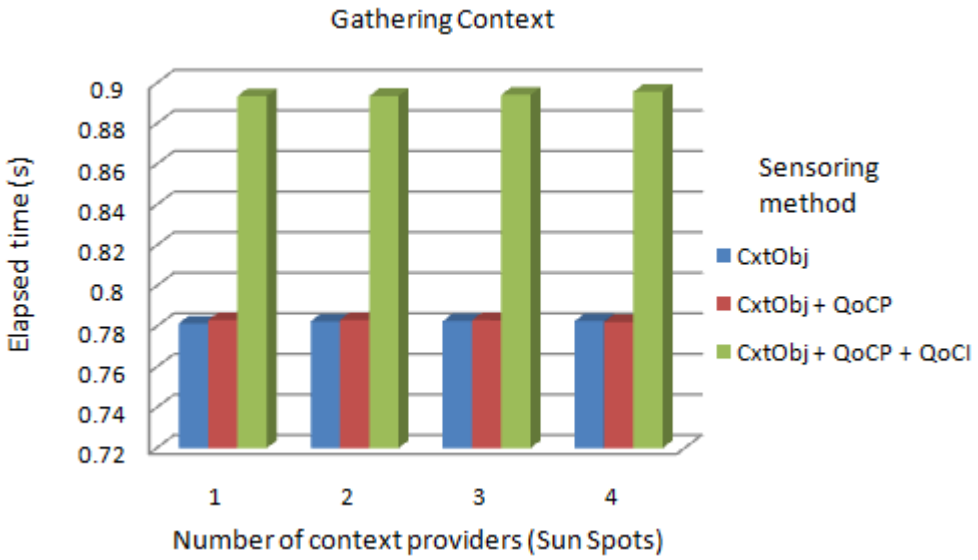


Figure 7.9: The time overhead of gathering context information.

Figure 7.9 shows the results of time overhead for gathering context information. We observed that the time spent for sensing context information (CxtObj) remained almost constant in relation with the growth in the number (1 to 4) of context providers registered in the $CxtMF$.

Moreover, we observed that the execution time of the proposed QoCI measuring methods is approximately 100 milliseconds, which is an acceptable impact on the $CxtMF$ performance.

Figure 7.10 illustrates the time spent for gathering the temperature of two context providers (CP_1 and CP_2) located in the room D322, and the global QoC of each one (QoC_1 and QoC_2) calculated using the average.

We have compared two selection approaches of context providers deployed to evaluate our $CxtMF$: (i) FIFO (First in, first out) and (ii) global QoC-aware approach. By using the FIFO approach, we observed that approximately 40% of cases the CIS selected the information provided by the CP_2 and in the remaining 60% was selected the information from CP_1. This may have been caused by the difference between the distances of each one in relation to the CIS, or yet by synchronization problems. Using the global QoC-aware approach, however, in 100% of cases the CIS has selected the temperature provided by the CP_2.

This occurred because the precision of the temperature provided by the CP_1 (medium, that means that the temperature is described with only one decimal place) was lower than the precision of temperature provided by the CP_2 (high, with two decimal place). In this case, once the CIS identified that CP_2 offers information with higher global quality, it can select the CP_2 and put the CP_1 on its list of context providers that need to be checked before being able to provide context information

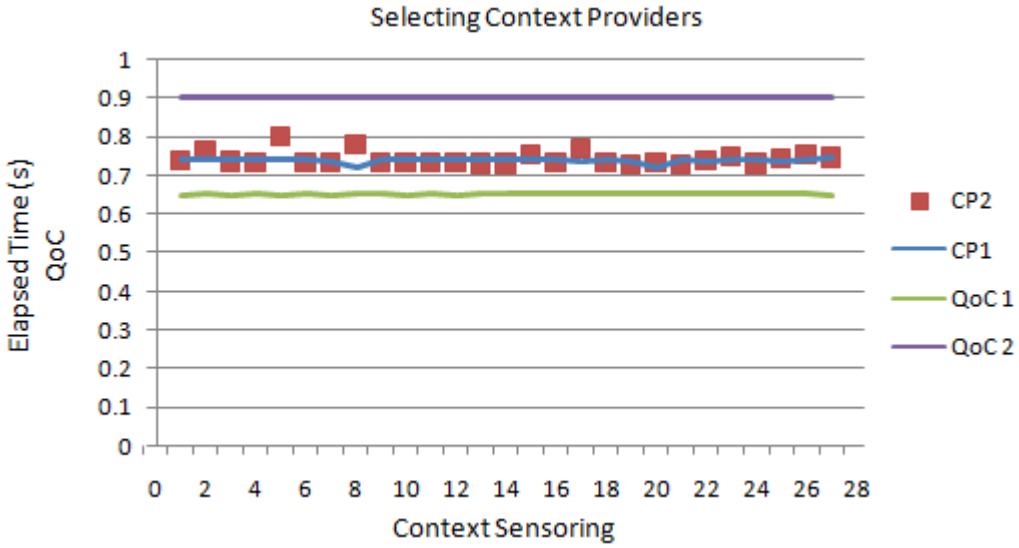


Figure 7.10: QoC-based and FIFO-based selection approaches.

in the future.

7.4 Conclusion

We discussed in this Chapter our semantic approach for modeling and measuring quality of raw, derived, and inferred context information. We showed the influence of quality of contextual information on context management steps. Some new QoC indicators have been defined (sensitiveness) and new QoC measuring methods have been proposed to evaluate the following QoC indicators: sensitiveness, completeness, access-security, precision, and resolution. Moreover, QoC indicators have also been evaluated and can be provided to context-aware applications and services (e.g., a CxtBAC-based solution) along with context information by the proposed framework. The proposed QoC measuring methods and the Context and QoC Ontologies can be re-used and extended for improving existing context management architectures, respectively.

CxtFM was developed in order to provide context information taking into account *QoC* requirements in all steps of context management process. In the following, we summarize the contributions related with this framework to the scientific community:

- The OWL-DL *QoCOnt* to model QoC information, classifying quality information as QoC parameters and QoC indicators. *QoCOnt* can be extended in order to accommodate others QoCI and QoCP;
- New QoC measuring methods to evaluate the quality of context from the

following points of view: privacy, security, resolution, completeness, and precision;

- The QoC evaluating methods that can be used to measure quality of raw, inferred, and derived context information. These measuring methods take into account that context information can be modified after sensing time. Therefore, derived and inferred context information from one or more raw context data can be also evaluated by the proposed QoC measuring methods;
- The context management framework *CxtMF*, which supports QoC in the various layers of context management process, i.e., sensing, inferring, deriving, processing, and providing QoC-enriched context to context-aware consumers.

In our approach, the enrichment of context information with QoC indicators enhances the perception of an application about the context information and enables the system to improve its context-aware decisions.

Chapter 8

CxtBAC Instantiation

Résumé: *Ce chapitre présente une instanciation d'un élément de la famille des modèles de contrôle d'accès proposé pour la protection des ressources multimédias personnelles dans un environnement mobile, en se basant sur un système d'annotations. Nous avons réalisé également une enquête auprès des utilisateurs dans le but d'identifier les informations contextuelles les plus utilisés lors de la définition des politiques de protection des documents multimédia. Cette étude nous a permis définir un ensemble des politiques par défaut, qui pourra être utilisé par les utilisateurs du système développé. Cette instance de modèle a été intégrée avec un outil de capture et partage de contenu multimédia, qui utilise l'architecture de gestion contextuel pour la collecte et l'annotation des documents produit à l'aide des dispositifs mobiles.*

Contents

8.1 Introduction

This chapter describes an access control infrastructure developed to protect multimedia documents. We have used the $S-CxtBAC$ (Social-Aware CxtBAC) model as basis to implement that instance of $CxtBAC$. This new solution is integrated with the $CxtMF$, which is in charge of gathering context information used for annotating and making access control decisions.

The proposed $CxtBAC$ instance, named $CxtANBAC$ (Contextual Annotation-Based Access Control), supports policies more restrictive based not only on the existing social relationships among users but also taking into account the context at creation time of multimedia documents (resource). We believe that users share their personal multimedia documents with other persons following a social network classification. Moreover, it is important to take into account the presence of these persons at creation time. For example, a user could grant read access on their videos made during a trip only with his/her friends who were around him at creation time.

CxtANBAC extends the expressiveness of context-based access policies by combining social relationships with contextual information. Internally to CxtANBAC, context of multimedia documents and the existing social relationships among users are represented by annotations[1].

In fact, with the increasing use of pervasive sensor-rich mobile devices as personal multimedia management tools, multimedia annotation becomes a powerful technology to facilitate the retrieval, organization, and enrichment of multimedia documents [Filho 2010b], such as photos, videos, audios, and micro-blog. The process of multimedia annotation can be performed at creation time by using contextual information filled by users or gathered from sensors embedded on pervasive mobile devices.

Annotation is a common mechanism used by Web 2.0 platforms for attaching information to shared documents. Services like Flickr[2], Picasa[3], ZoneTag[4], and Photomap[5] offer users means of associating manually and semi-automatically tag-based annotations with photos that could be used for improving the retrieval and organization operations of annotated photos. By using information gathered from embedded sensors for annotating multimedia files, it will be possible to characterize the creating situation (context) of these documents, such as location, Bluetooth address of nearby pervasive devices, user activity, and time.

However, existing multimedia systems

[1] In this work annotation refers to all metadata associated with multimedia documents or users for describing social relationships. Annotation can be added manually by users or automatically by software to describe, for instance, the situation at creation time of multimedia documents and their contents.

[2] http://www.flickr.com/

[3] http://picasaweb.google.com

[4] http://zonetag.research.yahoo.com/

[5] https://photomap.liglab.fr/PhotoMap/

[Kahan 2001, Schroeter 2006, Viana 2007a] do not exploit annotation as the central concept for defining access control policies in order to protect multimedia documents. For instance, by using these solutions it is not possible to define access control policies like *"I grant read access on my videos taken in Rome only to my friends"* and *"I grant read access on my photos to only persons from my social network who were around me when I took those photos."*

With this in mind, this chapter presents the use of annotation (contextual and social aspects) as a central concept to define access control policies for protecting multimedia documents. We developed a client-server application, named *PPlog (Pervasive Personal Blog)*, which uses the *CxtANBAC* for protecting the published multimedia documents. PPlog requires social and contextual-based policies for protecting multimedia and daily posts (we describe in detail each type of posts in Section 8.5).

The proposed access control approach is implemented using semantic Web technologies for describing and enforcing contextual annotation-based access control policies, such ontologies[6] and inference/derivation rules[7].

Moreover, we conducted an on-line survey of 200 people in order to identify the most important set of annotations that can be used to define access restrictions, focusing in the policies for protecting personal photos. We believe that the sharing behavior of users with regarding other multimedia content types is very similar. In addition, we have incorporated into *PPlog Application* this set of policies that can be used for protecting the published multimedia and daily posts.

The remainder of this Chapter is organized as follows: Section 8.2 presents an overview on existing annotation technologies. Section 8.3 presents the CxtANBAC proposed for controlling access of multimedia documents. Then, Section 8.4 discusses the results of the carried survey and Section 8.5 the PPlog application. Finally, we conclude this Chapter in Section 8.6.

8.2 Overview on Multimedia Annotation Technologies

Multimedia annotation can be classified according to the subject described by the annotation (e.g., content, context of creation, emotion), its representation (e.g., annotation can be embedded into the media file or described by an attached file), and the attaching process (e.g., manually, semi-automatically, and automatically).

Annotations vary from simple semantic tags (e.g., the services Flicks, ZoneTag, Picasa, and Google Earth[8] uses tag as annotations) to rich and structured annotations such as free text, hyperlinks, wikipedia[9] entries, ranking, language, audiovisual,

[6] http://www.w3.org/TR/owl-features/
[7] http://www.w3.org/Submission/SWRL/
[8] http://earth.google.com/intl/fr/
[9] http://www.wikipedia.org/

etc.

Annotations can be yet attached to the fine-grained segments or to regions of the document. For instance, existing annotation systems like Annotea [Kahan 2001] and Vannotea [Schroeter 2006] enable users to attach personal notes, questions, explanations, etc., to some content that can be categorized according to the media types, such as text, web pages, images, audio, video, and 3D document.

Technologies for representing annotation differ according to the expressiveness provided by its vocabulary to describe the multimedia documents, and its storage method. For instance, DCMES[10](Dublin Core Metadata Element Set), EXIF[11](Exchangeable Image File Format), and ID3[12] (IDentify an MP3) store metadata embedded into multimedia documents.

There exist annotation solutions that use ontologies [Halaschek-Wiener 2005, Naphade 2006, Viana 2007b] or the MPEG7[13] standard, which store the metadata into an external file associated with the multimedia document. These approaches are more powerful with regarding the expressiveness than those that have embedded annotations into multimedia files. In fact, ontologies offer means of describing relationships among annotations and inferring new information.

From our point of view, the most frequent types of annotation associated with multimedia documents include, but not limited to, content, contextual, and emotion annotations. Figure 8.1 illustrates these different sets of annotations, which are defined in the following:

- *Content Annotation:* it describes the content itself of the annotated document. For example, it describes the main subject of the document, such as objects and people that appear in a photo or video, the subtitles of a video file, or yet the transcription of an audio file. This kind of annotation can be attached manually by users or automatically generated by using, for instance, face[14], object[15], and speech[16] recognition algorithms;

- *Contextual Annotation:* it includes any information that can be used to describe the creating situation of multimedia documents. Such information can be automatically gathered from embedded sensors (e.g., GPS coordinates, time, nearby Bluetooth devices) or manually added by users in order to describe the situation at creation time of multimedia documents. For example, date, time, event, location (here we consider the location of the pervasive device as the location of the document at creation time), pervasive device

[10] http://dublincore.org/documents/dces/
[11] http://www.exif.org/
[12] http://www.id3.org/
[13] http://mpeg.chiariglione.org/standards/mpeg-7/mpeg-7.htm
[14] http://www.face-rec.org/algorithms/
[15] http://people.csail.mit.edu/torralba/shortCourseRLOC/
[16] http://cslu.cse.ogi.edu/HLTsurvey/ch1node4.html

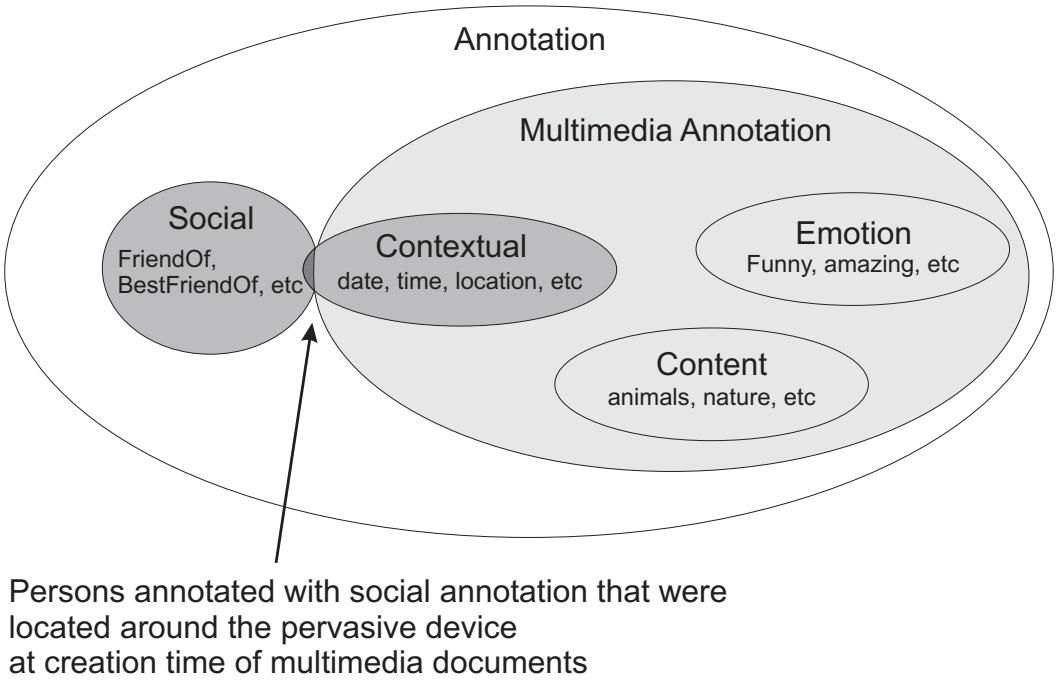


Figure 8.1: Different sets of annotations.

capabilities, nearby objects and persons, and physical addresses of nearby Bluetooth devices can be used to describe the situation at creation time of multimedia documents;

- *Emotion Annotation:* it describes personal feelings or opinion with regarding to the content of multimedia documents. For example, users could annotate photos with terms like funny, beautiful, amazing, etc.

Although the access control solution described in this chapter is generic enough to support any type of annotation, for the sake of simplicity we will restrict our interest to the following types of annotation: contextual annotation and social annotation. *Social Annotation (SA)* (see the definitions in Section 6.9) describes the existing social relationships among people (i.e., users). Moreover, we are interested in the intersection between these two sets of annotation that represents the persons from user's social network which were located around the pervasive device at creation time of multimedia documents (see Figure 8.1).

8.3 CxtANBAC: Contextual Annotation-Based Access Control

As described previously, $CxtANBAC$ is implemented using as basis the $S - CxtBAC$ from the $CxtBAC$ model family. We selected this model taking

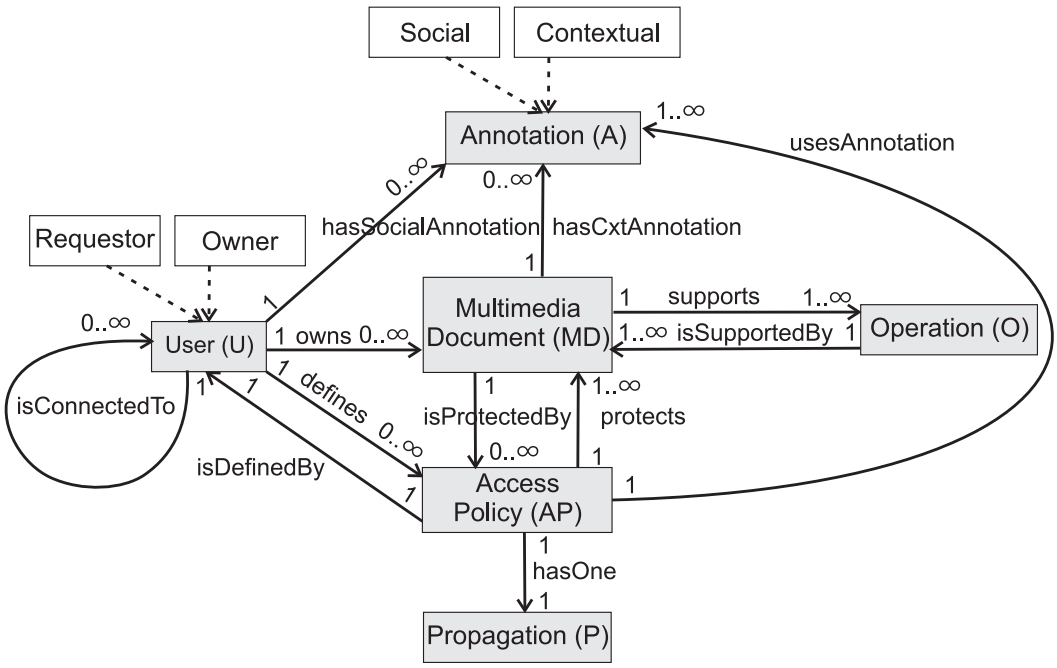


Figure 8.2: CxtANBAC - Contextual Annotation-based access control.

into account the following requirements of pervasive multimedia applications: the support to social-based access control policies; the support to CxtRes-based access control policies, i.e., context of resources; and (Social and CxtRes)-based access control policies.

We are using the term *Annotation* instead of *access context* as central concept for granting permission. In fact, *annotation* is the semantic technology used for implementing the *access context* concept that we defined in Chapter 6.

Moreover, as users should be able to define access control policies for protecting their personal multimedia documents, $CxtANBAC$ is a Discretionary Access Control solution (DAC)[17]. $CxtANBAC$ supports two different types of annotation-based access control policies: *static* and *dynamic* policies (see the definition in Chapter 6).

Figure 8.2 illustrates the CxtANBAC that is composed by six elements and relationships between them: User (U), Annotation (A), Multimedia Document (MD), Operation (O), Propagation (P), and Access Policy (AP).

In the CxtANBAC, a *User* is connected to (*isConnectedTo*) zero or various other users by means of existing social relationships represented in their social network. Each user is able to add manually into his/her FOAF[18] profile these social

[17] Discretionary access control puts control of an object into the hands of the person who creates it.

[18] The Friend of a Friend (FOAF): http://xmlns.com/foaf/spec/

relationships with other users by means of social annotations.

Users can be classified as resource owners, resource requestors, or both simultaneously. This classification is important to identify the correspondent set of context information that will be used for evaluating access control policies. For instance, a resource owner always have access to its multimedia documents, i.e., it will be not necessary to enforce any access control policy associated with its resources in order to grant access permission to him.

A *User* owns (relation *owns*) zero or various *Multimedia_Document*(*MD*). *Multimedia_Document* is a subset of *Resource* defined in the $S-CxtBAC$ model (*Multimedia_Document*(*MD*) $\subseteq$ *Resource*(*R*)). A *MD* can be a post of a blog, a photo, a video, an audio, etc.

Moreover, a *User* is able to define (relation *defines*) zero or various *Access_Policy* to protect his/her set of *MD*. An *Annotation* represents one information associated with a *Multimedia_Document*. A *Multimedia_Document* is annotated (*hasCxtAnnotation*) with zero or various contextual *Annotation*. Moreover, each social relationship among users is annotated (*hasSocialAnnotation*) with zero or various social *Annotation* (e.g., Friend, Parent, bestFriend, etc).

A *Multimedia_Document* is owned by only one *User* that has all possible permission on that *MD*. Each *Operation* represents an action that can be performed on a *Multimedia_Document*. The set of supported operations contains read and write operations. An *Operation* can be associated with one or various *Multimedia_Document*, and it is possible to grant one or various *Operation* to each *Multimedia_Document*.

A *Multimedia_Document* is accessible by users in the form of URI. A *MD* can be protected by zero or various *Access_Policy*. In the case where none *Access_Policy* is defined for protecting a *Multimedia_Document*, then that *MD* will not be accessible by anyone besides its owner.

An *Access_Policy* is defined by one *User* and protects one or various *Multimedia_Document*. An *Access_Policy* makes reference to one or various *Annotation* associated with *Multimedia_Document* and *User* that should be evaluated. Moreover, each *Access_Policy* is associated with exactly one value of *Propagation*.

Propagation is a numeric value indicating the number of hops in the social network that connects the users that might be affected by the policy. For example, if a user defines an *Access_Policy* that grants read access on the *Photo*1 to her/his friends with a *Propagation* value equal to 2, then the read operation will be granted also to the friends of her/his friends if their context meets the constraints defined in the *access policy*.

There exist several non-functional requirements for implementing correctly the *CxtANBAC*. We describe these requirements in the following:

- Only resource owners are able to define *Access_Policies* for protecting their resources, i.e., it is a user-centric access control model;
- We need explicitly define policies in order to grant access permission on our *MD* to other users;
- A *User u* obtains access to a *Multimedia_Document md* if and only if: (i) there exist an *Access_Policy ac* protecting that *md*; (ii) *ac* evaluates true for the set of *Annotation* associated with user and *md*;
- If a *User* obtains access to a *MD* and he/she copies that *MD* to her/his resources, she/he will be also a resource owner of that *MD*. However, the original resource owner will keep the ownership as well;
- Support to context gathering and annotating features of *MD*. Moreover, support to social annotating of users.

In the following, we describe how access policies are semantically represented and enforced in the *CxtANBAC* service.

8.3.1 Defining and Enforcing CxtANBAC Policies

We are using as basis the ECA model (Event-Condition-Action) proposed by Bailey *et al.* [Bailey 2002] for describing and enforcing CxtANBAC policies. This model is divided in three sections: *Event*, *Condition*, and *Action*. Basically, the section event describes the observed event in the system that will trigger the action defined in the ECA rule. However, this action will be performed if only if (iff) the constraint(s) described in the section Condition evaluate(s) true.

In an CxtANBAC policy, the *Event* represents a request access on a protected multimedia document, the section *Condition* represents the access policy associated with that multimedia document, and *Action* the granted permission if the *Condition* evaluates true. *CxtMF* [Filho 2010a] is used for gathering contextual information used for annotating multimedia documents at creation time.

Condition describes a set of valid contextual annotation constraints, and *Action* describes permission that will be granted if *condition* evaluates true for the contextual and social annotation associated with the resource and resource requestor, respectively.

Figure 8.3 illustrates the ECA-based schema defined for enforcing annotation-based access control policies. When the *CxtANBAC* intercepts an access request, it evaluates the contextual annotation associated with the multimedia content and with the resource requestor in order to grant/deny access on the protected content.

There exist several ways to represent and implement ECA rules. We are using a representation based on inference rules. Inference rules are generally based on

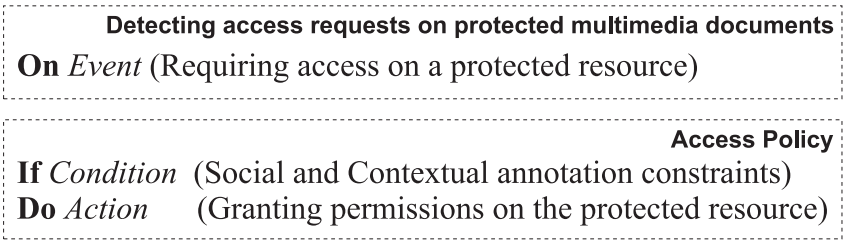


Figure 8.3: ECA schema for defining CxtANBAC policies.

logic programming. For the sake of decidability, ontology languages do not offer the expressiveness we need to describe access control policies. However, inference rules do it well. We opted by using inference rules because it offers means of exploiting directly semantic metadata (i.e., contextual and social annotation) represented by ontologies.

We are using the language SWRL[19] for describing the annotation-based access rules. In fact, SWRL rules have the form of an implication between an antecedent (body) and consequent (head). Whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold.

SWRL also supports a range of built-in predicates, which greatly expand its expressive power. SWRL built-ins are predicates that accept several arguments. They are described in detail in the SWRL Built-in Specification. The simplest built-ins are comparison operations, such as *swrlb:lessThan* ($>$), *swrlb:lessThanOrEqual* ($\leq$). These SWRL built-ins are useful for describing conditions on access policies. We have adapted the contextual rule approach proposed by our team in [Ramos 2007] for enforcing access control policies.

In our approach, the antecedent of an inference rule contains $facts$ (i.e., gathered contextual annotations) and the activation conditions (i.e., contextual and social constraints) of an action that refer those $facts$. The resultant is an operation (read, write) on the protected multimedia document that the $CxtANBAC$ should grant to the resource requestor. We describe our approach in the following:

- *Facts* are represented by contextual and social annotation meta-data associated with the protected multimedia and the user requesting access, respectively;
- Context constraints describe the valid contextual annotation (i.e., the *access_policy*;
- The resultant adds relations into the ontology to grant access on multimedia document (read or write operation).

[19] http://www.w3.org/Submission/SWRL/

```
context rule := '(' facts ') conditions '->' grant
facts := 'User(?user)' {^ Grant(operation)} {[fact]}
fact := {^ ('DataProperty |
            ObjectProperty |
                Individuals ')'}      /* Contextual Annotation */
conditions := {^ ('atom')'}           /* Atom includes SWRL expressions */
operation := 'read'| 'write'
grant := 'hasAccess (?user, true)'
{'^ hasAccessOf(?user, operation)'}
```

Figure 8.4: A part of the grammar EBNF of annotation-based access rules.

Figure 8.4 shows a part of the grammar EBNF (Extended Backus-Naur Form)[20] defined to represent annotation-based access rules. Rules has a variable indicating the users from the social network ($User(?user)$) and operations on multimedia resources available in the system (read, write). Then, attributes of contextual annotation meta-data are listed in the rule. These attributes may be optional in the access rule, since inference engines provide other forms for injecting fact, such as indicating an existing OWL document. Contextual conditions might make reference to social, computational, spatial, temporal, and spatio-temporal data described by a instance of annotation ontologies (see the proposed ontologies for describing annotations in Section 8.5). After executing the annotation-based access rules the access decisions will be taken, granting or denying access on the protected multimedia document.

We observed, however, that the task of defining annotation-based access policies is a complex activity to be performed by users. This occurs because users are asked to imagine situations in which they could possibly grant permission on their multimedia documents to other users. In order to facilitate this task and increase the usability of the $CxtANBAC$, we should offer users a pattern set of policies that can be used to protect their personal multimedia documents.

Therefore, we carried out a survey in order to identify the set of most relevant contextual annotations for defining annotation-based access policies. In this survey, we tried to identify the sharing behavior of users in order to propose a predefined set of access policy templates.

8.4 On-line Survey

We carried out a survey with 200 persons in order to identify the behavior of users when they share their personal photos. For the sake of simplicity, we considered only photo in this survey. Basically, we seek answers to the following questions: *What is the most common behavior of users with regarding to the sharing operation of their personal photos? What information that describes the situation (context)*

[20] http://www.garshol.priv.no/download/text/bnf.html

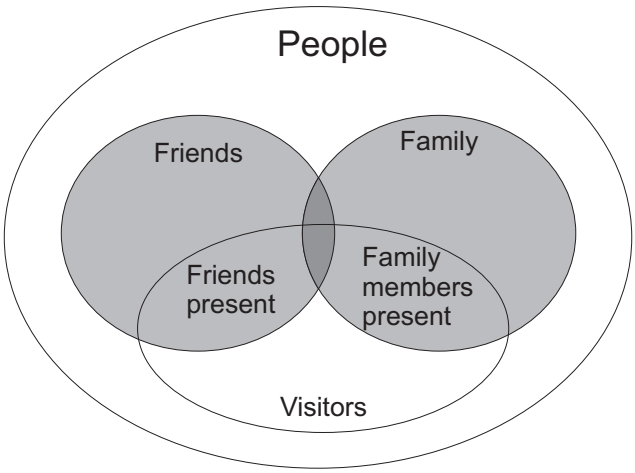


Figure 8.5: Social spheres considered for the study.

Sharing with ...	Family members present			Family			Friends present			Friends		
	AP	SP	NP	AP	SP	NP	AP	SP	NP	AP	SP	NP
Photos of family	73,0%	19,5%	6,0%	49,0%	42,0%	8,0%	21,0%	53,0%	26,0%	7,0%	47,5%	43,5%
Photos of friends	36,5%	47,5%	12,5%	12,0%	54,0%	32,0%	70,5%	25,0%	4,0%	25,5%	59,5%	13,5%
Photos of yourself	37,0%	51,5%	10,0%	21,5%	65,0%	12,0%	36,0%	53,5%	10,0%	20,0%	63,5%	15,0%
Photos of visitors	25,0%	42,0%	31,5%	12,5%	37,0%	48,5%	26,5%	38,5%	34,5%	13,0%	42,5%	41,5%

Sharing with ...	Everyone			Tourism office			Other visitors		
	AP	SP	NP	AP	SP	NP	AP	SP	NP
Photos of family	4,5%	44,0%	49,5%	1,0%	11,0%	87,0%	0,0%	11,5%	87,5%
Photos of friends	8,5%	53,5%	36,5%	0,5%	13,0%	86,0%	0,0%	13,5%	85,5%
Photos of yourself	7,0%	54,0%	37,5%	0,5%	14,0%	85,0%	0,5%	13,0%	86,0%
Photos of visitors	6,5%	35,5%	55,0%	5,5%	21,0%	73,5%	3,5%	21,5%	74,5%

Legend
Higher freq. of responses
Intermediate frequency
Lower freq. of responses
AP - All Photos
SP - Some photos
NP - No photo

Figure 8.6: Behavior of users when sharing personal photos.

at creation time of photos is more relevant to define access policies for protecting them?

In order to answer these questions, we carried out a survey that is composed by five parts: 1) Creating and sharing photos; 2) Grouping photos for sharing operations; 3) Sharing photos; 4) Accepting/rejecting photo recommendations; 5) Multimedia document types. Appendix B presents the questions of that survey (in French), and Appendix C presents the frequency tables obtained from the answers for each question. We will discuss in this chapter only the results obtained in the first and second part of that survey, since these section are the most important to guide us for defining the pattern set of access policies.

At the beginning of that survey, we ask the respondents to imagine the following scenario: *On a walk with some members of your family and some friends, you visited the castle of Versailles. During this visit, you took various photos using your smartphone. You took photos of your family, friends, the castle, yourself, and other visitants.*

Then, in the first part of that survey we ask the respondents to describe how

	Quite interesting	Interesting	Not very interesting	Not at all interesting	No preference
Location	61,5%	30,5%	3,5%	3,0%	1,5%
Date	47,5%	35,0%	8,0%	4,0%	5,5%
Season	6,5%	23,5%	30,0%	23,0%	16,5%
Time interval	17,5%	36,5%	18,0%	15,0%	11,5%
Activity	13,5%	38,0%	26,0%	12,5%	9,0%
Groups or persons	19,5%	45,5%	16,5%	11,5%	5,5%
Event	46,5%	45,5%	6,0%	0,5%	1,5%
Thematic	5,0%	25,0%	25,0%	31,0%	13,0%

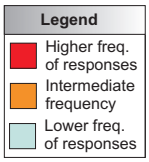


Figure 8.7: List of contextual annotation considered in that survey.

they share each set of photos (i.e., photos of family members, friends, yourself, and visitants) with the following set of people: their family, the family members present during the visit, their friends, their friends present during the visit, everyone, tourist office, and the visitors of the castle. Figure 8.5 illustrates a pictorial description of the social spheres considered in that survey.

Figure 8.6 illustrates the results of the first part of that survey. We will give a special attention to data that make up the higher frequency of responses (cells in red). We observe that people usually respects the social sphere of present persons at creation time of photos (percentages in white) when they share their photos with others. When we asked about the sharing of photos with each social group (family and friend) that were not present at creation time, we observe that people typically want to select the photos to share with them (see the percentages of column SP). Moreover, people do not want share their photos with others that are not part of their social network (see the percentages for everyone, tourism office, and other visitors). Thus, we conclude that people prefer to share their photos with people from their social network who were present at creation time. Moreover, people do not want share their photos with strangers.

In the second part of that survey we ask the respondents about their preference for grouping photos by using annotation information. Our idea in this part is to identify the most relevant set of contextual annotation for defining access control policies. We asked about the following list of contextual annotation: location, date, season, time interval, activity (e.g., skiing, cycling), groups or persons present at creation time of photos, event (e.g., wedding, city toor), and thematic (e.g., animals, flowers).

Figure 8.7 illustrates the survey results about the relevance of contextual annotation from user point view. We observe that location, date, and event composes the set of the most relevant contextual information for grouping photos. Time interval, activity, and groups and persons are also frequently considered by user for grouping their photos. Therefore, CxtANBAC supports this set of information (i.e., location,

date, time interval, event, and groups and persons) for defining annotation-based access control policies.

8.5 PPlog: Pervasive Personal Blog

We have evaluated the *CxtANBAC* to control access permission of multimedia content generated by the PPlog application. As described previously, PPlog offers users means of describing their daily life by using personal pervasive devices. PPlog is a client-server application. The PPlog client-side application is developed for the J2ME platform and the server side for the J2SE platform. PPlog client allows users to create and publish (i.e., send to the PPlog server) the following types of posts:

- *Multimedia post:* it is a multimedia post that can be a text message, a photo, a video, or an audio document created by users. Multimedia posts (i.e., multimedia files enriched with contextual annotation) can be automatically annotated with time, location, and Bluetooth address of pervasive devices around the users. This set of contextual information is gathered by the Context Provider (CP) deployed in the pervasive device. Each multimedia post is associated with an instance of *Post Annotation* Ontology proposed to describe the contextual annotation of that multimedia file;
- *Daily post:* it is a post that is composed by one or various multimedia posts, i.e., photos, videos, audios, and text messages. The main idea behind this kind of post is that it should represent the user's daily life. Thus, daily post should be published only once a day. However, PPlog client application imposes no restriction to the amount of daily posts that can be accomplished in a day. A daily post is associated with an interval time explicitly determined by users. In fact, the interval time is determined according to the moment that users initiate and complete a post by using the PPlog client application. Moreover, users can configure the PPlog client application to annotate this kind of post with the path taken by them (i.e., a set of GPS coordinates automatically gathered that compose their daily path). Each daily post is associated with an instance of *Daily Post Annotation* Ontology.

In the next section we present the proposed ontologies for annotating multimedia and daily posts.

8.5.1 Annotation of Multimedia and Daily posts

We have defined two ontologies using as basis the *CxtResource* Ontology for describing the annotation associated with multimedia and daily posts: *Post Annotation* Ontology and *Daily Post* Ontology. Figure 8.8 illustrates the contextual

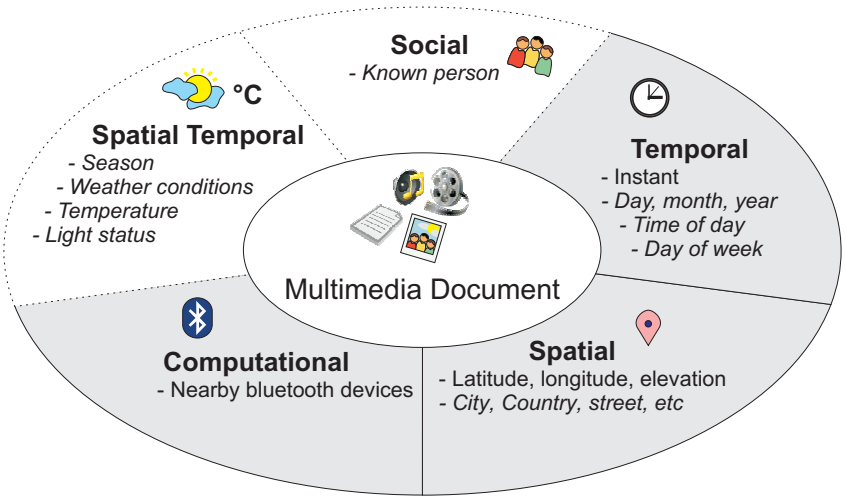


Figure 8.8: The contextual annotation associated with Multimedia posts.

information associated with multimedia posts that is used as basis to define the *Post Annotation* Ontology. The dimensions in gray (i.e., computational, spatial, and temporal) have some contextual information captured by the context provider (CP) of CxtMF, which is deployed in the pervasive device. For instance, nearby Bluetooth devices, instant, and the triple latitude, longitude, and elevation (i.e., the GPS coordinate) compose the set of gathered context information by the PPlog client application. The contextual information written in italics (e.g., day, month, year, time of day) are obtained by performing derivation/inference operations. The other dimensions (i.e., social and spatial-temporal) also will be inferred/derived by the *Context Reasoner Components* (CRC) of our CxtMF, which are integrated with the PPlog server-side applicaction (i.e., these operations will be performed only at publish time of multimedia and daily posts).

Figure 8.9 illustrates the *Post Annotation* Ontology. This ontology is used to describe the contextual annotation of multimedia posts (i.e., multimedia document). For each multimedia document created by the PPlog client application, an instance of that ontology will be create to describe the contextual information gathered at creation time of that multimedia document. $Multimedia\_Document$ class is a subclass of $cres : Resource$ defined in the CxtRes Ontology. $Multimedia\_Document$ has four subclass: $Video$, $Audio$, $Photo$, and $Text\_Message$. These classes can be annotated with tags (property $hasTag$) defined manually by users.

The class $time : Instant$ (from the Time ontology[21]) is used to annotate the temporal aspect of that multimedia content. We are reusing the NeoGeo ontology[22], that is an OWL representation of GML (Geographic Markup Language), to represent the spatial dimensions. Thus, the class $gml : Point$ represents the GPS coordinates

[21] http://www.w3.org/TR/owl-time/
[22] http://mapbureau.com/neogeo/neogeo.owl

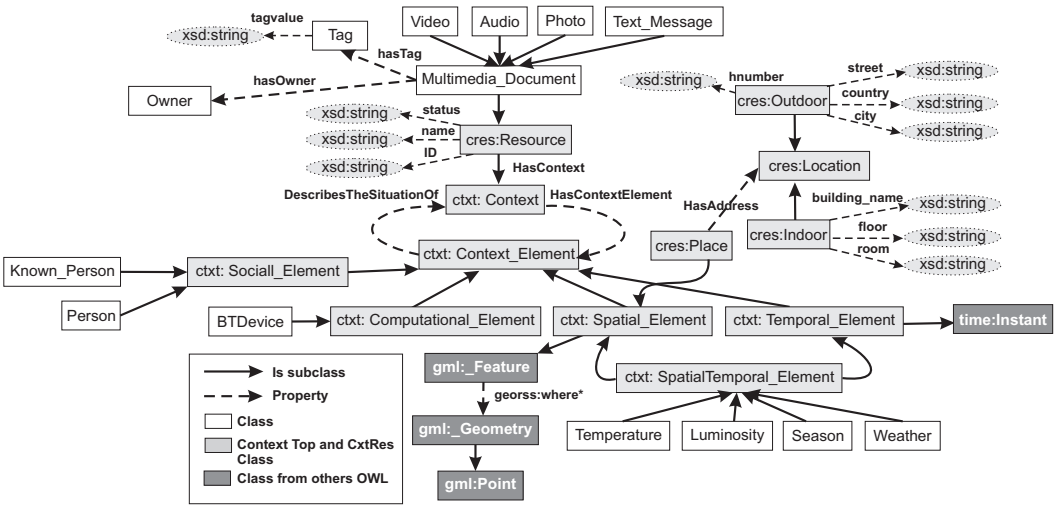


Figure 8.9: Post Ontology.

with the datatype property $gml : pos$. $BTDevice$ represents the nearby devices detected by using the Bluetooth interface. This class has the datatype property $btaddress$ to describe the Bluetooth address of nearby pervasive devices. In fact, this datatype property is used to derive the social dimensions (i.e., $Person$ and $Known\_Person$ concepts) from the $FOAF$ profile of users.

Figure 8.10 illustrates the annotations associated with daily posts. Each daily post is automatically annotated with computational (Bluetooth address of nearby devices), temporal (a interval defined with a start and end time), and spatial dimensions (a set of GPS coordinates,i.e., track points) by the PPlog Client Application. The social dimension is derived from the user's FOAF profile and the Bluetooth address of nearby pervasive devices detected around the user in his/her daily life.

Figure 8.11 illustrates the Daily Post Ontology defined for annotating daily posts. A daily post (class $Daily\_Post$) is associated with one or various $Multimedia\_Document$ (class $cmd : Multimedia\_Document$ of Post Annotation ontology). The track points are represented by $gml : Point$ class. $Interval$ class has two datatype properties: $start\_time$ and end_time. A daily post is annotated with one or more Bluetooth address of nearby pervasive devices sensed during the creation of a daily post. We use this information to derive the $persons$ and $known\_persons$ that we met during the creation of a daily post.

8.5.2 Annotating Social Relationships

By using the PPlog client application, users are able to annotate manually their contacts (i.e., social network), classifying them into personalized groups (e.g., Friend, Family, Football fellows, etc). This process is performed by users during the entire cycle life of PPlog application. PPlog client application offers functionalities to easily

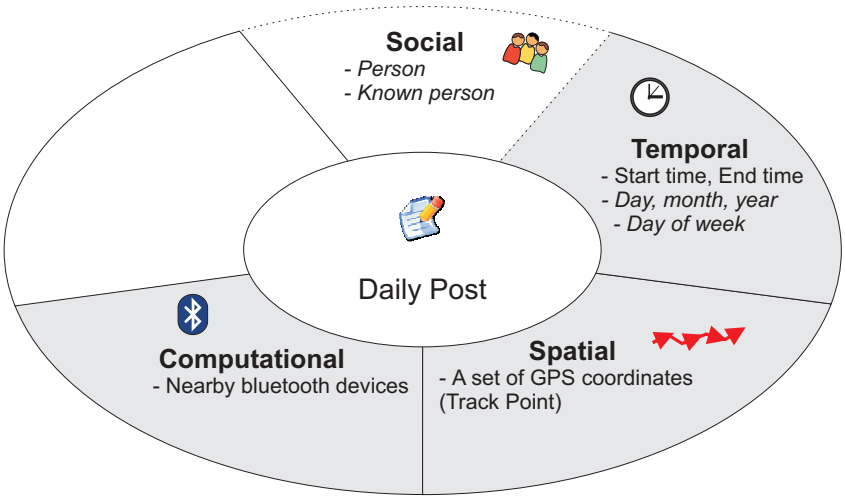


Figure 8.10: The contextual annotation associated with Daily posts.

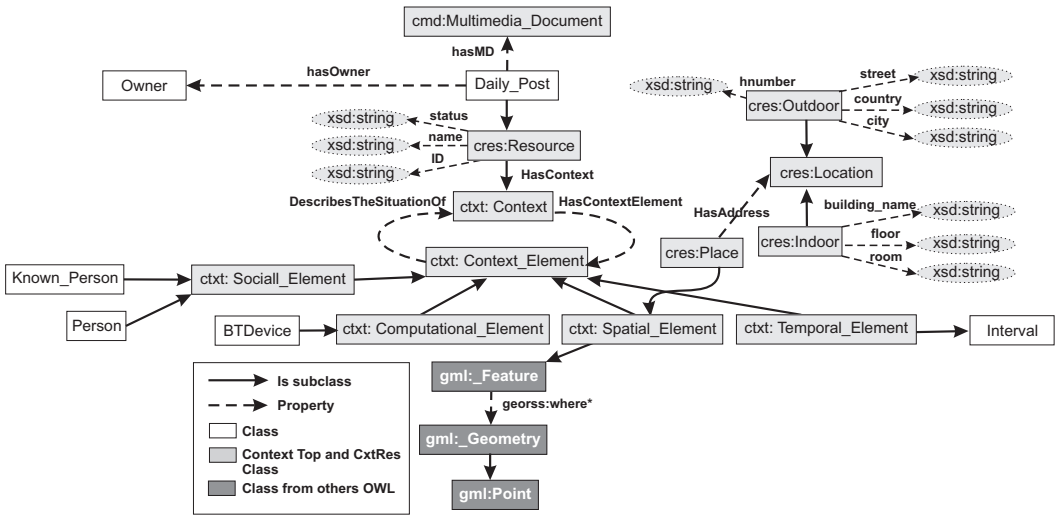
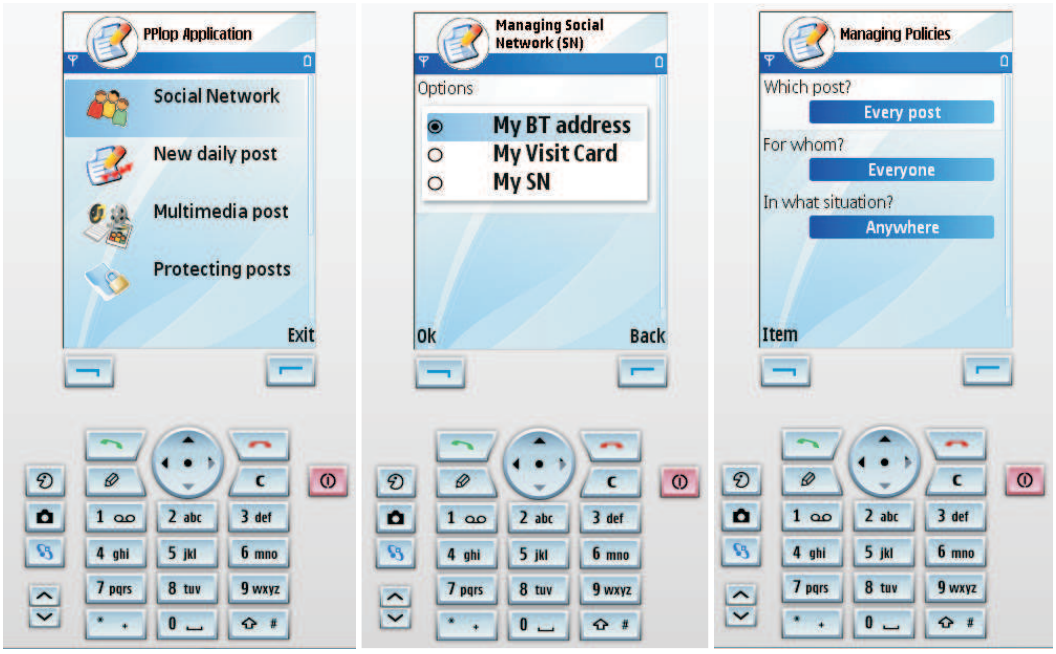


Figure 8.11: Daily Post Ontology.



a) Main menu **b) Managing Social Network** **c) Managing policies**

Figure 8.12: Screen shots of main features of PPlog client application.

define social annotations. As described previously, we have extended the FOAF ontology in order to accommodate new social classifications, such as the properties *Friend*, *Family*, *WorkFellow*, and *PersonalizedRelation*. These properties are, in fact, subproperties of $foaf:knows$.

To support personalized relationships defined by users, we defined the subproperty *PersonalizedRelation* that has a datatype property named *relationName* for describing the name of the new relationship defined by users. Moreover, we have added a new property, named *BTDevice*, to describe the Bluetooth address of user's pervasive devices. With this information it will be possible to infer the persons around users at creation time of multimedia and daily posts. The instance of that extended FOAF profile should be synchronized with the PPlog Server Application in order to offer means of exploring social annotation.

8.5.3 PPlog client application

PPlog client application supports the following main features: configure user's social network based on an extended version of FOAF profile; create and send multimedia and daily posts; and define/manage SWRL policies for protecting created posts. Figure 8.12 illustrates some screen shots of PPlog client application. The main menu offers users the following features (Figure 8.12a):

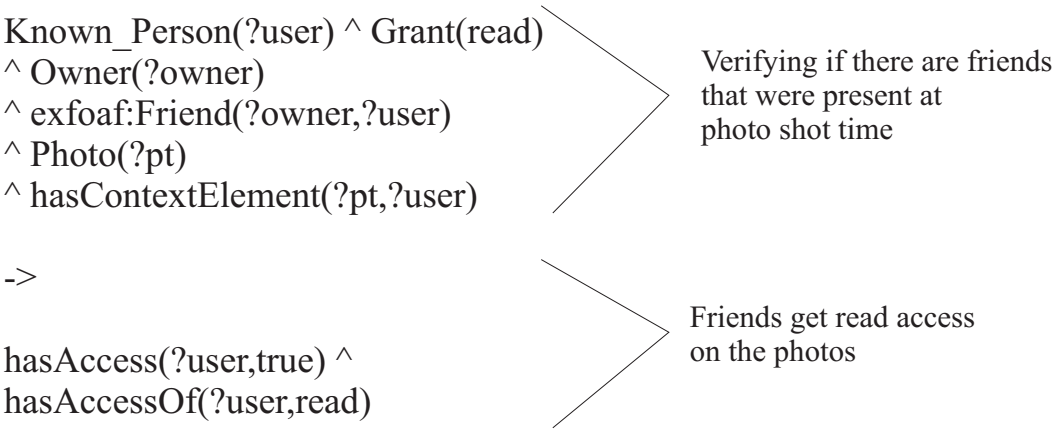


Figure 8.13: Example of access control SWRL rule.

- *Social network:* in this option users will define their social relationships with other users. It offers three main operations: detect local Bluetooth address (My BT Address), configure and send visit card (My Visit Card), and configure social network (SN) (My SN)(see Figure 8.12b). The option *My BT Address* requests the Context Provider (CP) in order to discover the Bluetooth address of user's pervasive device. This information is registered in the user's FOAF profile. The option *My Visit Card* offers users a form for defining their visit card that can be composed by the following information: First and Last name, Bluetooth address of her/his device, profession, telephone number (house and mobile), a photo, e-mail, home page, and personal address. Moreover, there is an option for sending the visit card of users. Users can add a new contact in their social network by two means: receiving visit cards of other users; adding manually new contacts in their social network. When users send their visit card, they are able to agree/disagree the disclosure of their Bluetooth address. In this case, only users that have disclosed their Bluetooth address might be annotated with the posts for describing the social dimension (i.e., persons present at post creation time). When users receive a visit card in her/his personal pervasive device, they can save it and annotate users in their social network. The option *My SN* offers users operations for managing their social network, such as insert/modify/remove contacts and synchronize the FOAF with the server. Figure 8.14 illustrates an instance of Extended FOAF ontology generated by the PPlog client application for describing User1's social relationships. User1 knows three other users: User2 (Friend), User3 (Family), and User4 (WorkFellow). User1 has received the visit cards of each other user in order to define his/her social network;

- *New daily post:* in this option users are able to create new daily posts. According to the PPlog configuration defined by the user (option configuration), track point and Bluetooth address of nearby devices will be gathered and associated with daily posts by means of annotation. Each daily post has a title

and can be annotated manually with personalized tags;

- *Multimedia post:* in this option users are able to create multimedia files that will be automatically annotated with contextual annotation. Moreover, users are able to annotate these posts with personalized tags. Text messages are limited to 140 characters, and the contextual annotation associated with video and audio files are gathered only at start time when creating these multimedia posts;

- *Protecting posts:* this option is the main PPlog client module in charge of controlling access on multimedia and daily posts (see figure 8.12c). In this option, users are able to define two types of policies for protecting their posts according to the scope of applicability: global and specific policies. Global policies might be applied to one or various posts created by users, while specific policies are defined to protect a determined post created in a specific situation. In fact, global policies are translated in specific policies and applied for each affected post created after their activation. The PPlog application form for defining policies asks users the following parameters: *Which post? For whom? In what situation?* In the *which post* question, users can select one of the following options: *every post*, *every multimedia post*, *every photo post*, *every video post*, *every audio post*, *every text post*, *every daily post*, and *personalized list of posts*. The last option asks users for the list of post in which policies will be applied. In the question *For whom?* users can select one of the following options: *everyone*, *list of groups*, and *list of users*. If users select one of the options *list of groups* or *list of users*, they should inform the social annotation (e.g., friend, family, etc) and names of users from their social network (i.e., FOAF profile), respectively. In the question about the situation, users can select *anywhere* or *nearby me*. The option anywhere do not take into account contextual annotation associated with posts, while *nearby me* considers only the persons selected in the option *For whom?* that were present at creation time of the posts selected in the option *Which post?* For instance, User1 intends to grant read access on her photos to users annotated as Friend who were present when the photo was taken. In this case, in *Which post?* User1 should select the option *every photo post*, in *For whom?* User1 should select the option list of group and after the sub-option *Friend*, and *In what situation?* User1 should select the option *nearby me*. After creating this policy, each instance of Post Annotation ontology associated with a created photo will contain the SWRL rule illustrated in the Figure 8.13.

- *Configuration* : users are able to define some configurations, such as frequency of gathering context information (location and BT devices).

```
<rdf:RDF
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:foaf="http://xmlns.com/foaf/0.1/"
     xmlns:admin="http://webns.net/mvcb/">
<foaf:PersonalProfileDocument rdf:about="">
  <foaf:maker rdf:resource="#me"/>
  <foaf:primaryTopic rdf:resource="#me"/>
</foaf:PersonalProfileDocument>
<foaf:Person rdf:ID="me">
<foaf:name>Pervasive User1</foaf:name>
<foaf:title>Mr</foaf:title>
<foaf:givenname>User1</foaf:givenname>
<foaf:family_name>Pervasive</foaf:family_name>
<foaf:nick>user1</foaf:nick>
<foaf:mbox rdf:resource="mailto:user1@domain.com"/>
<foaf:depiction rdf:resource="user1.jpg"/>

<exfoaf:BTDevice>00:0A:D9:EB:40:C9</exfoaf:BTDevice>

<exfoaf:Friend>
  <foaf:Person>
    <foaf:name>user2</foaf:name>
    <exfoaf:BTDevice>00:0A:D9:EB:66:C7</exfoaf:BTDevice>
    <foaf:mbox rdf:resource="mailto:user2@domain.com"/>
    <rdfs:seeAlso rdf:resource="http://www.domain.com/user2.owl"/>
  </foaf:Person>
<exfoaf:Friend>

<exfoaf:Family>
  <foaf:Person>
    <foaf:name>user3</foaf:name>
    <exfoaf:BTDevice>00:0A:D9:EB:50:B3</exfoaf:BTDevice>
    <foaf:mbox rdf:resource="mailto:user3@domainX.com"/>
    <rdfs:seeAlso rdf:resource="http://www.domainX.com/user3.owl"/>
  </foaf:Person>
<exfoaf:Family>


<exfoaf:WorkFellow>
  <foaf:Person>
    <foaf:name>user4</foaf:name>
    <exfoaf:BTDevice>00:0A:D9:EF:33:Q2</exfoaf:BTDevice>
    <foaf:mbox rdf:resource="mailto:user3@domainY.com"/>
    <rdfs:seeAlso rdf:resource="http://www.domainY.com/user4.owl"/>
  </foaf:Person>
</exfoaf:WorkFellow>

</foaf:Person>
</rdf:RDF>
```

User1 and his BT address

User2 that is Friend of User1

User3 that belongs to the User1's family

User4 that is a workfellow of User1

Figure 8.14: Instance of Extended FOAF profile.

8.5.4 PPlog server application

PPlog server application offers users web-based interfaces for managing and visualizing published posts. After being identified by the system, users are able to access their published posts and the published posts of their social contacts. Figure 8.15 illustrates the initial page of the User1's plog.

At the left window, we can see the last daily post published by User1. It is composed by a text, a photo and a video (i.e., three multimedia posts). Each multimedia document is geo-referenced and can be seen on the map by clicking in the link *See it on the map*. At the bottom window, we can see the photo of persons from User1's social network detected during the creation of this post (i.e., User1 may have met User2 at creation time of that daily post). Moreover, we can see the path taken by the User1 when he/she made that post. At the top of right window, we can see the hierarchy of posts classified by date (year, month, and title of post). At the bottom of right window, we have the contacts from User1's social network. We can access the Plog page of our social contacts by clicking on their photos.

Figure 8.16 illustrates the User2's plog page. *User1* will see only the posts disclosed to him (i.e., the posts in which *User2* has defined a SWRL rule grating read access to *User1*). When *User1* tries to access a post by clicking on its title, PPlog server application requests the CxtANBAC for enforcing the SWRL rules associated with that protected post. Then, if the resultant instance of Annotation Post ontology associated with that post has the properties *hasAccess* and *hasAccessOf* associated with User1, then PPlog server application will show it to *User1*. The enforcing process of CxtANBAC policies are described in the next section.

8.5.5 PPlog application integrated with CxtANBAC and CxtMF

Figure 8.17 illustrates an overview on PPlog application. PPlog application is built on CxtANBAC and CxtMF. The main components of CxtANBAC is based on the eXtensible Access Control Markup Language(XACML)[23] entities, such as PEP (Policy Enforcement Point) and PDP (Policy Decision Point) components. PEP and PDP are entities in charge of querying and enforcing process of annotation-based access control policies, respectively (see in [Filho 2009] for more details). We have used the server-based approach described in Section 6.14 for implementing the CxtANBAC. In this case, PEP and PDP entities are deployed in the server side.

When users make multimedia and daily posts, PPlog client application requests location (l), time (t), and Bluetooth address (bt) of nearby devices to the Context Provider (CP) deployed in the pervasive device. We have implemented three sensor components that are in charge of gathering GPS coordinates (GPSSensor), Bluetooth address (BTSensor), and time (TimeSensor), respectively (see in Figure 8.17).

[23] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

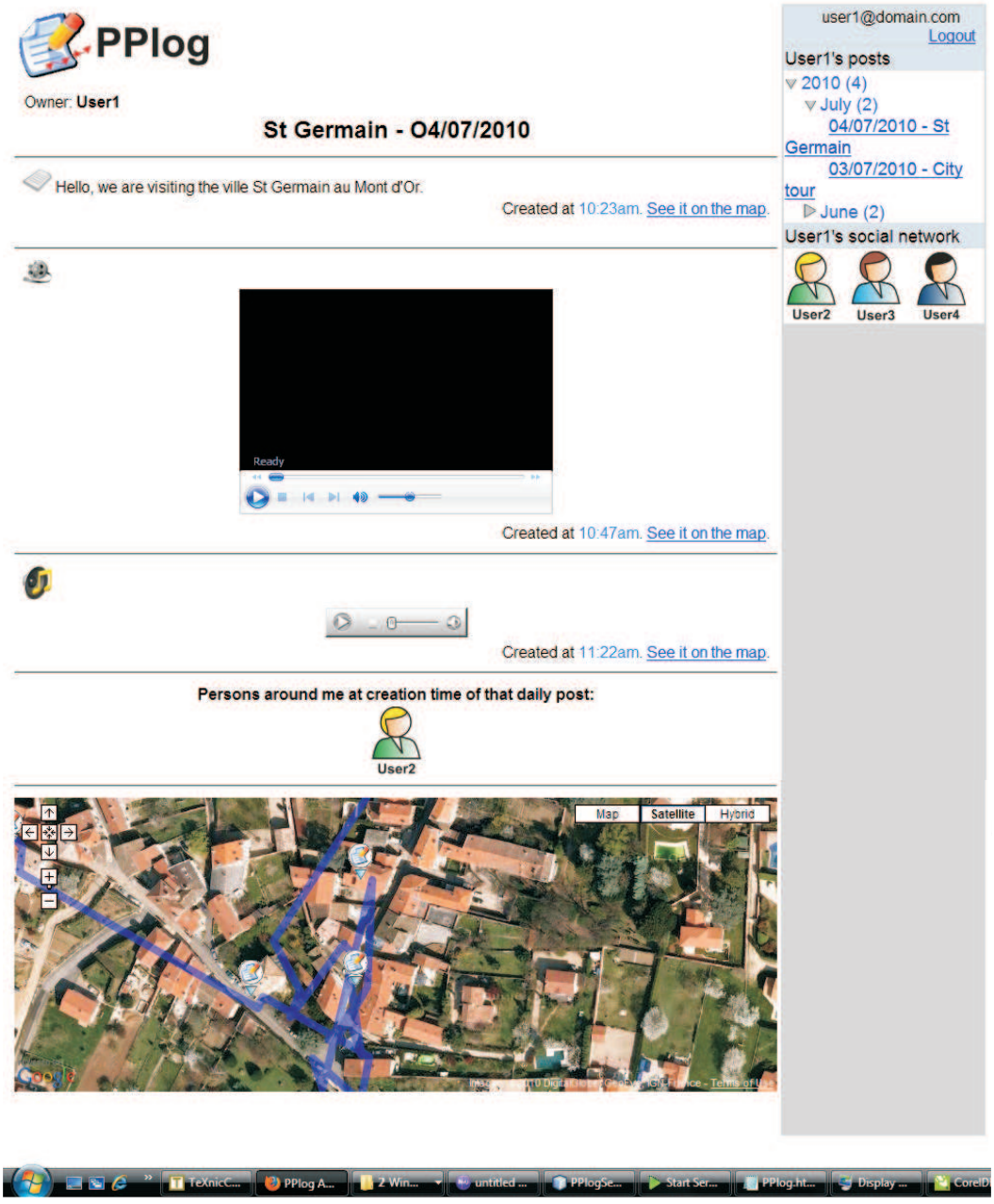


Figure 8.15: The main interface of PPlog service application.

Figure 8.16: The PPlog main page of User2.

In this case, PPlog client application (i.e., the context consumer) communicates directly with the CP. CP has the PPlog client registered as a trusted context consumer.

After creating multimedia and daily posts, users should transfer them to the PPlog server application in order to publish these posts using one of the two synchronization modes: connecting directly with the server application for sending them or synchronizing these posts with a desktop computer that will upload them to the PPlog server application (1). Multimedia and daily posts are composed by multimedia documents (text message, video, audio, photo) and instances of Post Annotation and Daily Post ontologies, respectively. SWRL rules defined by users are embedded into these ontology instances, which will be enforced by the PDP in order to make access control decisions.

When the PPlog server application receives multimedia and daily posts, it requests the CIS sub-services (Context Reasoner) of CxtMF (2) for inferring or deriving new annotations associated with multimedia and daily posts, sending these enriched instances (3) to the PCP (Policy Context Information Point). PCP receives these instances and updates their base of protected resources. In this case, each instance of Post Annotation and Daily Post ontology is associated with a protected resource.

When a user tries to access a published post by using the PPlog server application, this last requests access permission to the CxtANBAC PEP (4). Then, PEP requests access permission on that protected post to PDP (5). PDP requests (6) to

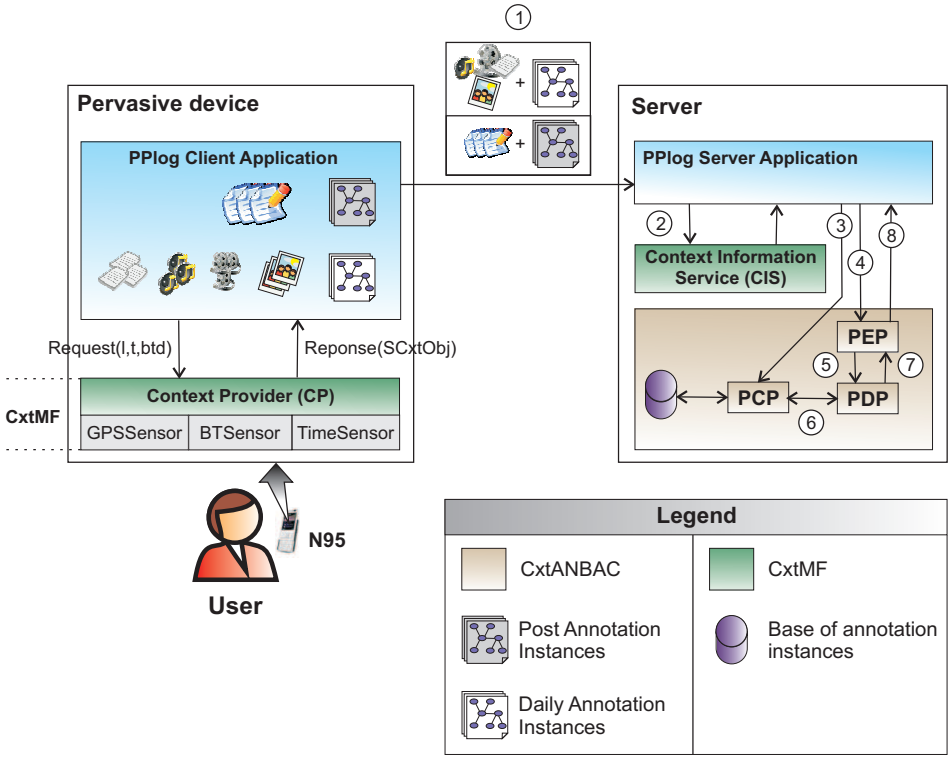


Figure 8.17: PPlog Application built on CxtANBAC and CxtMF.

PCP the ontology instance associated with the requested post. On receiving this instance, PDP enforces the SWRL rules for making access control decisions. Finally, PDP grants/denies access (7) on the requested resource. PEP receives the decision resending it to the PPlog server application.

8.6 Conclusion

This Chapter presented an instantiation of S-CxtBAC for protecting personal multimedia applications, named CxtANBAC. We presented an application, named PPlog, that was integrated with CxtANBAC and CxtMF in order to protect multimedia and daily posts. CxtANBAC uses social, contextual, and social-contextual information for defining annotation-based access control policies.

Contextual annotation can be attached to multimedia documents and used for defining access control policies. In fact, CxtANBAC extends the existing annotation-based access control approaches by supporting social, contextual, and social-contextual annotations. Owners of multimedia documents are able to define access control policies based on social and contextual information for protecting their resources. According to our knowledge, none of existing annotation-based access control approaches consider this kind of annotation when making access control decisions.

We have used semantic technologies (ontologies, SWRL rules) for describing and enforcing annotation-based access control policies. When implementing the PPlog application, we observed that CxtANBAC can be used as a service for adapting multimedia content.

We plan to extend CxtANBAC in order to take into account the context of resource requestor and resource owner at request time of protected resources. In the current version, CxtANBAC considers only the context of resource at creation time and the existing social relationships among resource owner and resource requestor. In addition, we plan to integrate a mechanism to dynamically and statically detect and resolve conflicting access control policies.

Chapter 9

Conclusion and Future Work

Résumé: *Ce chapitre présente les conclusions de ce travail de thèse. Nous présentons brièvement les principales contributions : la famille de modèle de contrôle d'accès, l'architecture de gestion d'information contextuel, les méthodes d'estimation des indicateurs de qualités, les modèles d'informations sémantiques contextuelles et de qualités et l'intégration de l'architecture avec la famille des modèles proposée. Finalement, nous présentons quelques perspectives pour la continuité de ce travail de recherche.*

Contents

9.1 Summary of Contribution

The main objective of this Thesis is the proposal of using context information for making access control decisions in pervasive environments. It also aims to determine the impact of the quality of context information on the context sensitive applications and services.

We discussed the traditional access control models (Chapter 2) and the existing context-based and context-aware access control approaches (Chapter 3), presenting the advantages and disadvantages of using each solution for protecting pervasive resources. In Chapter 5, we pointed out the need for defining access control models

that use context as central concept for enforcing access control policies. Moreover, as context information is used to authenticate users in the same way that identity and role are used in traditional access control solutions for granting permission, it is also necessary to gather context information with quality and security, preserving the privacy of users.

Most of the existing approaches extend the RBAC, which is a model by nature unsuitable for pervasive environment, because it is context-unaware and user and permission assignments are statically defined. In addition, existing solutions do not take into account the quality of context information used for making access control decisions, and do not make clear distinction among context associated with the access entities and the environment: resource requestor, resource owner, and resource itself.

With this in mind, we proposed the *CxtBAC* (Chapter 6) that defines context-based access control models for pervasive environments. In addition, we proposed the *CxtMF* (Chapter 7) that is a context management framework in charge of gathering context information with quality and security, taking into account privacy requirements of users. Such family model may be used as a basis to define new context-based access control solutions for protecting pervasive resources, i.e., *CxtBAC* can be instantiated/extended in order to accommodate new requirements of pervasive environments.

This chapter describes our contributions and some perspectives of this work. It is divided into three major results:

- *CxtBAC:* the specification of a family of context-based access control models, named *CxtBAC*;
- *CxtMF:* the development of a QoC and privacy-aware context management framework, named *CxtMF*;
- *Instantiation of the CxtBAC:* the instantiation of a *CxtBAC* ($S - CxtBAC$) model applied to the domain of Mobile Multimedia Applications.

9.1.1 CxtBAC Models

The proposed family of context-based access control models offer means of assigning access permission based on the concept of *access context*, different from the concept of roles. *CxtBAC* is composed by 8 models that can be used to implement access control solutions for pervasive environments. In pervasive environment, the interactions (i.e., user-to-user and user-to-environment) occur in a ad-hoc manner, mainly resulting from the user mobility situations. Therefore, in some scenarios the concept of role is not applied in a natural way, since the user can not be recognized by the environment.

We adopted the concept of *access context* that represents the situation in which users (resource owners and resource requestors) and resources are part at request time of a protected resource. *Access context* is dynamically assigned to users since their associated context constraints are evaluated as true according to the current context of the observed entities (resource owner, resource requestor, resource, and environment).

Basically, the difference between *CxtBAC* and the existing works described in Chapter 2 and Chapter 3, is the focus given on the specification of access control models that use contextual information as basis for assigning access permission to users. Moreover, *CxtBAC* takes into account the existing social relationship between users, the quality and privacy requirements on context information used for making access control decisions.

CxtBAC can be instantiate as a discretionary or mandatory access control model, offering means of defining user-level and system-level access control policies, respectively. However, a service in charge of context management operations should be built in order to support these CxtBAC instances. Thus, we proposed a context management framework (CxtMF) that is in charge of gathering, inferring, deriving, protecting, and providing QoC-enriched context information for CxtBAC-based solutions.

9.1.2 CxtMF framework

The component-based *CxtMF* is composed by services in charge of managing context information, performing operations such as gathering, inferring, deriving, protecting, and providing context information for registered consumers.

Unlike the existing work described in Chapter 4, *CxtMF* supports QoC at all steps involved in the context management process. Moreover, the proposed approach of QoC measuring can be applied on both raw context data and high-level context information obtained from inference and derivation operations.

Context consumers are able to define QoC thresholds that the *CxtMF* should take into account before providing context information. Context information can be enriched with QoC (i.e., QoCI and QoCP) in order to offer means of improving context sensitive decisions. Global QoC thresholds can be also defined internally to the *CxtMF*.

For example, context providers (CP) can use these thresholds in order to filter information from two or more registered sensors of the same type. *CP* can eliminate the raw sensed data that do not reach the set of Global QoC thresholds at the gathering level in order to reduce the processing load. At application or service level, QoC can be used to verify the quality of the context used for decision making, reducing thus the likelihood of making a wrong decision.

The Context and QoC models proposed in this work offer means of representing

semantically context and QoC information managed by the $CxtMF$. For each type of observed entity (i.e., user, resource, and environment) we have defined a new ontology model reusing the Context Top Model that we proposed in [Viana 2008]. Finally, the *Access Control* ontology is defined by reusing these ontologies to describe the context of the observed entities that is relevant for decision making.

9.1.3 CxtBAC instantiation

In order to validate the $CxtBAC$ and the $CxtMF$ in a real scenario of application, we instantiate the $S{-}CxtBAC$ (Social-Aware CxtBAC model) that was built on the $CxtMF$ for protecting mobile multimedia applications. The access control decisions are based on the social relationship between users and the context characterizing the creation situation of resource (i.e., context of resource). We developed a client-server application, named PPlog (Pervasive Personal Blog), which is able to annotate multimedia resources (i.e., video, photos, mini Blogs) automatically with context information at creation time.

The annotation associated with multimedia resources is used to enrich the user blog with context information automatically gathered from the environment at creation time. Moreover, it can be used to define access control policies for protecting annotated resources. For example, *a user could define an access policy for protecting his/her photos, allowing only the persons from his/her social network present at creation time to access them.*

PPlog used a CP to gather context information for annotating multimedia resources, and offer users means of requesting multimedia resource of any person belonging their social network. We used a semantic approach for evaluating access control policies, which are described as SWRL rules. Moreover, we conducted an on-line survey of 200 people in order to identify the most important set of annotations that can be used to define policies for protecting personal photos based on context information.

9.2 Future Work

The study that we presented in this thesis leaves us some interesting perspectives for the continuation of our work. We classify this future work in two categories: short-term and long-term goals.

9.2.1 Short-term goals

As a short-term perspective, we plan to extend the $CxtBAC$ with a new model for integrating context with roles. The main objective is to facilitate the transition of RBAC-based solutions existing in legacy systems with our new approach. However,

as that new model will use context information for adapting permission, it should take into account also the quality of context information. It is very important to take into account this requirement, since user's mobile devices are part of the chain of generating context information used for making decisions.

We observed in Chapter 8 that the definition of context-based access control policies is not an easy and intuitive task. Therefore, we plan to develop a user-friendly interface that allows users or administrators to easily define context-based access control policies described semantically as SWRL rules. In addition, we plan to instantiate the *CxtBAC* for protecting resources in a real professional scenario. We conducted an on-line survey of 60 people in order to identify the most important set of context-based access control policies for protecting personal resources in a professional environment. We plan to use an XACML-based approach for evaluating the context-based access control policies of this *CxtBAC* instance.

Moreover, we plan to extend the QoC evaluating components (QoCE) for measuring the other QoC indicators (e.g., trust-worthiness, correctness, accuracy) and to verify their effectiveness. Moreover, we plan to verify how these QoC indicators depend on each other (e.g., in our proposition, the QoCI completeness is dependent of QoCI up-to-dateness).

9.2.2 Long-term goals

As a long-term perspective, we expect to specify a QoC-based policy language for describing QoC requirement of context consumers. Policies defined using that language should take into account one or more QoC indicators in order to improve the context based decisions. We plan to add a management mechanism to control reliable and unreliable sensors/context providers. This mechanism should be able to answer the following questions: *What characterizes a reliable sensor/CP? When a reliable sensor/CP becomes a unreliable entity, and vice versa?*

Another aspect that should be considered in the future work is the integration of context-based authentication services in the *CxtBAC* and *CxtFM*. We expect to conduct a study in order to identify the existing authentication approaches that only use context information to authenticate users (e.g., voice, image, movement, proximity to someone, biometric sensors, etc). For example, permission could be granted to the users based on the certainty level of authentication methods.

We expect to propose an adapting approach for security mechanisms (e.g., the service used for protecting end-to-end communication channels) that takes into account the *context of risk*. The main idea is to identify the context of risk (e.g., the risk of having information captured by an unauthorized entity is greater if the user is connected to a network of an airport than if he/she is connected to his/her home network). Therefore, the context of risk is any information that can be used to characterize a situation of risk for the security of applications and users. According to

the identified context of risk, the security infrastructure is able to adapt internally its security mechanisms in order to protect user and application data.

Finally, we want to point out as a prospect the integration of our solutions in cloud computing environments. The emergence of such paradigm leads to very interesting issues about privacy and security aspects, such as authentication, authorization, access control, and delegation.

Appendix A

Publication List

A.1 Conference and Workshop

1. Bringel Filho, J., Viana, W., Gensel, J., Martin, H. A Contextual Annotation-based Access Control Model for Pervasive Environments. In: Second International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use. Second International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use. Conference PERVASIVE, 2010.

2. Bringel Filho, J., Martin, H. A Generalized Context-based Access Control Model for Pervasive Environments, 2nd SIGSPATIAL ACM GIS International Workshop on Security and Privacy in GIS and LBS (SPRINGL 2009), November 3, Seattle, WA, USA.

3. Bringel Filho, J., Martin, Hervé. QACBAC: An owner-centric QoC-Aware Context-Based Access Control Model for Pervasive Environments. 1st ACM GIS Workshop on Security and Privacy in GIS and LBS, SPRINGL 2008, November 4, Irvine, CA, USA.

4. Bringel Filho, J., Martin, Hervé. A Quality-Aware Context-Based Access Control Model for Ubiquitous Applications. Third IEEE International Conference on Digital Information Management ICDIM 2008, November 13-16, London, UK.

5. Bringel Filho, J., Martin, Hervé. Using Context Quality Indicators for Improving Context-based Access Control in Pervasive Environments. IEEE/IFIP International Symposium on Trust, Security and Privacy for Pervasive Applications (TSP-08), Shanghai, China - December 17-20.

6. Bringel Filho, J. Gestion d'informations contextuelles relatives à la vie privée dans les systèmes pervasifs. In: INFORSID 2007, Perros-Guirec, 2007.

A.1.1 Joint Work

1. Oliveira, M., Hairon, C., Andrade, O., Moura, R., Sicotte, C., Denis, J., Fernandes, S., Gensel, J., Bringel Filho, J., Martin, H. A context-aware framework for health care governance decision-making systems: A model based on

the Brazilian Digital TV. In: IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, 2010, Montreal, QC, Canada. WoWMoM 2010. Los Alamitos, CA : IEEE Computer Society, 2010. p. 1-6.

2. Oliveira, M., Odorico, L., Bringel Filho, J., Mesquita, P., Figueiredo, V., Hairon, C. Pimenter-L: o Computador 24 horas Sensível ao Contexto para Aplicações em Saúde. 9th I2TS - International Information and Telecommunication Technologies Conference. Rio de Janeiro State, Brazil, December 13-15, 2010.

3. Viana, W., Bringel Filho, J., Gensel, J., Villanova-Oliver, M., Martin, H. Déploiement adaptatif d'applications mobiles et sensibles au contexte adaptation. 8ème atelier sur l'Evolution, la Réutilisation et la Traçabilité des Systèmes d'Information. INFORSID 2009, May 29, Toulouse, France.

4. Viana, W., Bringel Filho, J., Freire Junior, J. C., Gensel, Jerome, Vilanova-Oliver, Marlene, Martin, Hervé. CAUS: Uma arquitetura sensível ao contexto e orientada a componentes para sistemas de administração de ensino. In: SBCUP - I Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2009, Bento gonçalves. SBCUP - I Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2009.

5. Bergeret, G., Viana, W., Bringel Filho, José, Celso Freire, J., Villanova-Oliver, M., Gensel, J. Martin, H. Une architecture sensible au contexte pour le développement des systèmes d'administration pour l'enseignement. 5èmes journées Francophones Mobilité et Ubiquité (UBIMOB'09), July 7-8, 2009, Lille, France.

6. Viana, W., Bringel Filho, J., Gensel, J., Villanova-Oliver, M., Martin, H. PhotoMap - Automatic Spatiotemporal Annotation for Mobile Photos. W2GIS 2007, p. 187-201.

7. Viana, W., Bringel Filho, J., Gensel, J., Villanova-Oliver, M., Martin, H. A Semantic Approach and a Web Tool for Contextual Annotation of Photos Using Camera Phones. WISE 2007, p. 225-23.

8. Viana, W., Bringel Filho, J., Gensel, J., Villanova-Oliver, M., Martin, H. PhotoMap: annotations spatio-temporelles automatiques de photos personnelles pour les utilisateurs nomades, Conférence Québéco-Française de Développement de la Géomatique CQFD-Géo 2007, Clermont-Ferrand, June 20, France, 2007 (in French).

A.2 Journal

1. Bringel Filho, J., MARTIN, H. Towards Awareness of Privacy and Quality of Context in Context-Based Access Control for Ubiquitous Applications. Jour-

nal of Digital Information Management, v. 7, p. 219-226, 2009.

A.2.1 Joint Work

1. Viana, W., Bringel Filho, J., Gensel, J., Villanova-Oliver, M., and Martin, H. 2008. PhotoMap: from location and time to context-aware photo annotations. J. Locat. Based Serv. 2, 3 (Sep. 2008), 211-235.

Appendix B

On-line survey: Contextual Annotation

Partage de contenu assisté par téléphone portable

Dans le cadre de la thèse de Monsieur José BRINGEL (La boratoire d'Informatique de Grenoble - LIG), nous di ffusons ce questionnaire qui va nous permettre d'étudier l'utilisation des téléphones portables pour la gestion e t le partage de contenu personnel (photos, vidéos, etc.). Cette étude nous permettra d'enrichir ce travail d e recherche et également de proposer des solutions de cont rôle d'accès adaptés aux besoins des utilisateurs de télép hones portables. Ainsi, votre avis nous intéresse réellement et nous aimerions que vous répondiez à ce q uestionnaire.

1.Création et partage de photos

1.1 Question générale

a) À quelle fréquence prenezvous des photos en ut ilisant votre téléphone portable dans les situation s suivantes ?

	Plus de 3 fois par semaine	Jusqu'à 3 fois par semaine	Rarement	Jamais
Avec vos amis	○	○	○	○
Avec votre famille	○	○	○	○
Durant des événements	○	○	○	○
Lors d'activités	○	○	○	○

b) Merci d'indiquer cidessous si vous prenez des p hotos dans d'autres circonstances et à quelle fréqu ence (tous les jours, très souvent, rarement, jamais) :

1.2 Question spécifique

Imaginons qu'à l'occasion d'une ballade en famille ou avec vos amis, vous avez visité le Château de Ve rsailles. Lors de cette visite des photos ont été prises avec vos téléphones porta bles : des photos de votre famille, de vos amis, de vous-même, du château et d'autres personnes. Nous aimerions savoir comment v ous choisiriez de partager ces photos prises lors d e cette visite. Pour cela nous vous demandons de répondre aux questions suiva ntes:

a) Avec les membres de votre famille présents lors de la visite, vous partagez les photos sur lesquell es se trouve(nt) :

	Toutes les photos	Certaines photos	Aucune photo
votre famille présente lors de la visite	○	○	○
vos amis présents lors de la visite	○	○	○
vous-même	○	○	○
d'autres personnes	○	○	○

b) Avec d'autres membres de votre famille, vous par tagez les photos sur lesquelles se trouve(nt) :

	Toutes les photos	Certaines photos	Aucune photo
votre famille présente lors de la visite	○	○	○
vos amis présents lors de la visite	○	○	○
vous-même	○	○	○
d'autres personnes	○	○	○

c) Avec vos amis présents lors de la visite, vous p artagez les photos sur lesquelles se trouve(nt) :

	Toutes les photos	Certaines photos	Aucune photo
votre famille présente lors de la visite	○	○	○
vos amis présents lors de la visite	○	○	○
de vous-même	○	○	○
d'autres personnes	○	○	○

d) Avec d'autres amis, vous partagez les photos sur lesquelles se trouve(nt) :

	Toutes les photos	Certaines photos	Aucune photo
votre famille présente lors de la visite	○	○	○

Figure B.1: On-line survey: contextual annotation.

vos amis présents lors de la visite
vous-même
d'autres personnes

e) Avec tout le monde, vous partagez les photos sur lesquelles se trouve(nt) :

	Toutes les photos	Certaines photos	Aucune photo
votre famille présente lors de la visite			
vos amis présents lors de la visite			
vous-même			
d'autres personnes			

f) Avec l'office du tourisme de Versailles ou l'adm inistration du Château, vous partagez les photos su r lesquelles se trouve(nt) :

	Toutes les photos	Certaines photos	Aucune photo
votre famille présente lors de la visite			
vos amis présents lors de la visite			
vous-même			
d'autres personnes			

g) Avec les prochains visiteurs du Château, vous pa rtagez les photos sur lesquelles se trouve(nt) :

	Toutes les photos	Certaines photos	Aucune photo
votre famille présente lors de la visite			
vos amis présents lors de la visite			
vous-même			
d'autres personnes			

h) Lors de votre visite au château, lesquelles auri ez vous accepté de partager sur place avec d'autres visiteurs présents ? Celles sur lesquelles se trouvent :

	Toutes les photos	Certaines photos	Aucune photo
votre famille présente lors de la visite			
vos amis présents lors de la visite			
vous même			
d'autres personnes			

i) Avec quels autres groupes souhaiteriez vous part ager ces photos et pourquoi ?

Étape 2/5: Regroupement de photos pour le partage

Imaginons maintenant que vous avez pris beaucoup de photos avec votre téléphone portable lors de vos v oyages, de vos loisirs, etc. Ces photos sont associées à des informations, comme la date, le lieu, etc. Quelles sont les infor mations qui vous intéressent le plus pour regrouper vos photos ?

2.1) Regrouper mes photos par ...

	Très intéressant	Plutôt intéressant	Plutôt pas intéressant	Pas du tout intéressant	Sans préférence
lieux géographiques (Brésil, Paris, la Tour Eiffel)					
date de prise de vue					
saison (été, hiver, etc)					
période définie par vous-même (par exemple, du 1/12/2005 au 2/02/2006)					
activité (sport, visites, restaurant)					
groupes ou noms des personnes présentes (famille, m es amis, etc)					
événement (mariage, anniversaire, etc)					
selon des thématiques (fleurs, oiseaux, animaux, me r, montagne, etc)					

Étape 3/5: Partager vos photos

Pour les questions suivantes, indiquez si vous souh aitez partager des photos prises dans certaines occ asions et avec qui.

3.1) Pour des photos prises dans un lieu (Brésil, P aris, la Tour Eiffel), je suis prêt(e) à les partager avec :

Figure B.2: On-line survey: contextual annotation.

	Toutes les photos	Certaines photos	Aucune photo
Membres de la famille présents sur la photo			
Autres membres de la famille			
Amis présents sur la photo			
Autres amis			
Choix de personnes une par une			
Autres visiteurs du même lieu			
tout le monde			

3.2) Pour des photos prises à une date ou sur une période de temps, je suis prêt(e) à les partager avec :

	Toutes les photos	Certaines photos	Aucune photo
Membres de la famille présents sur la photo			
Autres membres de la famille			
Amis présents sur la photo			
Autres amis			
Choix de personnes une par une			
Autres visiteurs du même lieu			
Tout le monde			

3.3) Pour des photos prises lors d'un événement (Noël, □, je suis prêt(e) à les partager avec :

	Toutes les photos	Certaines photos	Aucune photo
Membres de la famille présents sur la photo			
Autres membres de la famille			
Amis présents sur la photo			
Autres amis			
Choix de personnes une par une			
Autres visiteurs du même lieu			
Tout le monde			

3.4) Pour des photos qui évoquent une thématique, je suis prêt(e) à les partager avec :

	Toutes les photos	Certaines photos	Aucune photo
Membres de la famille présents sur la photo			
Autres membres de la famille			
Amis présents sur la photo			
Autres amis			
Choix de personnes une par une			
Autres visiteurs du même lieu			
Tout le monde			

Étape 4.a/5: Recommander vos photos

Imaginez que vous êtes devant la Tour Eiffel le 31 décembre. Que répondez-vous si votre portable vous envoie les messages suivants (les propositions suivantes sont gratuites , par exemple si vous acceptez de recevoir une photo d'un visiteur cela ne vous coûte rien) :

4.a.1) Souhaitezvous voir des photos de la Tour Eiffel prises par des visiteurs précédents sous différents angles de vue ?

Oui, tout à fait — Non, plutôt pas — Sans avis
Oui, plutôt — Non, pas du tout

4.a.2) Souhaitezvous voir des photos de la Tour Eiffel prises par d'autres visiteurs en même temps que vous sous différents angles de vue ?

Oui, tout à fait — Non, plutôt pas — Sans Avis
Oui, plutôt — Non, pas du tout

4.a.3) Accepteriez vous de partager les photos que vous êtes en train de faire, avec les futurs visiteurs de la Tour Eiffel ?

Oui, tout à fait — Non, plutôt pas — Sans avis
Oui, plutôt — Non, pas du tout

4.a.4) Des commentaires et des suggestions sont disponibles sur ce lieu, souhaitezvous les consulter ?

Oui, tout à fait — Non, plutôt pas — Sans avis
Oui, plutôt — Non, pas du tout

Figure B.3: On-line survey: contextual annotation.

4.a.5) D'autres photos ont été prises, ici même, par un de vos amis. Souhaitezvous les voir ?

Oui, tout à fait | Non, plutôt pas | Sans avis
Oui, plutôt | Non, pas du tout

4.a.6) Accepteriez vous de partager les photos que vous êtes en train de faire, avec les personnes visitant la Tour Eiffel en même temps que vous ?

Oui, tout à fait | Non, plutôt pas | Sans avis
Oui, plutôt | Non, pas du tout

4.b/5: Recommander vos photos

Imaginez maintenant que vous venez d'arriver dans une ville différente de celle que vous habitez. Que répondez-vous si votre portable vous envoie les messages suivants (les propositions suivantes sont gratuites, par exemple si vous acceptez de recevoir une photo d'un visiteur cela ne vous coûte rien) :

4.b.1) Souhaitezvous voir des photos de la ville prises par d'autres visiteurs ?

Oui, tout à fait | Non, plutôt pas | Sans avis
Oui, plutôt | Non, pas du tout

4.b.2) Des commentaires et des suggestions sont disponibles sur cette ville, souhaitezvous les consulter ?

Oui, tout à fait | Non, plutôt pas | Sans avis
Oui, plutôt | Non, pas du tout

4.b.3) D'autres photos ont été prises, dans cette ville, pas un de vos amis. Souhaitezvous les voir ?

Oui, tout à fait | Non, plutôt pas | Sans avis
Oui, plutôt | Non, pas du tout

4.b.4) Accepteriez vous de partager celles que vous êtes en train de faire, avec les personnes visitant la ville en même temps que vous ?

Oui, tout à fait | Non, plutôt pas | Sans avis
Oui, plutôt | Non, pas du tout

4.b.5) Accepteriez vous de partager celles que vous êtes en train de faire, avec les futurs visiteurs de la ville ?

Oui, tout à fait | Non, plutôt pas | Sans avis
Oui, plutôt | Non, pas du tout

Étape 5/5: Types de documents

Selon les types de document (vidéo, audio, ...), quelles sont les opérations qui vous paraissent intéressantes de faire avec votre téléphone portable :

1) Vidéo

créer | visualiser | envoyer
partager | modifier

2) Photo

créer | visualiser | modifier
partager | envoyer

3) Audio

créer | écouter | envoyer
partager | modifier

4) Texte

créer | visualiser | envoyer
partager | modifier

5) Commentaire/recommandation

créer | visualiser | envoyer
partager | modifier

Figure B.4: On-line survey: contextual annotation.

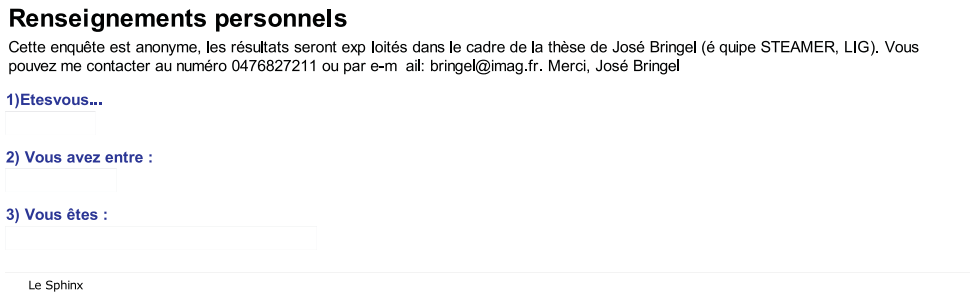


Figure B.5: On-line survey: contextual annotation.

Appendix C

Frequency tables of survey questions

1.1.a.1) Avec vos amis				
V1	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Jamais	33	16.50	33	16.50
Jusqu'à 3 fois par semaine	22	11.00	55	27.50
Plus de 3 fois par semaine	5	2.50	60	30.00
Rarement	140	70.00	200	100.00

1.1.a.2) Avec votre famille				
V2	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Jamais	52	26.13	52	26.13
Jusqu'à 3 fois par semaine	9	4.52	61	30.65
Plus de 3 fois par semaine	1	0.50	62	31.16
Rarement	137	68.84	199	100.00

1.1.a.3) Durant des événements				
V3	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Jamais	40	20.51	40	20.51
Jusqu'à 3 fois par semaine	34	17.44	74	37.95
Plus de 3 fois par semaine	10	5.13	84	43.08
Rarement	111	56.92	195	100.00

1.1.a.4) Lors d'activités				
V4	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Jamais	57	28.93	57	28.93
Jusqu'à 3 fois par semaine	25	12.69	82	41.62
Plus de 3 fois par semaine	3	1.52	85	43.15
Rarement	112	56.85	197	100.00

Figure C.1: Question 1.1

1.2.a.1) votre famille présente				
V6	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	12	6.09	12	6.09
Certaines photos	39	19.80	51	25.89
Toutes les photos	146	74.11	197	100.00

1.2.a.2) vos amis présents à la visite				
V7	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	27	13.85	27	13.85
Certaines photos	95	48.72	122	62.56
Toutes les photos	73	37.44	195	100.00

1.2.a.3) vous même				
V8	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	20	10.15	20	10.15
Certaines photos	103	52.28	123	62.44
Toutes les photos	74	37.56	197	100.00

1.2.a.4) d'autres personnes				
V9	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	63	31.98	63	31.98
Certaines photos	84	42.64	147	74.62
Toutes les photos	50	25.38	197	100.00

Figure C.2: Question 1.2a

1.2.b.1) votre famille présente				
V10	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	16	8.04	16	8.04
Certaines photos	85	42.71	101	50.75
Toutes les photos	98	49.25	199	100.00

1.2.b.2) vos amis présents à la visite				
V11	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	64	32.49	64	32.49
Certaines photos	108	54.82	172	87.31
Toutes les photos	25	12.69	197	100.00

1.2.b.3) vous même				
V12	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	24	12.18	24	12.18
Certaines photos	130	65.99	154	78.17
Toutes les photos	43	21.83	197	100.00

1.2.b.4) d'autres personnes				
V13	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	97	49.49	97	49.49
Certaines photos	74	37.76	171	87.24
Toutes les photos	25	12.76	196	100.00

Figure C.3: Question 1.2b

1.2.c.1) votre famille présente				
V14	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	52	26.00	52	26.00
Certaines photos	106	53.00	158	79.00
Toutes les photos	42	21.00	200	100.00

1.2.c.2) vos amis présents à la visite				
V15	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	8	4.02	8	4.02
Certaines photos	50	25.13	58	29.15
Toutes les photos	141	70.85	199	100.00

1.2.c.3) vous même				
V16	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	20	10.05	20	10.05
Certaines photos	107	53.77	127	63.82
Toutes les photos	72	36.18	199	100.00

1.2.c.4) d'autres personnes				
V17	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	69	34.67	69	34.67
Certaines photos	77	38.69	146	73.37
Toutes les photos	53	26.63	199	100.00

Figure C.4: Question 1.2c

1.2.d.1) votre famille présente				
V18	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	87	44.39	87	44.39
Certaines photos	95	48.47	182	92.86
Toutes les photos	14	7.14	196	100.00

1.2.d.2) vos amis présents à la visite				
V19	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	27	13.71	27	13.71
Certaines photos	119	60.41	146	74.11
Toutes les photos	51	25.89	197	100.00

1.2.d.3) vous même				
V20	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	30	15.23	30	15.23
Certaines photos	127	64.47	157	79.70
Toutes les photos	40	20.30	197	100.00

1.2.d.4) d'autres personnes				
V21	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	83	42.78	83	42.78
Certaines photos	85	43.81	168	86.60
Toutes les photos	26	13.40	194	100.00

Figure C.5: Question 1.2d

1.2.e.1) votre famille présente				
V22	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	99	50.51	99	50.51
Certaines photos	88	44.90	187	95.41
Toutes les photos	9	4.59	196	100.00

1.2.e.2) vos amis présents				
V23	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	73	37.06	73	37.06
Certaines photos	107	54.31	180	91.37
Toutes les photos	17	8.63	197	100.00

1.2.e.3) vous même				
V24	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	75	38.07	75	38.07
Certaines photos	108	54.82	183	92.89
Toutes les photos	14	7.11	197	100.00

1.2.e.4) d'autres personnes				
V25	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	110	56.70	110	56.70
Certaines photos	71	36.60	181	93.30
Toutes les photos	13	6.70	194	100.00

Figure C.6: Question 1.2e

1.2.f.1) votre famille présente				
V26	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	174	87.88	174	87.88
Certaines photos	22	11.11	196	98.99
Toutes les photos	2	1.01	198	100.00

1.2.f.2) vos amis présents à la visite				
V27	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	172	86.43	172	86.43
Certaines photos	26	13.07	198	99.50
Toutes les photos	1	0.50	199	100.00

1.2.f.3) vous même				
V28	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	170	85.43	170	85.43
Certaines photos	28	14.07	198	99.50
Toutes les photos	1	0.50	199	100.00

1.2.f.4) d'autres personnes				
V29	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	147	73.50	147	73.50
Certaines photos	42	21.00	189	94.50
Toutes les photos	11	5.50	200	100.00

Figure C.7: Question 1.2f

1.2.g.1) votre famille présente				
V30	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	175	88.38	175	88.38
Certaines photos	23	11.62	198	100.00

1.2.g.2) vos amis présents à la visite				
V31	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	171	86.36	171	86.36
Certaines photos	27	13.64	198	100.00

1.2.g.3) vous même				
V32	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	172	86.43	172	86.43
Certaines photos	26	13.07	198	99.50
Toutes les photos	1	0.50	199	100.00

1.2.g.4) d'autres personnes				
V33	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	149	74.87	149	74.87
Certaines photos	43	21.61	192	96.48
Toutes les photos	7	3.52	199	100.00

Figure C.8: Question 1.2g

1.2.h.1) votre famille présente				
V34	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	144	75.00	144	75.00
Certaines photos	44	22.92	188	97.92
Toutes les photos	4	2.08	192	100.00

1.2.h.2) vos amis présents à la visite				
V35	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	134	69.79	134	69.79
Certaines photos	55	28.65	189	98.44
Toutes les photos	3	1.56	192	100.00

1.2.h.3) vous même				
V36	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	137	70.98	137	70.98
Certaines photos	55	28.50	192	99.48
Toutes les photos	1	0.52	193	100.00

1.2.h.4) d'autres personnes				
V37	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	105	54.97	105	54.97
Certaines photos	68	35.60	173	90.58
Toutes les photos	18	9.42	191	100.00

Figure C.9: Question 1.2h

2.1.1) lieux géographique				
V39	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Pas du tout intéressant	6	3.00	6	3.00
Plutôt intéressant	61	30.50	67	33.50
Plutôt pas intéressant	7	3.50	74	37.00
Sans préférence	3	1.50	77	38.50
Très intéressant	123	61.50	200	100.00

2.1.2) date				
V40	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Pas du tout intéressant	8	4.00	8	4.00
Plutôt intéressant	70	35.00	78	39.00
Plutôt pas intéressant	16	8.00	94	47.00
Sans préférence	11	5.50	105	52.50
Très intéressant	95	47.50	200	100.00

2.1.3) saison				
V41	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Pas du tout intéressant	46	23.12	46	23.12
Plutôt intéressant	47	23.62	93	46.73
Plutôt pas intéressant	60	30.15	153	76.88
Sans préférence	33	16.58	186	93.47
Très intéressant	13	6.53	199	100.00

2.1.4) période				
V42	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Pas du tout intéressant	30	15.23	30	15.23
Plutôt intéressant	73	37.06	103	52.28
Plutôt pas intéressant	36	18.27	139	70.56
Sans préférence	23	11.68	162	82.23
Très intéressant	35	17.77	197	100.00

Figure C.10: Question 2

2.1.5) activité				
V43	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Pas du tout intéressant	25	12.63	25	12.63
Plutôt intéressant	76	38.38	101	51.01
Plutôt pas intéressant	52	26.26	153	77.27
Sans préférence	18	9.09	171	86.36
Très intéressant	27	13.64	198	100.00

2.1.6) groupes ou noms				
V44	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Pas du tout intéressant	23	11.68	23	11.68
Plutôt intéressant	91	46.19	114	57.87
Plutôt pas intéressant	33	16.75	147	74.62
Sans préférence	11	5.58	158	80.20
Très intéressant	39	19.80	197	100.00

2.1.7) événements				
V45	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Pas du tout intéressant	1	0.50	1	0.50
Plutôt intéressant	91	45.50	92	46.00
Plutôt pas intéressant	12	6.00	104	52.00
Sans préférence	3	1.50	107	53.50
Très intéressant	93	46.50	200	100.00

2.1.8) thématique				
V46	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Pas du tout intéressant	62	31.31	62	31.31
Plutôt intéressant	50	25.25	112	56.57
Plutôt pas intéressant	50	25.25	162	81.82
Sans préférence	26	13.13	188	94.95
Très intéressant	10	5.05	198	100.00

Figure C.11: Question 2

3.1.1) membres famille				
V47	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	4	2.01	4	2.01
Certaines photos	45	22.61	49	24.62
Toutes les photos	150	75.38	199	100.00

3.1.2) autres membres famille				
V48	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	7	3.52	7	3.52
Certaines photos	125	62.81	132	66.33
Toutes les photos	67	33.67	199	100.00

3.1.3) amis photos				
V49	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	4	2.01	4	2.01
Certaines photos	52	26.13	56	28.14
Toutes les photos	143	71.86	199	100.00

3.1.4) autres amis				
V50	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	10	5.05	10	5.05
Certaines photos	148	74.75	158	79.80
Toutes les photos	40	20.20	198	100.00

Figure C.12: Question 3.1

3.1.5) choix par personne				
V51	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	17	8.72	17	8.72
Certaines photos	144	73.85	161	82.56
Toutes les photos	34	17.44	195	100.00

3.1.6) autres visiteurs				
V52	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	125	64.10	125	64.10
Certaines photos	61	31.28	186	95.38
Toutes les photos	9	4.62	195	100.00

3.1.7) Tout le monde				
V53	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	113	57.36	113	57.36
Certaines photos	81	41.12	194	98.48
Toutes les photos	3	1.52	197	100.00

Figure C.13: Question 3.1

3.2.1) membres famille				
V54	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	6	3.06	6	3.06
Certaines photos	62	31.63	68	34.69
Toutes les photos	128	65.31	196	100.00

3.2.2) autres membres famille				
V55	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	8	4.06	8	4.06
Certaines photos	141	71.57	149	75.63
Toutes les photos	48	24.37	197	100.00

3.2.3) amis présent photo				
V56	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	8	4.06	8	4.06
Certaines photos	65	32.99	73	37.06
Toutes les photos	124	62.94	197	100.00

3.2.4) autres amis				
V57	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	17	8.72	17	8.72
Certaines photos	149	76.41	166	85.13
Toutes les photos	29	14.87	195	100.00

Figure C.14: Question 3.2

3.2.5) choix par personne				
V58	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	22	11.34	22	11.34
Certaines photos	147	75.77	169	87.11
Toutes les photos	25	12.89	194	100.00

3.2.6) autres visiteurs				
V59	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	123	63.40	123	63.40
Certaines photos	65	33.51	188	96.91
Toutes les photos	6	3.09	194	100.00

3.2.7) tout le monde				
V60	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	115	59.28	115	59.28
Certaines photos	75	38.66	190	97.94
Toutes les photos	4	2.06	194	100.00

Figure C.15: Question 3.2

3.3.1) membres famille				
V61	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	4	2.05	4	2.05
Certaines photos	45	23.08	49	25.13
Toutes les photos	146	74.87	195	100.00

3.3.2) autres membres famille				
V62	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	9	4.59	9	4.59
Certaines photos	111	56.63	120	61.22
Toutes les photos	76	38.78	196	100.00

3.3.3) amis photos				
V63	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	6	3.06	6	3.06
Certaines photos	60	30.61	66	33.67
Toutes les photos	130	66.33	196	100.00

3.3.4) autres amis				
V64	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	24	12.31	24	12.31
Certaines photos	143	73.33	167	85.64
Toutes les photos	28	14.36	195	100.00

Figure C.16: Question 3.3

3.3.5) choix par personne				
V65	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	29	15.03	29	15.03
Certaines photos	138	71.50	167	86.53
Toutes les photos	26	13.47	193	100.00

3.3.6) autres visiteurs				
V66	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	129	66.84	129	66.84
Certaines photos	57	29.53	186	96.37
Toutes les photos	7	3.63	193	100.00

3.3.7) Tout le monde				
V67	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	125	64.43	125	64.43
Certaines photos	66	34.02	191	98.45
Toutes les photos	3	1.55	194	100.00

Figure C.17: Question 3.3

3.4.1) membres famille				
V68	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	12	6.09	12	6.09
Certaines photos	59	29.95	71	36.04
Toutes les photos	126	63.96	197	100.00

3.4.2) autres membres famille				
V69	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	16	8.12	16	8.12
Certaines photos	127	64.47	143	72.59
Toutes les photos	54	27.41	197	100.00

3.4.3) amirs photo				
V70	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	11	5.61	11	5.61
Certaines photos	64	32.65	75	38.27
Toutes les photos	121	61.73	196	100.00

3.4.4) Autres amis				
V71	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	24	12.37	24	12.37
Certaines photos	129	66.49	153	78.87
Toutes les photos	41	21.13	194	100.00

Figure C.18: Question 3.4

3.4.5) Choix par personne				
V72	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	32	16.58	32	16.58
Certaines photos	131	67.88	163	84.46
Toutes les photos	30	15.54	193	100.00

3.4.6) Autres visiteurs				
V73	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	114	58.46	114	58.46
Certaines photos	71	36.41	185	94.87
Toutes les photos	10	5.13	195	100.00

3.4.7) Tout le monde				
V74	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Aucune photo	107	55.44	107	55.44
Certaines photos	78	40.41	185	95.85
Toutes les photos	8	4.15	193	100.00

Figure C.19: Question 3.4

4.a.1) photos visiteurs précédents				
V75	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	39	29.55	39	29.55
Non, plutôt pas	29	21.97	68	51.52
Oui, plutôt	42	31.82	110	83.33
Oui, tout à fait	18	13.64	128	96.97
Sans avis	4	3.03	132	100.00

4.a.2) photo TE visiteurs même temps				
V76	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	35	26.72	35	26.72
Non, plutôt pas	27	20.61	62	47.33
Oui, plutôt	35	26.72	97	74.05
Oui, tout à fait	21	16.03	118	90.08
Sans Avis	13	9.92	131	100.00

4.a.3) photo TE futurs				
V77	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	51	38.93	51	38.93
Non, plutôt pas	36	27.48	87	66.41
Oui, plutôt	31	23.66	118	90.08
Oui, tout à fait	7	5.34	125	95.42
Sans avis	6	4.58	131	100.00

Figure C.20: Question 4.a

4.a.4) com/sug				
V78	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	17	8.63	17	8.63
Non, plutôt pas	39	19.80	56	28.43
Oui, plutôt	107	54.31	163	82.74
Oui, tout à fait	24	12.18	187	94.92
Sans avis	10	5.08	197	100.00

4.a.5) autres photos				
V79	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	11	5.50	11	5.50
Non, plutôt pas	15	7.50	26	13.00
Oui, plutôt	97	48.50	123	61.50
Oui, tout à fait	70	35.00	193	96.50
Sans avis	7	3.50	200	100.00

4.a.6) partager avec visiteurs				
V80	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	85	42.50	85	42.50
Non, plutôt pas	65	32.50	150	75.00
Oui, plutôt	42	21.00	192	96.00
Oui, tout à fait	4	2.00	196	98.00
Sans avis	4	2.00	200	100.00

Figure C.21: Question 4.a

4.b.1) voir photos autres personnes				
V81	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	44	22.22	44	22.22
Non, plutôt pas	38	19.19	82	41.41
Oui, plutôt	76	38.38	158	79.80
Oui, tout à fait	30	15.15	188	94.95
Sans avis	10	5.05	198	100.00

4.b.2) com/sug ville				
V82	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	7	3.55	7	3.55
Non, plutôt pas	19	9.64	26	13.20
Oui, plutôt	102	51.78	128	64.97
Oui, tout à fait	61	30.96	189	95.94
Sans avis	8	4.06	197	100.00

4.b.3) photos prises dans ville (amis)				
V83	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	9	4.52	9	4.52
Non, plutôt pas	17	8.54	26	13.07
Oui, plutôt	84	42.21	110	55.28
Oui, tout à fait	84	42.21	194	97.49
Sans avis	5	2.51	199	100.00

Figure C.22: Question 4.b

4.b.4) partager avec autres visiteurs				
V84	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	62	31.16	62	31.16
Non, plutôt pas	63	31.66	125	62.81
Oui, plutôt	53	26.63	178	89.45
Oui, tout à fait	10	5.03	188	94.47
Sans avis	11	5.53	199	100.00

4.b.5) partage futurs visiteurs				
V85	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Non, pas du tout	49	37.40	49	37.40
Non, plutôt pas	37	28.24	86	65.65
Oui, plutôt	33	25.19	119	90.84
Oui, tout à fait	6	4.58	125	95.42
Sans avis	6	4.58	131	100.00

Figure C.23: Question 4.b

5.1) vidéo opération				
V86	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
créer	7	3.59	7	3.59
créer ; envoyer	4	2.05	11	5.64
créer ; envoyer ; modifier	1	0.51	12	6.15
créer ; modifier	2	1.03	14	7.18
créer ; partager	3	1.54	17	8.72
créer ; partager ; visualiser	9	4.62	26	13.33
créer ; partager ; visualiser ; envoyer	16	8.21	42	21.54
créer ; visualiser	46	23.59	88	45.13
créer ; visualiser ; envoyer	12	6.15	100	51.28
créer ; visualiser ; envoyer ; modifier	3	1.54	103	52.82
créer ; visualiser ; envoyer ; partager	20	10.26	123	63.08
créer ; visualiser ; envoyer ; partager ; modifier	19	9.74	142	72.82
créer ; visualiser ; modifier	1	0.51	143	73.33
créer ; visualiser ; partager	17	8.72	160	82.05
envoyer ; créer ; partager ; visualiser	7	3.59	167	85.64
envoyer ; partager	1	0.51	168	86.15
envoyer ; partager ; visualiser ; créer	1	0.51	169	86.67
partager	1	0.51	170	87.18
partager ; créer ; visualiser	1	0.51	171	87.69
visualiser	18	9.23	189	96.92
visualiser ; créer ; partager	1	0.51	190	97.44
visualiser ; envoyer	2	1.03	192	98.46
visualiser ; envoyer ; créer ; partager	1	0.51	193	98.97
visualiser ; partager	2	1.03	195	100.00

Figure C.24: Question 5.1

5.2) photo opérations				
V87	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
créer	3	1.52	3	1.52
créer ; envoyer	3	1.52	6	3.03
créer ; modifier	2	1.01	8	4.04
créer ; modifier ; envoyer	1	0.51	9	4.55
créer ; modifier ; partager	1	0.51	10	5.05
créer ; modifier ; partager ; envoyer	1	0.51	11	5.56
créer ; partager	2	1.01	13	6.57
créer ; partager ; envoyer	3	1.52	16	8.08
créer ; partager ; visualiser	11	5.56	27	13.64
créer ; partager ; visualiser ; envoyer	32	16.16	59	29.80
créer ; visualiser	20	10.10	79	39.90
créer ; visualiser ; envoyer	20	10.10	99	50.00
créer ; visualiser ; modifier	1	0.51	100	50.51
créer ; visualiser ; modifier ; envoyer	6	3.03	106	53.54
créer ; visualiser ; modifier ; partager	3	1.52	109	55.05
créer ; visualiser ; modifier ; partager ; envoyer	35	17.68	144	72.73
créer ; visualiser ; partager	8	4.04	152	76.77
créer ; visualiser ; partager ; envoyer	23	11.62	175	88.38
partager ; créer	1	0.51	176	88.89
visualiser	12	6.06	188	94.95
visualiser ; envoyer	1	0.51	189	95.45
visualiser ; partager	3	1.52	192	96.97
visualiser ; partager ; créer	1	0.51	193	97.47
visualiser ; partager ; envoyer	5	2.53	198	100.00

Figure C.25: Question 5.2

5.3) audio opérations				
V88	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
créer	4	2.06	4	2.06
créer ; envoyer	1	0.52	5	2.58
créer ; partager ; écouter	5	2.58	10	5.15
créer ; partager ; écouter ; envoyer	20	10.31	30	15.46
créer ; écouter	30	15.46	60	30.93
créer ; écouter ; envoyer	15	7.73	75	38.66
créer ; écouter ; envoyer ; modifier	1	0.52	76	39.18
créer ; écouter ; envoyer ; partager	17	8.76	93	47.94
créer ; écouter ; envoyer ; partager ; modifier	11	5.67	104	53.61
créer ; écouter ; modifier	3	1.55	107	55.15
créer ; écouter ; partager	6	3.09	113	58.25
créer ; écouter ; partager ; modifier	1	0.52	114	58.76
écouter	45	23.20	159	81.96
écouter ; envoyer	6	3.09	165	85.05
écouter ; envoyer ; partager	15	7.73	180	92.78
écouter ; modifier	1	0.52	181	93.30
écouter ; partager	12	6.19	193	99.48
écouter ; partager ; modifier	1	0.52	194	100.00

Figure C.26: Question 5.3

5.4) texte opérations				
V89	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
créer	2	1.04	2	1.04
créer ; envoyer	16	8.33	18	9.38
créer ; envoyer ; modifier	3	1.56	21	10.94
créer ; envoyer ; partager	3	1.56	24	12.50
créer ; envoyer ; partager ; modifier	1	0.52	25	13.02
créer ; modifier	1	0.52	26	13.54
créer ; partager	9	4.69	35	18.23
créer ; partager ; modifier	1	0.52	36	18.75
créer ; partager ; visualiser	7	3.65	43	22.40
créer ; partager ; visualiser ; envoyer	28	14.58	71	36.98
créer ; visualiser	18	9.38	89	46.35
créer ; visualiser ; envoyer	10	5.21	99	51.56
créer ; visualiser ; envoyer ; modifier	23	11.98	122	63.54
créer ; visualiser ; envoyer ; partager	7	3.65	129	67.19
créer ; visualiser ; envoyer ; partager ; modifier	38	19.79	167	86.98
créer ; visualiser ; modifier	5	2.60	172	89.58
créer ; visualiser ; partager	2	1.04	174	90.63
créer ; visualiser ; partager ; modifier	2	1.04	176	91.67
envoyer	2	1.04	178	92.71
partager	1	0.52	179	93.23
partager ; créer	2	1.04	181	94.27
visualiser	7	3.65	188	97.92
visualiser ; partager	1	0.52	189	98.44
visualiser ; partager ; créer	2	1.04	191	99.48
visualiser ; partager ; modifier	1	0.52	192	100.00

Figure C.27: Question 5.4

5.5) com/recom opérations				
V90	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
créer	10	5.56	10	5.56
créer ; envoyer	4	2.22	14	7.78
créer ; envoyer ; modifier	1	0.56	15	8.33
créer ; envoyer ; partager	4	2.22	19	10.56
créer ; modifier	3	1.67	22	12.22
créer ; partager	4	2.22	26	14.44
créer ; partager ; modifier	1	0.56	27	15.00
créer ; partager ; modifier ; envoyer	1	0.56	28	15.56
créer ; partager ; visualiser	5	2.78	33	18.33
créer ; partager ; visualiser ; envoyer	27	15.00	60	33.33
créer ; visualiser	11	6.11	71	39.44
créer ; visualiser ; envoyer	3	1.67	74	41.11
créer ; visualiser ; envoyer ; modifier	5	2.78	79	43.89
créer ; visualiser ; envoyer ; partager	9	5.00	88	48.89
créer ; visualiser ; envoyer ; partager ; modifier	27	15.00	115	63.89
créer ; visualiser ; modifier	3	1.67	118	65.56
créer ; visualiser ; partager	4	2.22	122	67.78
créer ; visualiser ; partager ; envoyer	1	0.56	123	68.33
créer ; visualiser ; partager ; modifier	4	2.22	127	70.56
envoyer ; partager	1	0.56	128	71.11
partager	4	2.22	132	73.33
partager ; créer	4	2.22	136	75.56
visualiser	34	18.89	170	94.44
visualiser ; envoyer	1	0.56	171	95.00
visualiser ; envoyer ; partager	5	2.78	176	97.78
visualiser ; modifier	1	0.56	177	98.33
visualiser ; partager	1	0.56	178	98.89
visualiser ; partager ; créer	1	0.56	179	99.44
visualiser ; partager ; modifier	1	0.56	180	100.00

Figure C.28: Question 5.5

5.5) com/recom opérations				
V90	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
créer	10	5.56	10	5.56
créer ; envoyer	4	2.22	14	7.78
créer ; envoyer ; modifier	1	0.56	15	8.33
créer ; envoyer ; partager	4	2.22	19	10.56
créer ; modifier	3	1.67	22	12.22
créer ; partager	4	2.22	26	14.44
créer ; partager ; modifier	1	0.56	27	15.00
créer ; partager ; modifier ; envoyer	1	0.56	28	15.56
créer ; partager ; visualiser	5	2.78	33	18.33
créer ; partager ; visualiser ; envoyer	27	15.00	60	33.33
créer ; visualiser	11	6.11	71	39.44
créer ; visualiser ; envoyer	3	1.67	74	41.11
créer ; visualiser ; envoyer ; modifier	5	2.78	79	43.89
créer ; visualiser ; envoyer ; partager	9	5.00	88	48.89
créer ; visualiser ; envoyer ; partager ; modifier	27	15.00	115	63.89
créer ; visualiser ; modifier	3	1.67	118	65.56
créer ; visualiser ; partager	4	2.22	122	67.78
créer ; visualiser ; partager ; envoyer	1	0.56	123	68.33
créer ; visualiser ; partager ; modifier	4	2.22	127	70.56
envoyer ; partager	1	0.56	128	71.11
partager	4	2.22	132	73.33
partager ; créer	4	2.22	136	75.56
visualiser	34	18.89	170	94.44
visualiser ; envoyer	1	0.56	171	95.00
visualiser ; envoyer ; partager	5	2.78	176	97.78
visualiser ; modifier	1	0.56	177	98.33
visualiser ; partager	1	0.56	178	98.89
visualiser ; partager ; créer	1	0.56	179	99.44
visualiser ; partager ; modifier	1	0.56	180	100.00

Figure C.29: Question 5.5

Etes-vous...				
V91	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Un homme	69	34.50	69	34.50
Une femme	131	65.50	200	100.00

Vous avez entre				
V92	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
18-25 ans	170	85.00	170	85.00
26-35 ans	26	13.00	196	98.00
35-45 ans	2	1.00	198	99.00
46-55 ans	1	0.50	199	99.50
plus de 55 ans	1	0.50	200	100.00

Vous êtes				
V93	Fréquence	Pourcentage	Fréquence cumulée	Pourcent. cumulé
Autre activité	3	1.50	3	1.50
Cadre et profession intellectuelle supérieure	15	7.50	18	9.00
Employé	2	1.00	20	10.00
Etudiant, Lycéen	179	89.50	199	99.50
Profession intermédiaire	1	0.50	200	100.00

Figure C.30: Personal information

Bibliography

[802.11 2009] IEEE 802.11. *The Working Group for WLAN Standards*. http://grouper.ieee.org/groups/802/11/, 2009. 3

[802.15 2009] IEEE 802.15. *The Working Group for WPAN*. http://ieee802.org/15/index.html, 2009. 3

[802.16 2009] IEEE 802.16. *The Working Group on Broadband Wireless Access Standards*. http://grouper.ieee.org/groups/802/16/, 2009. 3

[Abid 2009] Zied Abid, Sophie Chabridon and Denis Conan. *A framework for quality of context management*. In QuaCon'09: Proceedings of the 1st international conference on Quality of context, pages 120–131, Berlin, Heidelberg, 2009. Springer-Verlag. iv, v, 95, 96, 97, 110

[Aich 2007] Subhendu Aich, Shamik Sural and Arun K. Majumdar. *STARBAC: Spatiotemporal Role Based Access Control*. In OTM Conferences (2), pages 1567–1582, 2007. 45, 112

[Aich 2009] Subhendu Aich, Samrat Mondal, Shamik Sural and Arun Kumar Majumdar. *Role Based Access Control with Spatiotemporal Context for Mobile Applications*. Transactions on Computational Science IV: Special Issue on Security in Computing, pages 177–199, 2009. 45

[Anderson 1972] James P. Anderson. *Computer Security Technology Planning Study*. Rapport technique, Hanscom AFB, Bedford, MA 01731, October 1972. 16

[Bailey 2002] James Bailey, Alexandra Poulovassilis and Peter T. Wood. *An event-condition-action language for XML*. In WWW '02, pages 486–495, New York, NY, USA, 2002. ACM. 204

[Baldauf 2007] Matthias Baldauf, Schahram Dustdar and Florian Rosenberg. *A survey on context-aware systems*. Int. J. Ad Hoc Ubiquitous Comput., vol. 2, no. 4, pages 263–277, 2007.

[Barker 2009] Steve Barker. *The next 700 access control models or a unifying meta-model?* In SACMAT '09: Proceedings of the 14th ACM symposium on Access control models and technologies, pages 187–196, New York, NY, USA, 2009. ACM.

[Bell 1973] David E. Bell and Leonard J. LaPadula. *Secure computer systems: Mathematical foundations*. Rapport technique, Electronic Systems Division, November 1973. 21, 23, 48

[Bell 1974] David E. Bell. *Secure computer systems: A refinement of the mathematical model.* Rapport technique, Electronic Systems Division, April 1974. 21

[Bell 1976] David E. Bell and Leonard J. LaPadula. *Secure computer system: Unified exposition and MULTICS interpretation.* Rapport technique, The MITRE Corporation, March 1976. 21

[Bellavista 2003] Paolo Bellavista, Antonio Corradi, Rebecca Montanari and Cesare Stefanelli. *Context-Aware Middleware for Resource Management in the Wireless Internet.* IEEE Trans. Software Eng., vol. 29, no. 12, pages 1086–1099, 2003.

[Bertino 2001] Elisa Bertino, Piero Andrea Bonatti and Elena Ferrari. *TRBAC: A temporal role-based access control model.* ACM Trans. Inf. Syst. Secur., vol. 4, no. 3, pages 191–233, 2001. iii, 36, 37, 45, 48, 112

[Bertino 2005] Elisa Bertino, Barbara Catania, Maria Luisa Damiani and Paolo Perlasca. *GEO-RBAC: A spatially aware RBAC.* In SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies, pages 29–37, New York, NY, USA, 2005. ACM. iii, 39, 41, 48, 112

[Biba 1977] K. J. Biba. *Integrity Considerations for Secure Computer Systems.* Rapport technique, MITRE Corp., 04 1977. 21, 23, 48

[Brewer 1989] Dr. David F.C. Brewer and Dr. Micheal J. Nash. *The Chinese Wall Security Policy.* Security and Privacy, IEEE Symposium on, vol. 0, page 206, 1989. 21, 24, 48

[Brucker 2009] Achim D. Brucker and Helmut Petritsch. *Extending access control models with break-glass.* In SACMAT '09: Proceedings of the 14th ACM symposium on Access control models and technologies, pages 197–206, New York, NY, USA, 2009. ACM.

[Bu 2006] Yingyi Bu, Tao Gu, Xianping Tao, Jun Li, Shaxun Chen and Jian Lu. *Managing Quality of Context in Pervasive Computing.* In QSIC '06: Proceedings of the Sixth International Conference on Quality Software, pages 193–200, Washington, DC, USA, 2006. IEEE Computer Society. iv, 94, 95, 99, 108, 110

[Buchholz 2003] Thomas Buchholz, Axel Küpper and Michael Schiffers. *Quality of Context: What It Is And Why We Need It.* In (HPOVUA 2003), Geneva, 2003, 2003. iv, 80, 82, 83, 84, 85, 86, 92, 97, 99, 100, 108, 113, 176, 177

[Buchholz 2004] Thomas Buchholz, Michael Krause, Claudia Linnhoff-Popien and Michael Schiffers. *CoCo: Dynamic Composition of Context Information.* Mobile and Ubiquitous Systems, Annual International Conference on, vol. 0, pages 335–343, 2004. 89

[Chae 2006] Song-hwa Chae, Wonil Kim and Dong-kyoo Kim. *Role-Based Access Control Model for Ubiquitous Computing Environment.* In Joo-Seok Song, Taekyoung Kwon and Moti Yung, editeurs, Information Security Applications, volume 3786, pages 354–363, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. iv, 64, 65, 77

[Chandran 2005] Suroop Mohan Chandran and James B. D. Joshi. *LoT-RBAC: A Location and Time-Based RBAC Model.* In Anne H. H. Ngu, Masaru Kitsuregawa, Erich J. Neuhold, Jen-Yao Chung and Quan Z. Sheng, editeurs, WISE, volume 3806 of *Lecture Notes in Computer Science*, pages 361–375. Springer, 2005. iii, 43, 112

[Chen 2008] Liang Chen and Jason Crampton. *On spatio-temporal constraints and inheritance in role-based access control.* In ASIACCS '08: Proceedings of the 2008 ACM symposium on Information, computer and communications security, pages 205–216, New York, NY, USA, 2008. ACM.

[Cholewka 2000] Damian G. Cholewka, Reinhardt A. Botha and Jan H. P. Eloff. *A Context-Sensitive Access Control Model and Prototype Implementation.* In Proceedings of the IFIP TC11 Fifteenth Annual Working Conference on Information Security for Global Information Infrastructures, pages 341–350, Deventer, The Netherlands, The Netherlands, 2000. Kluwer, B.V.

[Conan 2007] Denis Conan, Romain Rouvoy and Lionel Seinturier. *Scalable processing of context information with COSMOS.* In DAIS'07: Proceedings of the 7th IFIP WG 6.1 international conference on Distributed applications and interoperable systems, pages 210–224, Berlin, Heidelberg, 2007. Springer-Verlag. 95

[Conferences 2005] IARIA Conferences, editeur. Telecommunications 2005: Advanced industrial conference on telecommunications / service assurance with partial and intermittent resources conference / e-learning on telecommunications workshop (aict / sapir / elete 2005), 17-22 july 2005, lisbon, portugal. IEEE Computer Society, 2005. 275

[Consortium 1999] OpenGIS Consortium. *OpenGIS Simple Features Specification for SQL. Technical Report OGC 99-049.* Rapport technique, Open GeoSpatial Consortium, 1999. 40

[Corradi 2004a] Antonio Corradi, Rebecca Montanari and Daniela Tibaldi. *Context-Based Access Control for Ubiquitous Service Provisioning.* In COMPSAC '04: Proceedings of the 28th Annual International Computer Software and Applications Conference, pages 444–451, Washington, DC, USA, 2004. IEEE Computer Society. iv, 4, 70, 71, 77, 112

[Corradi 2004b] Antonio Corradi, Rebecca Montanari and Daniela Tibaldi. *Context-Based Access Control Management in Ubiquitous Environments.* Network

Computing and Applications, IEEE International Symposium on, vol. 0, pages 253–260, 2004. iv, 70, 71, 77, 112

[Covington 2000] Michael J. Covington, Matthew J. Moyer and Mustaque Ahamad. *Generalized Role-Based Access Control for Securing Future Applications.* In In Proceedings of the National Information Systems Security Conference (NISSC), 2000. iii, 46, 48, 49, 52, 53, 112

[Covington 2001] Michael Covington, Wende Long, Srividhya Srinivasan, Anind Dev, Mustaque Ahamad and Gregory Abowd. *Securing context-aware applications using environment roles.* In SACMAT '01: Proceedings of the sixth ACM symposium on Access control models and technologies, pages 10–20, New York, NY, USA, 2001. ACM. iii, 3, 52, 54, 112

[Covington 2002] Michael J. Covington, Prahlad Fogla, Zhiyuan Zhan and Mustaque Ahamad. *A Context-Aware Security Architecture for Emerging Applications.* In ACSAC '02: Proceedings of the 18th Annual Computer Security Applications Conference, page 249, Washington, DC, USA, 2002. IEEE Computer Society.

[Covington 2006] Michael J. Covington and Manoj R. Sastry. *A Contextual Attribute-Based Access Control Model.* In Meersman et al. [Meersman 2006], pages 1996–2006. iv, 67, 68, 69, 77, 112

[Damiani 2007] Maria Luisa Damiani, Elisa Bertino, Barbara Catania and Paolo Perlasca. *GEO-RBAC: A spatially aware RBAC.* ACM Trans. Inf. Syst. Secur., vol. 10, no. 1, page 2, 2007. 39, 41, 48, 112

[Denning 1976] Dorothy E. Denning. *A lattice model of secure information flow.* Commun. ACM, vol. 19, no. 5, pages 236–243, 1976. 22

[Dersingh 2010] Anand Dersingh, Ramiro Liscano, Allan Jost, John Finnson and Rajiv Senthilnathan. *Utilizing Semantic Knowledge for Access Control in Pervasive and Ubiquitous Systems.* Mob. Netw. Appl., vol. 15, no. 2, pages 267–282, 2010.

[Dey 1999] Anind K. Dey and Gregory D. Abowd. *The Context Toolkit: Aiding the Development of Context-Aware Applications.* In Workshop on Software Engineering for Wearable and Pervasive Computing, pages 434–441. ACM Press, 1999. 55

[Dey 2001] Anind K. Dey. *Understanding and Using Context.* Personal Ubiquitous Computing, vol. 5, no. 1, pages 4–7, 2001. 4, 127, 128

[Emami 2007] Sareh Sadat Emami, Morteza Amini and Saadan Zokaei. *A Context-Aware Access Control Model for Pervasive Computing Environments.* Intelligent Pervasive Computing, International Conference on, vol. 0, pages 51–56, 2007.

[Fahy 2004] Patrick Fahy and Siobhan Clarke. *CASS: a middleware for mobile context-aware applications.* In Workshop on Context Awareness, MobiSys, 2004. 168

[Ferraiolo 1992] D.F. Ferraiolo and D.R. Kuhn. *Role-Based Access Control*, 1992. 4, 6, 14, 27, 48, 125

[Ferraiolo 2003] David F. Ferraiolo, D. Richard Kuhn and Ramaswamy Chandramouli. Role-based access control. Artech House, Inc., Norwood, MA, USA, 2003. 6, 27, 48, 132

[Filho 2005] José Bringel Filho, Windson Viana, Reinaldo Braga and Rossana Andrade. *FRAMESEC: A Framework for the Application Development with End-to-End Security Provision in the Mobile Computing Environment.* In Conferences [Conferences 2005], pages 72–77. 187

[Filho 2009] José Bringel Filho and Hervé Martin. *A generalized context-based access control model for pervasive environments.* In SPRINGL '09: Proceedings of the 2nd SIGSPATIAL ACM GIS 2009 International Workshop on Security and Privacy in GIS and LBS, pages 12–21, New York, NY, USA, 2009. ACM. 6, 126, 217

[Filho 2010a] José Bringel Filho, Alina Dia Miron, Ichiro Satoh, Jérôme Gensel and Hervé Martin. *Modeling and Measuring Quality of Context Information in Pervasive Environments.* In AINA, 2010. 6, 9, 204

[Filho 2010b] Jose Bringel Filho, Windson Viana, Jerome Gensel and Herve Martin. *A Contextual Annotation-based Access Control Approach for Pervasive Environments.* In 2nd International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use (IWSSI SPMU), Pervasive 2010, Helsinki, Finland, 2010. 4, 198

[Finin 2008] T. Finin, A. Joshi, L. Kagal, J. Niu, R. Sandhu, W. Winsborough and B. Thuraisingham. *ROWLBAC: representing role based access control in OWL.* In SACMAT '08: Proceedings of the 13th ACM symposium on Access control models and technologies, pages 73–82, New York, NY, USA, 2008. ACM.

[Garcia-Morchon 2010] Oscar Garcia-Morchon and Klaus Wehrle. *Modular context-aware access control for medical sensor networks.* In SACMAT '10: Proceeding of the 15th ACM symposium on Access control models and technologies, pages 129–138, New York, NY, USA, 2010. ACM.

[Georgakopoulos 1995] Diimitrios Georgakopoulos, Mark Hornick and Amit Sheth. *An overview of workflow management: From process modeling to workflow automation infrastructure.* Distributed and Parallel Databases, vol. 3, no. 2, pages 119–153, April 1995. 125

[Georgiadis 2001] Christos K. Georgiadis, Ioannis Mavridis, George Pangalos and Roshan K. Thomas. *Flexible team-based access control using contexts.* In SACMAT '01: Proceedings of the sixth ACM symposium on Access control models and technologies, pages 21–27, New York, NY, USA, 2001. ACM.

[Gligor 1998] V.D. Gligor, S.I. Gavrila and D. Ferraiolo. *On the Formal Definition of Separation-of-Duty Policies and their Composition.* Security and Privacy, IEEE Symposium on, vol. 0, page null, 1998. 141

[Graham 1972] G. Scott Graham and Peter J. Denning. *Protection: principles and practice.* In AFIPS '72 (Spring): Proceedings of the May 16-18, 1972, spring joint computer conference, pages 417–429, New York, NY, USA, 1972. ACM. 15

[Grossmann 2009] Matthias Grossmann, Nicola Hönle, Carlos Lübbe and Harald Weinschrott. *An abstract processing model for the quality of context data.* In QuaCon'09: Proceedings of the 1st international conference on Quality of context, pages 132–143, Berlin, Heidelberg, 2009. Springer-Verlag. 97, 98, 99, 101, 110, 113

[Gu 2004] Tao Gu, Hung Keng Pung, Da Qing Zhang and Xiao Hang Wang. *A Middleware for Building Context-Aware Mobile Services.* In In Proceedings of IEEE Vehicular Technology Conference (VTC, 2004. 168

[Halaschek-Wiener 2005] Christian Halaschek-Wiener, Jennifer Golbeck, Andrew Schain, Michael Grove, Bijan Parsia and Jim Hendler. *Photostuff - an image annotation tool for the semantic web.* In Poster Proceedings of the 4th International Semantic Web Conference, 2005. 200

[Hansen 2003] Frode Hansen and Vladimir Oleshchuk. *SRBAC: A Spatial Role-Based Access Control Model for Mobile Systems.* In Nordsec 2003, pages 129–141, Gjøvik, Norway,, 15-17 October 2003. iii, 38, 48, 112

[Harrison 1976] Michael A. Harrison, Walter L. Ruzzo and Jeffrey D. Ullman. *Protection in operating systems.* Commun. ACM, vol. 19, no. 8, pages 461–471, 1976. 6, 15, 48

[Henricksen 2002] Karen Henricksen, Jadwiga Indulska and Andry Rakotonirainy. *Modeling Context Information in Pervasive Computing Systems.* In Pervasive '02: Proceedings of the First International Conference on Pervasive Computing, pages 167–180, London, UK, 2002. Springer-Verlag. 81

[Henricksen 2004] Karen Henricksen and Jadwiga Indulska. *Modelling and Using Imperfect Context Information.* In PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, page 33, Washington, DC, USA, 2004. IEEE Computer Society. 80

[Hofer 2002] Thomas Hofer, Mario Pichler, Gerhard Leonhartsberger, Josef Altmann and Werner Retschitzegger. *Context-awareness on mobile devices: the hydrogen approach.* In Proceedings of the 36th Annual Hawaii International Conference on System Sciences, pages 292–302, 2002. 168

[Hu 2004] Junzhe Hu and Alfred C. Weaver. *Dynamic, context-aware access control for distributed healthcare applications.* In First Workshop on Pervasive Security, Privacy and Trust (PSPT), 2004.

[Huebscher 2005a] C. Huebscher and A. McCann. *An adaptive middleware framework for context-aware applications.* Personal Ubiquitous Comput., vol. 10, no. 1, pages 12–20, 2005.

[Huebscher 2005b] Markus C. Huebscher and Julie A. McCann. *A Learning Model for Trustworthiness of Context-Awareness Services.* In PERCOMW '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops, pages 120–124, Washington, DC, USA, 2005. IEEE Computer Society.

[Hulsebosch 2005] R. J. Hulsebosch, A. H. Salden, M. S. Bargh, P. W. G. Ebben and J. Reitsma. *Context sensitive access control.* In SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies, pages 111–119, New York, NY, USA, 2005. ACM. iv, 5, 71, 72, 77, 112

[Indulska 2003] Jadwiga Indulska, Ricky Robinson, Andry Rakotonirainy and Karen Henricksen. *Experiences in Using CC/PP in Context-Aware Systems.* In MDM '03: Proceedings of the 4th International Conference on Mobile Data Management, pages 247–261, London, UK, 2003. Springer-Verlag. 81

[Jea 2007] David Jea, Ian Yap and Mani B. Srivastava. *Context-aware access to public shared devices.* In HealthNet '07: Proceedings of the 1st ACM SIGMOBILE international workshop on Systems and networking support for healthcare and assisted living environments, pages 13–18, New York, NY, USA, 2007. ACM.

[Jiang 2002] Changhao Jiang and Peter Steenkiste. *A Hybrid Location Model with a Computable Location Identifier for Ubiquitous Computing.* In UbiComp '02: Proceedings of the 4th international conference on Ubiquitous Computing, pages 246–263, London, UK, 2002. Springer-Verlag. 38, 112

[Joshi 2005] James B.D. Joshi, Elisa Bertino, Usman Latif and Arif Ghafoor. *A Generalized Temporal Role-Based Access Control Model.* IEEE Transactions on Knowledge and Data Engineering, vol. 17, pages 4–23, 2005. 37, 43, 44, 48, 112

[Kahan 2001] José Kahan and Marja-Ritta Koivunen. *Annotea: an open RDF infrastructure for shared Web annotations.* In WWW '01, pages 623–632, New York, NY, USA, 2001. ACM. 199, 200

[Kim 2005] Young-Gab Kim, Chang-Joo Moon, Dongwon Jeong, Jeong-Oog Lee, Chee-Yang Song and Doo-Kwon Baik. *Context-Aware Access Control Mechanism for Ubiquitous Applications.* In AWIC, pages 236–242, 2005. iii, iv, 59, 60, 61

[Kim 2006a] Kyu-il Kim, Hyun-Sik Hwang, Hyuk-Jin Ko, Hae-Kyung Lee and Ungmo Kim. *Multi-Policy Access control considering Privacy in Ubiquitous Environment.* In ICHIT '06: Proceedings of the 2006 International Conference on Hybrid Information Technology, pages 216–222, Washington, DC, USA, 2006. IEEE Computer Society.

[Kim 2006b] Younghee Kim and Keumsuk Lee. *A Quality Measurement Method of Context Information in Ubiquitous Environments.* In ICHIT '06: Proceedings of the 2006 International Conference on Hybrid Information Technology, pages 576–581, Washington, DC, USA, 2006. IEEE Computer Society. 82, 97, 99, 101, 102, 108, 110, 113, 176, 177, 188

[Krause 2005] Michael Krause and Iris Hochstatter. *Challenges in Modelling and Using Quality of Context (QoC).* In MATA: Mobility Aware Technologies and Applications, Second International Workshop, volume 3744 of *Lecture Notes in Computer Science*, pages 324–333. Springer, 2005. iv, 80, 83, 84, 87, 89, 97, 110

[Kulkarni 2008] Devdatta Kulkarni and Anand Tripathi. *Context-aware role-based access control in pervasive computing systems.* In SACMAT '08: Proceedings of the 13th ACM symposium on Access control models and technologies, pages 113–122, New York, NY, USA, 2008. ACM.

[Kumar 2002] Arun Kumar, Neeran Karnik and Girish Chafle. *Context sensitivity in role-based access control.* SIGOPS Oper. Syst. Rev., vol. 36, no. 3, pages 53–66, 2002. iv, 61, 62, 112

[Lachmund 2006] Sven Lachmund, Thomas Walter and Laurent Bussard. *Context-Aware Access Control Making Access Control Decisions Based on Context Information.* Mobile and Ubiquitous Systems, Annual International Conference on, vol. 0, pages 1–8, 2006.

[Lampson 1974] Butler W. Lampson. *Protection.* In 5th Princeton Symposium on Information Science and Systems, Reprinted in ACM Operating Systems Review, vol. 8, no. 1, pages 18–24, 1974. 14, 15, 48

[LaPadula 1973] Leonard J. LaPadula and David E. Bell. *Secure computer systems: A mathematical model.* Rapport technique, Electronic Systems Division, November 1973. 21

[Lee 2009] Hyun Lee, Jae Sung Choi and Ramez Elmasri. *A classification and modeling of the quality of contextual information in smart spaces.* Pervasive

Computing and Communications, IEEE International Conference on, vol. 0, pages 1–5, 2009.

[Levy 1984] Henry M. Levy. Capability-based computer systems. Butterworth-Heinemann, Newton, MA, USA, 1984. 14

[Li 2005] Huiying Li, Xiang Zhang, Honghan Wu and Yuzhong Qu. *Design and Application of Rule Based Access Control Policies.* In Proc of the Semantic Web and Policy Workshop, pages 34–41, 2005.

[Li 2008] Lin Li and Tianjie Cao. *Context-Role Based Access Control Model for Ubiquitous Computing Environment.* Asian Journal of Information Technology, vol. 7, pages 74–78, 2008. 64, 77

[Manzoor 2008] Atif Manzoor, Hong-Linh Truong and Schahram Dustdar. *On the Evaluation of Quality of Context.* In EuroSSC '08: Proceedings of the 3rd European Conference on Smart Sensing and Context, pages 140–153, Berlin, Heidelberg, 2008. Springer-Verlag. iv, 89, 90, 91, 97, 99, 101, 102, 103, 108, 110, 113, 177, 188

[Manzoor 2009a] Atif Manzoor, Hong-Linh Truong and Schahram Dustdar. *Quality Aware Context Information Aggregation System for Pervasive Environments.* In WAINA '09: Proceedings of the 2009 International Conference on Advanced Information Networking and Applications Workshops, pages 266–271, Washington, DC, USA, 2009. IEEE Computer Society. 103, 110

[Manzoor 2009b] Atif Manzoor, Hong Linh Truong and Schahram Dustdar. *Using Quality of Context to Resolve Conflicts in Context-Aware Systems.* In Rothermel et al. [Rothermel 2009], pages 144–155. 103, 110

[Meersman 2006] Robert Meersman, Zahir Tari and Pilar Herrero, editeurs. On the move to meaningful internet systems 2006: Otm 2006 workshops, otm confederated international workshops and posters, awesome, cams, cominf, is, ksinbit, mios-ciao, monet, ontocontent, orm, persys, otm academy doctoral consortium, rdds, swws, and sebgis 2006, montpellier, france, october 29 - november 3, 2006. proceedings, part ii, volume 4278 of *Lecture Notes in Computer Science*. Springer, 2006. 274

[Moyer 2001] Matthew J. Moyer and Mustaque Ahamad. *Generalized Role-Based Access Control.* Distributed Computing Systems, International Conference on, vol. 0, page 0391, 2001. 46, 49, 112

[Mühlhäuser 2009] Max Mühlhäuser and Melanie Hartmann. *Interacting with Context.* In Rothermel et al. [Rothermel 2009], pages 1–14. 106

[Naphade 2006] Milind Naphade, John R. Smith, Jelena Tesic, Shih-Fu Chang, Winston Hsu, Lyndon Kennedy, Alexander Hauptmann and Jon Curtis. *Large-Scale Concept Ontology for Multimedia.* IEEE Multimedia, vol. 13, pages 86–91, 2006. 200

[Neisse 2008] Ricardo Neisse, Maarten Wegdam and Marten van Sinderen. *Trustworthiness and Quality of Context Information.* In ICYCS '08: Proceedings of the 2008 The 9th International Conference for Young Computer Scientists, pages 1925–1931, Washington, DC, USA, 2008. IEEE Computer Society.

[Neumann 2003] Gustaf Neumann and Mark Strembeck. *An approach to engineer and enforce context constraints in an RBAC environment.* In SACMAT '03: Proceedings of the eighth ACM symposium on Access control models and technologies, pages 65–79, New York, NY, USA, 2003. ACM. iv, 65, 67

[OWL 2009] W3 OWL. *OWL Web Ontology Language.* `http://www.w3.org/TR/owl-features/`, 2009. 9, 11

[Park 2006] Seon-Ho Park, Young-Ju Han and Tai-Myoung Chung. *Context-Role Based Access Control for Context-Aware Application.* In High Performance Computing and Communications, Second International Conference, HPCC 2006, Munich, Germany, September 13-15, 2006, Proceedings, volume 4208 of *Lecture Notes in Computer Science*, pages 572–580. Springer, 2006. iv, 63, 64

[Pemmaraju 2003] Sriram Pemmaraju and Steven Skiena. Computational discrete mathematics: Combinatorics and graph theory with mathematica. Cambridge University Press, New York, NY, USA, 2003. 140

[Preuveneers 2006] Davy Preuveneers and Yolande Berbers. *Quality Extensions and Uncertainty Handling for Context Ontologies.* In Pavel Shvaiko, Jérôme Euzenat, Alain Léger, Deborah L. McGuinness and Holger Wache, editeurs, Proceedings of (C&O 2006), pages 62–64, Riva del Garda, Italy, August 2006. iv, 92, 93, 110, 113, 176

[Priebe 2004] Torsten Priebe, Eduardo B. Fernandez, Jens I. Mehlau and Günther Pernul. *A Pattern System For Access Control.* In in Research Directions in Data and Applications Security XVIII, C. Farkas and P. Samarati (Eds.), Procs of the 18th. Annual IFIP WG 11.3 Working Conference on Data and Applications Security, pages 25–28. Kluwer, 2004. 14

[Priebe 2006] Torsten Priebe, Wolfgang Dobmeier and Nora Kamprath. *Supporting Attribute-based Access Control with Ontologies.* In ARES '06: Proceedings of the First International Conference on Availability, Reliability and Security, pages 465–472, Washington, DC, USA, 2006. IEEE Computer Society.

[Ramos 2007] Angela Carrillo Ramos, Marlène Villanova-Oliver, Jérôme Gensel and Hervé Martin. *Contextual User Profile for Adapting Information in Nomadic Environments.* In WISE Workshops, pages 337–349, 2007. 205

[Ray 2007] Indrakshi Ray and Manachai Toahchoodee. *A Spatio-temporal Role-Based Access Control Model.* In In Proceedings of the 21st Annual IFIP WG

11.3 Working Conference on Data and Applications Security, pages 211–226, 2007. 43, 112

[Ray 2008] Indrakshi Ray and Manachai Toahchoodee. *A Spatio-temporal Access Control Model Supporting Delegation for Pervasive Computing Applications.* In TrustBus '08: Proceedings of the 5th international conference on Trust, Privacy and Security in Digital Business, pages 48–58, Berlin, Heidelberg, 2008. Springer-Verlag. 43, 112

[Razzaque 2005] Mohammad Abdur Razzaque, Simon Dobson and Paddy Nixon. *Categorization and Modelling of Quality in Context Information.* In Proceedings of the IJCAI 2005 Workshop on AI and Autonomic Communications, 2005. iv, 87, 88, 113, 176, 177

[Rothermel 2009] Kurt Rothermel, Dieter Fritsch, Wolfgang Blochinger and Frank Dürr, editeurs. Quality of context, first international workshop, quacon 2009, stuttgart, germany, june 25-26, 2009. revised papers, volume 5786 of *Lecture Notes in Computer Science.* Springer, 2009. 279

[Samarati 2001] Pierangela Samarati and Sabrina Vimercati. *Access Control: Policies, Models, and Mechanisms.* Foundations of Security Analysis and Design, pages 137–196, 2001. 8, 14, 17, 48

[Sampemane 2002] Geetanjali Sampemane, Prasad Naldurg and Roy H. Campbell. *Access Control for Active Spaces.* In ACSAC '02: Proceedings of the 18th Annual Computer Security Applications Conference, page 343, Washington, DC, USA, 2002. IEEE Computer Society. iii, 55, 57, 58, 112

[Sandhu 1994] R. Sandhu and P. Samarati. *Access Control: Principles and Practice.* IEEE Communications, vol. 32, no. 9, pages 40–48, September 1994. 6

[Sandhu 1996] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein and Charles E. Youman. *Role-Based Access Control Models.* Computer, vol. 29, pages 38–47, 1996. iii, 14, 27, 28, 29, 30, 32, 34, 35, 47, 48, 53, 125, 129

[Sandhu 2000] Ravi Sandhu, David Ferraiolo and Richard Kuhn. *The NIST Model for Role-Based Access Control: Towards A Unified Standard.* In In Proceedings of the fifth ACM workshop on Role-based access control, pages 47–63, 2000. iii, 27, 28, 29, 30, 32, 34

[Schroeter 2006] Ronald Schroeter, Jane Hunter, Jonathon Guerin, Imran Khan and Michael Henderson. *A Synchronous Multimedia Annotation System for Secure Collaboratories.* In E-SCIENCE '06, page 41, Washington, DC, USA, 2006. IEEE Computer Society. 199, 200

[Sheikh 2007] Kamran Sheikh, Maarten Wegdam and Marten van Sinderen. *Middleware Support for Quality of Context in Pervasive Context-Aware Systems.* In

PERCOMW '07, pages 461–466, Washington, DC, USA, 2007. IEEE Computer Society. 113, 176, 177

[Sheikh 2008] Kamran Sheikh, Maarten Wegdam and Marten Sinderen. *Quality-of-Context and its use for Protecting Privacy in Context Aware Systems.* JSW, vol. 3, no. 3, pages 83–93, 2008. 97, 99, 108, 110, 113, 176, 177

[Strang 2004] Thomas Strang and Claudia Linnhoff-Popien. *A Context Modeling Survey.* In In: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England, 2004. 171

[Strembeck 2004] Mark Strembeck and Gustaf Neumann. *An integrated approach to engineer and enforce context constraints in RBAC environments.* ACM Trans. Inf. Syst. Secur., vol. 7, no. 3, pages 392–427, 2004.

[Tang 2007] Siliang Tang, Jianhua Yang and Zhaohui Wu. *A Context Quality Model for Ubiquitous Applications.* Network and Parallel Computing Workshops, IFIP International Conference on, vol. 0, pages 282–287, 2007. iv, 92, 94, 110

[TCSEC 1985] TCSEC. *Trusted Computer System Evaluation Criteria*, December 1985. 14, 15, 21, 48

[Toahchoodee 2008] Manachai Toahchoodee and Indrakshi Ray. *On the Formal Analysis of a Spatio-temporal Role-Based Access Control Model.* In Proceeedings of the 22nd annual IFIP WG 11.3 working conference on Data and Applications Security, pages 17–32, Berlin, Heidelberg, 2008. Springer-Verlag.

[Toahchoodee 2009] Manachai Toahchoodee, Indrakshi Ray, Kyriakos Anastasakis, Geri Georg and Behzad Bordbar. *Ensuring spatio-temporal access control for real-world applications.* In SACMAT '09: Proceedings of the 14th ACM symposium on Access control models and technologies, pages 13–22, New York, NY, USA, 2009. ACM.

[Toninelli 2006] Alessandra Toninelli, Rebecca Montanari, Lalana Kagal and Ora Lassila. *A Semantic Context-Aware Access Control Framework for Secure Collaborations in Pervasive Computing Environments.* In International Semantic Web Conference, pages 473–486, 2006. iv, 74, 75, 77, 112

[Viana 2007a] Windson Viana, José Bringel Filho, Jérôme Gensel, Marlene Villanova and Hervé Martin. *PhotoMap - Automatic Spatiotemporal Annotation for Mobile Photos.* In W2GIS 2007, Cardiff, UK, November 28-29, 2007, pages 187–201, 2007. 199

[Viana 2007b] Windson Viana, José Bringel Filho, Jérôme Gensel, Marlene Villanova and Hervé Martin. *A Semantic Approach and a Web Tool for Contextual Annotation of Photos Using Camera Phones.* In WISE, pages 225–236, 2007. 200

[Viana 2008] Windson Viana, José Bringel Filho, Jérôme Gensel, Marlene Villanova and Hervé Martin. *PhotoMap: from location and time to context-aware photo annotations.* J. Locat. Based Serv., vol. 2, no. 3, pages 211–235, 2008. v, 4, 172, 173, 177, 226

[Villalonga 2009] Claudia Villalonga, Daniel Roggen, Clemens Lombriser, Piero Zappi and Gerhard Troster. *Bringing quality of context into wearable human activity recognition systems.* In QuaCon'09: Proceedings of the 1st international conference on Quality of context, pages 164–173, Berlin, Heidelberg, 2009. Springer-Verlag. 107

[Wang 2008] Hua Wang, Yanchun Zhang and Jinli Cao. *Access control management for ubiquitous computing.* Future Gener. Comput. Syst., vol. 24, no. 8, pages 870–878, 2008.

[Wilikens 2002] Marc Wilikens, Simone Feriti, Alberto Sanna and Marcelo Masera. *A context-related authorization and access control method based on RBAC:.* In SACMAT '02: Proceedings of the seventh ACM symposium on Access control models and technologies, pages 117–124, New York, NY, USA, 2002. ACM.

[Wilkes 1979] M. V Wilkes. The cambridge cap computer and its operating system (operating and programming systems series). North-Holland Publishing Co., Amsterdam, The Netherlands, The Netherlands, 1979. 20, 48

[Wishart 2007] Ryan Wishart, Karen Henricksen and Jadwiga Indulska. *Context Privacy and Obfuscation Supported by Dynamic Context Source Discovery and Processing in a Context Management System.* In Jadwiga Indulska, Jianhua Ma, Laurence Yang, Theo Ungerer and Jiannong Cao, editeurs, Ubiquitous Intelligence and Computing, volume 4611 of *Lecture Notes in Computer Science*, pages 929–940. Springer Berlin, Heidelberg, 2007. 183

[XACML 2009] OASIS XACML. *OASIS eXtensible Access Control Markup Language (XACML).* http://www.oasis-open.org/, 2009.

[Yokoyama 2006] Shigetoshi Yokoyama, Eiji Kamioka and Shigeki Yamada. *An Anonymous Context Aware Access Control Architecture For Ubiquitous Services.* In MDM '06: Proceedings of the 7th International Conference on Mobile Data Management, page 74, Washington, DC, USA, 2006. IEEE Computer Society. iv, 72, 73, 74, 77, 112, 165

[Zhang 2004] Guangsen Zhang and Manish Parashar. *Context-Aware Dynamic Access Control for Pervasive Applications.* In Proc. Communication Networks

and Distributed Systems Modeling and Simulation Conference, 2004. iii, 58, 112

[Zhang 2006] Hong Zhang, Yeping He and Zhiguo Shi. *Spatial Context in Role-Based Access Control.* In Min Surp Rhee and Byoungcheon Lee, editeurs, ICISC - Information Security and Cryptology, 9th International Conference, volume 4296 of *Lecture Notes in Computer Science*, pages 166–178, Busan, Korea, November 2006. Springer. 41, 112

[Zimmer 2006] Tobias Zimmer, Telecooperation Office (teco and Universität Karlsruhe. *QoC: improving the performance of context-aware applications.* In Advances in Pervaisive Computing. Adjunct Proceedings of Pervasive 2006. APC, 2006.

Résumé: Dans les environnements pervasifs, le constant changement du contexte d'utilisation des applications, des services et des ressources distribués impose de nouvelles exigences pour la définition des solutions de contrôle d'accès. Celles-ci concernait notamment la sensibilité au contexte et la mise en ouvre distribuée des politiques d'accès. Pour prendre en compte ces besoins, nous proposons une famille de modèles de contrôle d'accès, appelé CxtBAC (Context-Based Access Control), qui se compose de huit modèles conceptuels et permet d'explorer les informations contextuelles caractérisant les entités suivantes : le propriétaire de la ressource, l'environnement, l'utilisateur et la ressource elle-même. Contrairement aux propositions existantes, basées sur le modèle RBAC (Role-Based Access Control), la famille de modèle proposé est centrée sur la notion de contexte et non de rôle. Par conséquent, les décisions d'accès aux ressources protégées sont prises en considérant les informations contextuelles qui caractérisent la situation des entités impliquées. Pour décrire les règles d'accès, les permissions sont associées à des contextes d'accès et les utilisateurs sont associés dynamiquement à ces dernières. Les modèles proposés sont indépendants du langage de spécification de la politique de sécurité. Dans le cadre de cette thèse, nous proposons également une solution pour la mise en ouvre basée sur un formalisme d'ontologies.

Mots-clés: Contrôle d'accès, environnements pervasifs, sensibilité au contexte, qualité du contexte, vie privée

Abstract: In pervasive environments, with the possibility of offering users distributed access on applications, services, and resources, from anywhere and at anytime, new issues arise with regard to access control mechanism. Generally, the existing access control solutions make static user-permission associations and are unaware about the situation (context) when defining and enforcing access control policies. In order to address these issues, we propose a family of Context-Based Access Control models, named CxtBAC (Context-Based Access Control), which is composed by eight conceptual models that can be used as basis to implement context-based access control solutions. CxtBAC models explore contextual information as central concept for assigning permissions to users. In fact, context information can describe the situation of resource owners, resource requestors, resources, and the environment around them. Unlike existing access control proposals such as RBAC-based solutions, CxtBAC makes access decisions taking into account the contextual information that characterizes the situation of involved entities. In a CxtBAC access rule, a set of permission is associated with an access context and users are dynamically associated with that access context. CxtBAC is independent of security policy language used to describe access control policies. Moreover, we have proposed an implementation of CxtBAC policies based on ontologies and inference rules.

Keywords: Access Control, Pervasive Environments, Context-awareness, Quality of Context, Privacy