

A CERTIFICATE BASED AUTHENTICATION CONTROL MODEL USING SMART  
MOBILE DEVICES FOR UBIQUITOUS COMPUTING ENVIRONMENTS

A THESIS SUBMITTED TO  
THE GRADUATE SCHOOL OF INFORMATICS  
OF  
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

DAVUT ÇAVDAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF  
MASTER OF SCIENCE  
IN  
THE DEPARTMENT OF INFORMATION SYSTEMS

SEPTEMBER 2011

Approval of the Graduate School of Informatics

---

Prof. Dr. Nazife Baykal

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

---

Prof. Dr. Yasemin Yardımcı Çetin

Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

---

Assist. Prof. Dr. P.Erhan Eren

Supervisor

Examining Committee Members

Assist. Prof. Dr. Sevgi Özkan (METU, II) \_\_\_\_\_

Assist. Prof. Dr. P. Erhan Eren (METU, II) \_\_\_\_\_

Assist. Prof. Dr. Aysu Betin Can (METU, II) \_\_\_\_\_

Dr. Nail Çadallı (KAREL A.Ş.) \_\_\_\_\_

Assoc. Prof. Dr. Altan Koçyiğit (METU, II) \_\_\_\_\_

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

**Name, Last name: Davut ÇAVDAR**

**Signature: \_\_\_\_\_**

## **ABSTRACT**

### **A CERTIFICATE BASED AUTHENTICATION CONTROL MODEL USING SMART MOBILE DEVICES FOR UBIQUITOUS COMPUTING ENVIRONMENTS**

Cavdar, Davut

M.Sc., Department of Information Systems

Supervisor: Assist. Prof. Dr. P.Erhan Eren

September 2011, 73 pages

In this thesis work, a certificate based authentication model supported by mobile devices is provided for ubiquitous computing environments. The model primarily aims to create an infrastructure for controlling and regulating access requests through mobile devices to local resources and services. The model also allows users from different domains to use local resources and services within the scope of agreements between domains. In addition to conceptual description of the model, a real prototype implementation is developed and successful application of the model is demonstrated. Within the prototype implementation, a mobile application is developed for access requests and sensors are used as representative local resources. Sample cases applied on the prototype demonstrate applicability and feasibility of the model.

Keywords: Access Control, Pervasive Computing, Context Aware Computing, Certificate Based Access Control, Smart Mobile Device

## ÖZ

### YAYGIN HESAPLAMA ORTAMLARI İÇİN AKILLI MOBİL CİHAZLAR KULLANAN SERTİFİKA TABANLI BİR ERİŞİM KONTROL MODELİ

Çavdar, Davut

Yüksek Lisans., Bilişim Sistemleri Bölümü

Tez Yöneticisi: Yrd. Doç. Dr. P.Erhan Eren

Eylül 2011, 73 sayfa

Bu tez çalışmasında dijital kaynakların çevreye yayıldığı ortamlar için mobil cihaz destekli ve sertifika tabanlı bir erişim kontrol modeli sunulmaktadır. Önerilen model öncelikli olarak, mobil cihazlardan lokal kaynak ve servislere doğru yapılan erişim isteklerini kontrol etmeyi ve düzenlemeyi amaçlamaktadır. Model aynı zamanda farklı alanlara ait kullanıcı gruplarının bu alanların kendi aralarındaki anlaşma durumlarına göre lokal kaynak ve servislere erişimine izin verir. Modelin kavramsal tanımının yanı sıra, gerçek bir prototipi geliştirilmiş ve modelin başarılı bir uygulaması gösterilmektedir. Bu prototip çalışma kapsamında, erişim isteklerinin iletilmesine olanak sağlayan bir mobil uygulama geliştirilmiş ayrıca sensör dataları lokal kaynak olarak kullanılmıştır. Prototip çalışma üzerinde uygulanan örnek senaryolar, çalışmanın uygulanabilirliğini ve yapılabirliğini göstermektedir.

Anahtar Kelimeler: Erişim Kontrolü, yaygın hesaplama ortamları, akıllı mobil cihaz, sertifika tabanlı erişim kontrolü, bağlam bilinçli hesaplama

## **ACKNOWLEDGEMENT**

I would like to thank my supervisor Assist. Prof. Dr. Erhan Eren, for his guidance, support and motivation throughout the development of this thesis.

I should thank my mother, father and sister for their guidance and unique supports in every moment of my life.

Samet Nargul, Burak Poyraz, Erhan iek, Uęur Bilgin and zeyir Doęan, you are the actors of our unforgettable memories. I would like to thank you for being with me whenever I need you.

I would like to thank to Serhat Peker, zge Grbz, Gken Yılmaz, M.Erhan Uyar for your precious friendships and motivations during my study.

I would like to thank to Esra Gnlal for her supports and motivations.

I would like to thank to Ahmet Yortanlı and Ahmet Oęuz Mermerkaya for their technical supports during development phase of my study.

I would like to thank to administrative personal of institute Necla Iřıklar, Sibel Gulnar and Ali Kantar for their help since the beginning of my M.Sc. study.

*This thesis is dedicated to my mother, my father and my sister.*

## TABLE OF CONTENTS

ABSTRACT .....	iv
ÖZ .....	v
ACKNOWLEDGEMENT .....	vi
LIST OF TABLES .....	x
TABLE OF FIGURES .....	xi
LIST OF ABBREVIATIONS AND ACRONYMS .....	xiii
CHAPTERS	
1 INTRODUCTION .....	1
1.1. Problem and Motivation .....	2
1.2. Scope of Study .....	3
1.3. Thesis Outline .....	3
2 LITERATURE REVIEW .....	4
2.1. Pervasive Computing .....	4
2.2. Pioneer Studies in Subject Domain .....	6
2.2.1. Studies at PARC: Pad and Tab .....	6
2.2.2. Project Aura .....	6
2.3. Context Aware Computing .....	8
2.4. Mobile Computing .....	9
2.5. Access Control Models .....	11
2.6. Certificate Based Access Control Models .....	11
2.6.1. Certificate .....	11
2.6.2. Certificate Based Access Control Models .....	11
3 PROPOSED MODEL .....	16
3.1. Main Characteristics of the Proposed Model .....	18
3.2. Components of the Proposed Model .....	20
3.3. Activity Flow of Proposed Model .....	21

3.4.	Mobile Device.....	24
3.4.1.	Hardware.....	24
3.4.2.	Software (Application).....	24
3.5.	Components of Gateway.....	26
3.5.1.	Interactions of Components of Gateway.....	26
3.5.2.	Certificate Service.....	26
3.5.3.	Context Engine .....	31
3.5.4.	Decision Engine .....	34
3.5.5.	Management Panel.....	36
4	PROTOTYPE IMPLEMENTATION.....	39
4.1.	Description of Prototype Implementation.....	39
4.2.	Components of Prototype .....	39
4.2.1.	Mobile Device.....	40
4.2.2.	Gateway .....	47
4.2.3.	Sensors .....	49
4.3.	Activity Flow of Prototype .....	53
4.4.	System Design .....	57
4.4.1.	Functional Requirements .....	59
4.4.2.	Assumptions.....	59
4.5.	Usage Cases of Prototype .....	59
4.5.1.	Parameters of Usage Cases .....	60
4.5.2.	Cases .....	61
5	CONCLUSION.....	67
5.1.	Summary .....	67
5.2.	Contribution and Discussion.....	68
5.3.	Future Work.....	68
	REFERENCES .....	70

## LIST OF TABLES

Table 2.1 Access Control Models.....	12
Table 4.1: Time Contexts for Usage Cases.....	61
Table 4.2: Access Rules for Usage Cases.....	61

## TABLE OF FIGURES

Figure 2.1 Shift between Computing Areas.....	5
Figure 2.2 The Aura Structure .....	7
Figure 2.3 Mobile Devices and Laptops Sales.....	9
Figure 2.4 Structure of Context Phone .....	10
Figure 2.5 Structure of a Certificate .....	13
Figure 3.1 General Structure of Proposed Model .....	17
Figure 3.2 Access Cases for Different Domain Users .....	19
Figure 3.3 Components and Sub-components of Proposed Model.....	20
Figure 3.4 Activity Flow of Proposed Model .....	23
Figure 3.5 Activity Diagram of Mobile Device.....	25
Figure 3.6 Interactions of Components of Gateway .....	27
Figure 3.7 Certificate Cancellation List Based Synchronization Method.....	29
Figure 3.8 Activity Diagram of Certificate Service.....	30
Figure 3.9 Components of Context Engine.....	31
Figure 3.10 Components of Decision Engine .....	36
Figure 3.11 Activity Diagram of Decision Engine .....	38
Figure 4.1 Components of Prototype .....	41
Figure 4.2 Structure of Android OS.....	42
Figure 4.3 Login Page of Mobile Application of Prototype .....	47
Figure 4.4 Interaction Diagram of Modules of Gateway Software.....	49
Figure 4.5 Microcontroller for Sensor Connections .....	50
Figure 4.6 Output of Sensors in Serial Monitor.....	51
Figure 4.7 Schema of Light Sensor Connection .....	52
Figure 4.8 Schema of Temperature Sensor Connection .....	52
Figure 4.9 Transmission of Sensor Data.....	53
Figure 4.10 Login and Certificate Selection Interfaces of Mobile Application.....	54
Figure 4.11 Browsing SD Card and Certificate Content Interfaces of Mobile Application ..	55
Figure 4.12 Resource Selection and Gateway Response Interfaces of Mobile Application..	56

Figure 4.13 Light Sensor Data and Access Denial Interfaces of Mobile Application .....	57
Figure 4.14 Data Flow Diagram of Prototype Implementation .....	58

## LIST OF ABBREVIATIONS AND ACRONYMS

PC	: Personal Computer
USB	: Universal Serial Bus
PARC	: Palo Alto Research Center
PDA	: Personal Digital Assistants
RFID	: Radio Frequency Identification
GPS	: Global Positioning System
CSL	: Computer Science Laboratory
QR	: Quick Response
DAC	: Discretionary Access Control
MAC	: Mandatory Access Control
RBAC	: Role-Based Access Control
TRBAC	: Temporal Role-Based Access Control
GRBAC	: Generalized Role-Based Access Control
DRBAC	: Dynamic Context Aware Access Control
RCBAC	: Role and Context Based Access Control
T-CAC	: Threshold Based Collaborative Access Control
CAAC-TC	: Context-aware access control using threshold cryptography
CA	: Certificate Authorities
LDAP	: Lightweight Directory Access Protocol
WLAN	: Wireless Local Area Network
CCL	: Certificate Cancellation List
ACL	: Active Certification List

AR	: Access Rules
ARS	: Access Resource System
AOSP	: Open Source Project
APK	: Android Package File
SOAP	: Simple Object Access Control
WSDL	: Web Services Description Language
IP	: Internet Protocol
DHCP	: Dynamic Host Configuration Protocol
I2C	: Inter-Integrated Circuit
METU	: Middle East Technical University
BOUN	: Bogazici University

# **CHAPTER 1**

## **INTRODUCTION**

Computers have emerged in order to make human life easier than mechanic and static life workflows. In the first states of its evolution, computers function as mainframes of projects and companies. They are used for increasing efficiency of required tasks and decreasing required labor force and time. These mainframe computers are static devices and have less interaction capabilities with human and other electronic devices.

After the rapid developments and changes in technological fields, computers are affected from this evolution both mechanically and logically. Computers started to be accepted as a part of human daily life. Their sizes have become smaller relative to mainframe computers and they started to be used in homes. Therefore, the “Personal Computer (PC)” term is born. They are used for special purposes and their interaction with human and other electronic devices have increased.

Big impact has occurred in computer evolution with the developments in electronic device production and computer networks basics. This state is defined as third era of computing (Weiser M. , 1991) and named as “Ubiquitous Computing” or “Pervasive Computing”. In this stage, standard computer perception changes both in appearance and logic. Devices become smaller relative to even personal computers and cheaper. Also device diversity increase with the needs of humans for daily life activities. Besides standard computers, mobile devices, sensors, actuators etc. have started to be used by humans in daily life. The major contribution of ubiquitous computing emerges in interaction of these devices. Unlike standard devices, these ubiquitous computing devices have effective interaction capabilities with both human and other electronic devices. The actual contribution of ubiquitous computing is that these small and smart computers are densely distributed in the environment; they work and interact in the background without appearing and disturbing people.

Context Aware Computing and Mobile Computing are supplementary areas of ubiquitous computing. Because of having sensing and interaction capabilities of ubiquitous devices, they can monitor environmental changes and react accordingly. These smart devices sense environmental contexts such as location, user, time, etc. and perform according to these contexts. Because of distributed resources and computers in the environment and nature of ubiquitous computing, smart mobile devices are used in order to reach to them and have interaction.

### **1.1. Problem and Motivation**

In today's world, humans are involved in many interactions with ubiquitous devices. Like any other system, access requests to devices should be controlled and regulated. Especially it is important to prevent unauthorized access to resources in ubiquitous environments.

Mainly two types of authentication methodology are offered for access control: one is static authentication, the other is dynamic authentication. Password based authentication is the most popular authentication method for static authentication. Smart card, biologic, Universal Serial Bus (USB) token and certificate based authentications are examples of dynamic authentication. For ubiquitous computing environments, static authentication is not suitable because access evaluation results can change according to user, location, time etc. contexts. In this study: a certificate based and context aware access control model using smart mobile devices is offered for ubiquitous and multi domain environments.

In the literature, certificate based access control models are offered, however they cannot satisfy three prerequisite conditions which are; supporting mobile device usage, working in multi domain environment and acting as context aware. For example, an extension model of Role Based Access Control (RBAC) is suggested for access control. (Chadwick, Otenko, & Ball, 2003), however, this model is not a context aware model. Also a certificate based access control model is offered (Yortanli, 2011), which is a context aware model and works in multi domain environment; however, it does not support mobile device usage.

## **1.2. Scope of Study**

Within the scope of this thesis, when user wants to reach a local resource or service, certificate data and requested resource type are sent to gateway. Gateway validates certificate by controlling required fields in certificate and also it collects location contexts using wireless connection properties and time context. System performs an evaluation according to pre-defined access rules and collected contexts in the scope of this study.

Applying context matching algorithms and controlling certificate integrity are out of the scope of this thesis.

## **1.3. Thesis Outline**

The thesis consists of five chapters. Chapter 2 includes literature information about pervasive computing, pioneer studies about context aware computing, mobile computing and role based access control models including certificate based access control models.

Chapter 3 explains the proposed model. Main characteristics as well as activity flow of the proposed model are explained. Mobile device structure and details gateway components are also described in this chapter.

Prototype implementation is introduced in Chapter 4. First components and activity flow of the prototype and next the system design and usage cases are described.

Chapter 5 concludes the thesis with summary of the study, contributions, discussions and possible future works of this study.

## **CHAPTER 2**

### **LITERATURE REVIEW**

This chapter provides literature information about the subject domain. Pioneer studies in the domain and offered models are also covered. First, Pervasive Computing is described, and then Mobile Computing and Context Aware Computing are analyzed. After that, authentication methods are explained, and finally offered access control models and certification based access control models are described.

#### **2.1. Pervasive Computing**

Pervasive Computing, also referred to as Ubiquitous Computing mainly reflects new era's computing trends. One of the first studies (Weiser, 1991) was conducted in Palo Alto Research Center (PARC), Xerox, CA, USA. Marc Weiser referred to as the thought leader in Ubiquitous Computing, analyzes user interactions with environment, human and other computers. He described Ubiquitous Computing as computers and electronic devices becoming smaller and cheaper, being deployed into environment, and finally getting embedded in the background and disappearing. He also emphasizes the increased interactions between humans and computers, also between computers and computers.

With the advances in technology, computers and smart electronic devices are getting integrated into humans' daily lives more frequently. Personal Digital Assistants (PDA), mobile phones, sensors, Radio Frequency Identification (RFID) based vehicle control systems and smart reaction systems in cars are some examples of such usage in daily life. The definition and perception of computers are changing as they are getting smaller and integrated into the environment. These devices, working in the background and inside many systems, may not even be visible. (Weiser & Brown, 1997)

Computing history can be partitioned into three main eras: Mainframe Computing, Personal Computing and Ubiquitous Computing each with its own characteristics. With the emergence of computers, first, mainframes serve humans' computing needs. Because of technical and economical reasons, mainframes are not convenient for daily usage. They have limited capacity and functionality, and also they are used by more than one person. In the second era, the number of computers increase and the term "Personal Computer" emerges. Each computer is generally used and owned by one person. And finally rapid development of technology creates Ubiquitous Computing wave. In this era, hundreds of computers are distributed into the environment and work in the background, having high interaction with humans and each other. Some examples are mobile devices, sensors, circuits, RFID, Global Positioning System (GPS), and Wi-Fi devices. These trends and the number of devices are illustrated in Figure 2.1 and this figure gives an idea about current computing trends. Mainframes appear first and the number of them increases, until personal computers start appearing which affects the rise of mainframes and eventually results in their decline. Increase rate of PCs decreases with the proliferation of ubiquitous computing devices, finally number of such devices reaches and passes that of PCs. (Weiser & Brown, 1997)

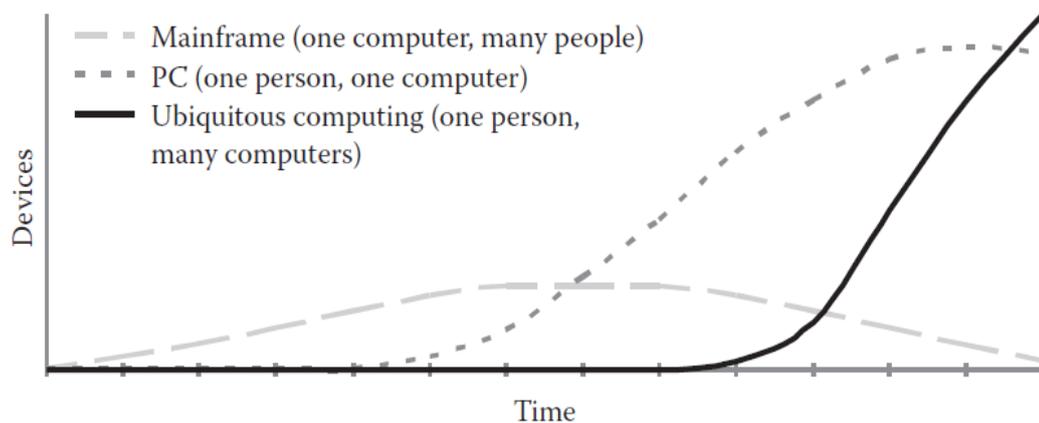


Figure 2.1 Shift between Computing Areas (Weiser M. , 1996)

## **2.2. Pioneer Studies in Subject Domain**

Marc Weiser, as the director of Computer Science Laboratory (CSL) at Palo Alto Research Center (PARC), and his colleagues started the study on ubiquitous computing environment and hardware solutions. They produced mobile devices named “Pad” and “Tab” that can be accepted as ancestors of today’s smart mobile devices. Another pioneer study is the Aura Project aiming to create an adapted pervasive environment using mobile devices. (Garlan, Siewiorek, Smailagic, & Steenkiste, 2002)

### **2.2.1. Studies at PARC: Pad and Tab**

Ubiquitous Computing environments provide distributed resources in the environment, hence the need for mobile devices emerges in order to reach and interact with them. CSL develops two different mobile devices in two different sizes. Inch-scale: ParcTab and foot-scale: ParcPad. (Weiser M. , 1991)

ParcTab is a pocket sized mobile device and has infrared wireless communication at 10 kbps. Each room has a base station on the ceiling and tabs connect to these base stations in order to join network, also base stations are connected to each other using wired connection.

ParcPad is a foot-scale mobile device having a pen and communication antenna. It uses low bandwidth radio link and short-range near-field radio (Katarjiev et al., 1993) principles in order to communicate with a base station. The pad has 3-4 meters wireless range and 250kbps communication speed.

### **2.2.2. Project Aura**

The main purpose of Aura is creating a ubiquitous environment and having all systems in this environment working in background unnoticed. All software and hardware operate without interfering with the daily life, sense their environment, collect required information and perform needed operations according to the situation, in other words adapting themselves to occurring changes. (Garlan, Siewiorek, Smailagic, & Steenkiste, 2002)

An Aura scenario and implementation that uses mobile authentication primarily focuses on working on different platforms and transferring related data between different environments. Fred who is the actor of in the scenario prepares a presentation and software demo in his office; however, he has not finished his preparation yet. He holds his smart mobile device and as he comes close to the door, Aura recognizes his location and mobile device (PALM). After mobile device authentication, Aura transfers his documents into his mobile device. He works on his presentation on his smart device while walking to the meeting room.

Aura also interprets his calendar and traces his walking direction and detects meeting room location, after which Aura transfers his presentation to a computer in the meeting room and warms the projector. No sooner than Fred enters the meeting room, Aura transfers final version of his presentation to the computer.

Aura Project is one of the initial successful ideas and implementations in the domain of mobile device usage in ubiquitous environments. The Aura architecture is illustrated in Figure 2.2.

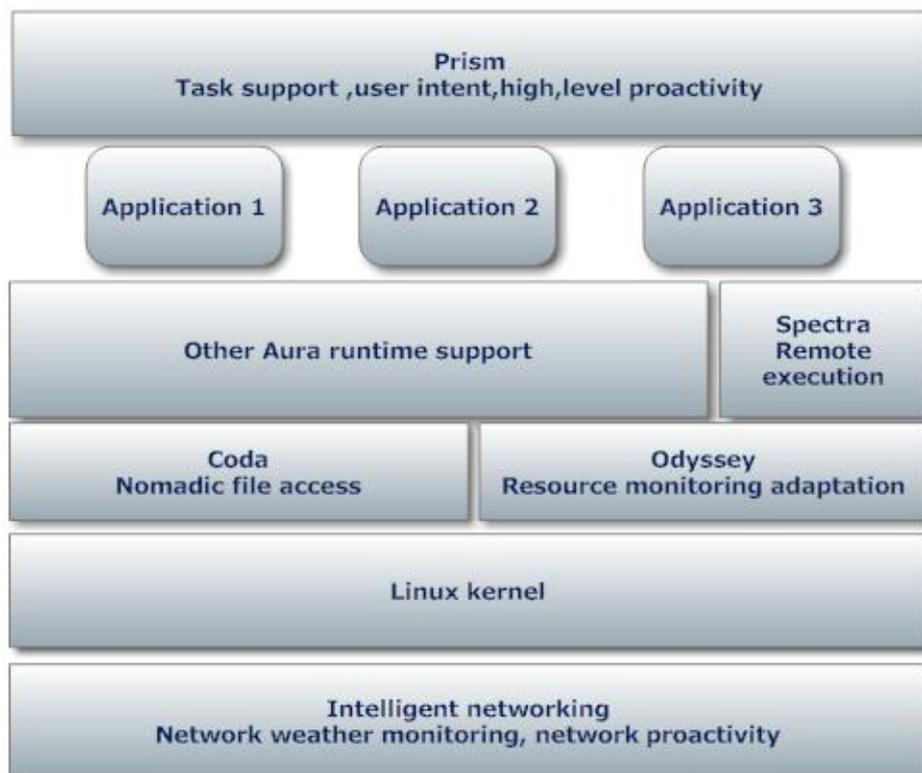


Figure 2.2 The Aura Structure (Garlan, Siewiorek, Smailagic, & Steenkiste, 2002)

### **2.3. Context Aware Computing**

Context Aware Computing has an important role in Pervasive Computing Environments. Computers and sensitive small electronic devices are distributed in the environment and they have interaction between among each other. Also they act according to the environment and may change their reaction according to changes in the environment.

According to the literature (Abowd & Mynatt, 2000) development process beginning with the studies by Marc Weiser and his colleagues at PARC about ubiquitous computing, incorporate three challenges. First one is using humans' "Natural Interface" for human computer interactions. Second is recording live experiences for future experiences.

Third challenge is "Context Awareness" of systems and devices in ubiquitous computing. Context Aware systems sense the changes in the environment; they adapt their behavior and perform according to contexts and pre-defined rules and manifests.

Context Aware Systems uses collected or created contexts for performing operations accordingly. Context as a term is described in studies and by researchers in many different ways. The five "W" descriptions (Abowd & Mynatt, 2000) of context include definitions that are "Who", "What", "Where", "When" and "Why". An answer of each of these questions generates a context. Location description of context (Schilit & Theimer, 1994) mainly focuses on the person's current location and descriptions of surrounding location. These location contexts are saved for future predictions on maps. Another context definition (Lieberman & Selker, 2000) is based on any input from outside environment to internal system.

A system can be defined as a Context Aware System as long as it senses the environment and reacts accordingly. Abowd and Dey, (1999) describe context aware systems as collecting required contexts in order to create meaningful information for users or systems. Another description of context aware systems (Fickas, Kortuem, & Segall, 1997) underlines adaptation of systems behaviors and reactions according to sensed changes of the environment. Also context aware systems are grouped as "active" and "passive" context aware systems. (Chen & Kotz., 2000) . Both of them monitor changes in the environment, however, active context aware systems adapt their reactions according to context and pre-defined rules, on the other hand, passive context aware systems sense the changes in the environment and send them to system users.

## 2.4. Mobile Computing

Mobile device usage has been increasing day by day with the help of rapid technological developments. Also changes in human behaviors and need for frequent communication result in high mobile device usage. In the emergence of mobile phones, oral communication and messaging among people are the basic needs. Later on, mobile phones start to be converted into smart mobile devices and these smart devices start to perform as mobile computers. Figure 2.3 gives an idea about the increase in smart mobile popularity in time.

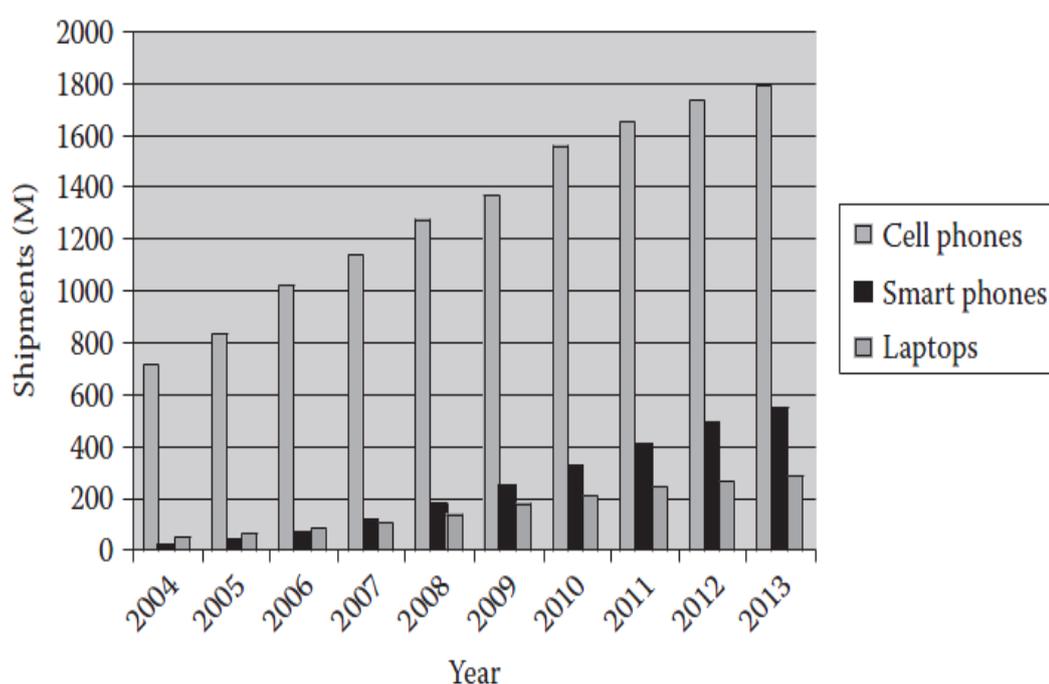


Figure 2.3 Mobile Devices and Laptops Sales (Want, 2009)

Smart mobile devices become an important part of ubiquitous computing due to their advanced functional specifications and also due to the importance of reaching distributed resources in ubiquitous environments. According to taxonomy (Satyanarayanan, 2001) adding features of mobile networking, mobile information access, adaptive applications, energy-aware systems and location sensitivity to distributed systems creates Mobile Computing philosophy also adding features of smart spaces, invisibility, localized scalability, and uneven conditioning to mobile computing creates Pervasive Computing.

One of the ubiquitous systems based on using smart mobile devices is Cooltown Project developed at HP laboratories (Kindberg, Barton, Morgan, Becker, Caswell, & Debaty, 2002). In this project, objects are labeled with Quick Response (QR) codes and when a user reads these QR codes using a mobile device's camera, it resolves an http address from the code and retrieves related content about the object from the Internet. Another pervasive system based on using a smart mobile device is the GUIDE Project developed at Lancaster University (Want, 2009). It is the first electronic city guide and with the capability of user location detection.

In addition, a context phone prototype and structure is offered in (Raento, Oulasvirta, Petit, & Toivonen, 2005). The structure of the context phone is illustrated in Figure 2.4 and it has mainly peripheral unit, internal hardware, communication unit, sensors, context module and applications. Similarly, in the prototype implementation of this thesis, a smart mobile device is used for sending access requests.

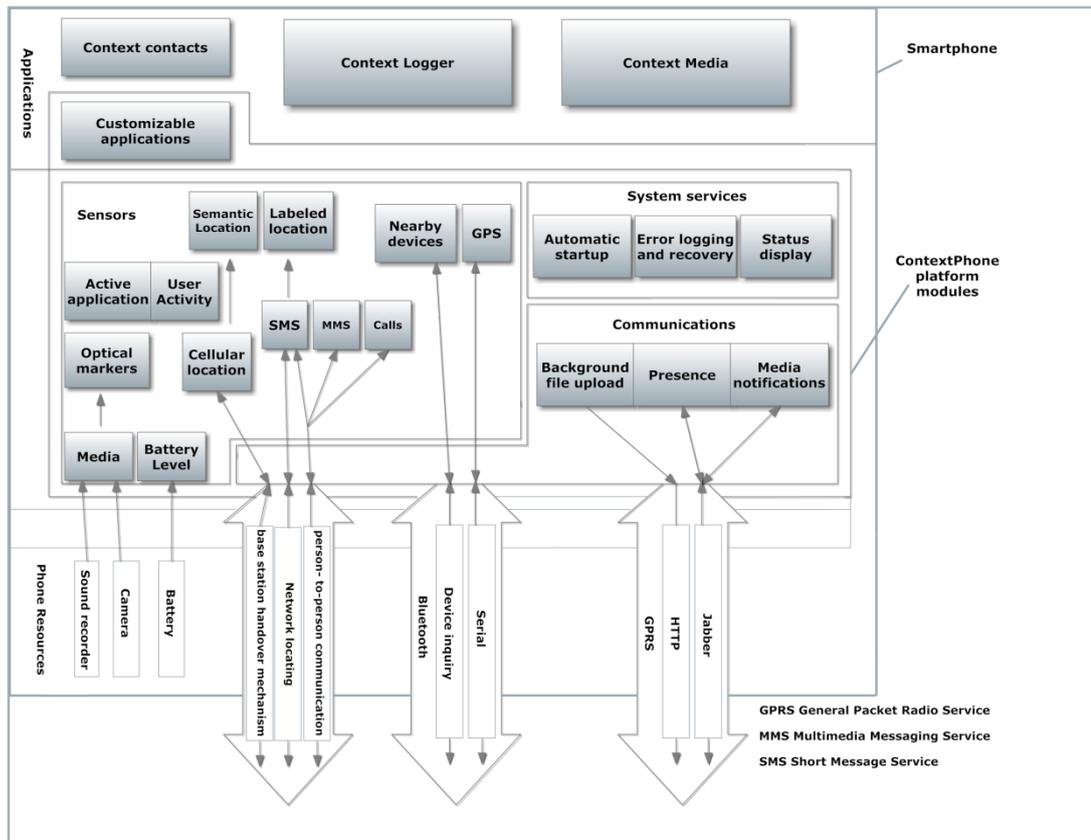


Figure 2.4 Structure of Context Phone (Raento, Oulasvirta, Petit, & Toivonen, 2005)

## **2.5. Access Control Models**

Almost all systems involve a security method in order to prevent unauthorized access attempts to resources. Role and context based access control systems are frequently used for access regulation in ubiquitous computing environments instead of using static access control methods such as using user name and password in access control. The main access control models for distributed and ubiquitous systems are summarized in Table 2.1.

## **2.6. Certificate Based Access Control Models**

In this section, certificate structure is described first, and then certificate based access control models are analyzed briefly.

### **2.6.1. Certificate**

Certificates are digital identities of a user, group, device or company. They show entities authenticity status. Certificates are generated by Certificate Authorities (CA) and validated according to CA validation methods. A certificate includes introductory sections containing subject name, time intervals, CA, public key and other information. Other part of a certificate is a signature section. It includes signature for certificate encryption and algorithm. The general structure of a certificate is illustrated in Figure 2.5

### **2.6.2. Certificate Based Access Control Models**

Certificate based authentication is generally performed using public key principles. In order to provide user authentication and certificate validation, a server creates random data and sends to the user who signs that data and sends it together with certificate back to the server. A server can validate certificates using public key algorithm. In this thesis, certificate integrity control is out of scope, certificate validation is done by checking user, certificate provider and valid time intervals. In this section, some offered certificate based access control methods in the literature are analyzed next.

Table2.1 Access Control Models

<b>Model</b>	<b>Description</b>
Discretionary Access Control (DAC) (Bertino, Bonatti, & Ferrari., 2001)	Access control over databases based on user identities and rules
Mandatory Access Control (MAC) (Ravi S. Sandhu, 1993)	Lattice based access control model. Primarily focuses on information flow of computer systems.
Role-Based Access Control (RBAC) (Sandhu, Coyne, Feinstein, & Youman., 1996)	Access grants are given according to role status of users instead of individual authorization.
Temporal Role-Based Access Control (TRBAC) (Bertino, Bonatti, & Ferrari., 2001)	Extension of RBAC model, time contexts can also be used in access control.
Generalized Role-Based Access Control (GRBAC) (Lim & Shin., 2007)	Access roles are defined in three different concepts which are subject roles, object roles and environmental roles.
Dynamic Context Aware Access Control (DRBAC) (Zhang & Parashar, 2003)	Provides a dynamic and adaptive access control mechanism monitoring user's dynamic context information.
Role and Context Based Access Control (RCBAC) (Han-bing, He-ping, Zheng-ding, & Rui-xuan., 2005)	Combines pre-defined users roles and users' current dynamic context information and regulates access requests.
Threshold Based Collaborative Access Control (T-CAC) (Alsulaiman, Mieke, & Saddik, 2007)	Access is allowed only when total access requests reach the defined threshold value.
Context-aware access control using threshold cryptography (CAAC-TC) (Al-Rwais & Al-Muhtadi, 2010)	Works based on "threshold cryptography" methodology and contexts. Data encryption and decryption are administered by access control rules.

```

Certificate:
  Data:
    Version: v3 (0x2)
    Serial Number: 3 (0x3)
    Signature Algorithm: PKCS #1 MD5 with RSA Encryption
    Issuer: OU=METU, O=Ace Industry, C=US
    Validity:
      Not Before: Fri Oct 17 18:36:25 1997
      Not After: Sun Oct 17 18:36:25 2011
    Subject: CN=davut, OU=Finance, O=Ace Industry, C=US
    Subject Public Key Info:
      Algorithm: PKCS #1 RSA Encryption
      Public Key:
        Modulus:
          00:ca:fa:79:98:8f:19:f8:d7:de:e4:49:80:48:e6:2a:2a:86:
          ed:27:40:4d:86:b3:05:c0:01:bb:50:15:c9:de:dc:85:19:22:
          43:7d:45:6d:71:4e:17:3d:f0:36:4b:5b:7f:a8:51:a3:a1:00:
          98:ce:7f:47:50:2c:93:36:7c:01:6e:cb:89:06:41:72:b5:e9:
          73:49:38:76:ef:b6:8f:ac:49:bb:63:0f:9b:ff:16:2a:e3:0e:
          9d:3b:af:ce:9a:3e:48:65:de:96:61:d5:0a:11:2a:a2:80:b0:
          7d:d8:99:cb:0c:99:34:c9:ab:25:06:a8:31:ad:8c:4b:aa:54:
          91:f4:15
        Public Exponent: 65537 (0x10001)
    Extensions:
      Identifier: Certificate Type
      Critical: no
      Certified Usage:
        SSL Client
      Identifier: Authority Key Identifier
      Critical: no
      Key Identifier:
        f2:f2:06:59:90:18:47:51:f5:89:33:5a:31:7a:e6:5c:fb:36:
        26:c9
    Signature:
      Algorithm: PKCS #1 MD5 with RSA Encryption
      Signature:
        6d:23:af:f3:d3:b6:7a:df:90:df:cd:7e:18:6c:01:69:8e:54:65:fc:06:
        30:43:34:d1:63:1f:06:7d:c3:40:a8:2a:82:c1:a4:83:2a:fb:2e:8f:fb:
        f0:6d:ff:75:a3:78:f7:52:47:46:62:97:1d:d9:c6:11:0a:02:a2:e0:cc:
        2a:75:Gc:8b:b6:9b:87:00:7d:7c:84:76:79:ba:f8:b4:d2:G2:58:c3:c5:
        b6:c1:43:ac:63:44:42:fd:af:c8:0f:2f:38:85:6d:d6:59:e8:41:42:a5:
        4a:e5:26:38:ff:32:78:a1:38:f1:ed:dc:0d:31:d1:b0:6d:67:e9:46:a8:
        dd:c4

```

Figure 2.5 Structure of a Certificate

A user authentication method using smart cards is offered as a certificate based authentication (Wang, Feng, & Wang, 2007). In this method, user certificate and other private information are stored in a smart card and the system performs authentication process based on the combination of smart card information and related context. Method focuses on authentication transactions; however, access control mechanisms and process are not discussed.

Another offered solution to access control uses certificate for access control for inter-domain environments. (Thompson, Johnston, Mudumbai, Hoo, Jackson, & Essiari, 1999). In this solution users send their certificates storing their roles in order to reach resources, however, major deficiency of this solution is that, it is not designed for context aware environments and context usage.

An access control method is offered for healthcare systems (Koufi & Vassilacopoulos, 2008). This method is designed for context aware environments. The system validates user roles stored in user certificates and evaluates certificate data with context information; however this model does not support giving access to other domain users for reaching resources.

An extension model of Role Based Access Control (RBAC) is suggested for access control. (Chadwick, Otenko, & Ball, 2003). The model uses X.509 based certificates that store user roles and definition for accessing resources. Access rules are defined as XML based policy rules and they are stored in Lightweight Directory Access Protocol (LDAP). The model also controls certification cancellation status using certificate revocation lists. However, the model is not suitable for context aware environment; this can be considered as the main shortcoming.

Ahmet Yortanlı (2011) offers a certificate based access control model for multi domain environments. The model allows other domain users to reach local resources with the service agreements between domains. Also the model is suitable for context aware systems because system performs evaluation according to both certificate data and related user location and time contexts.

The model (Yortanlı, 2011) offers a certification revocation list broadcasting between domains in order to inform domains about cancelled certificates. Each domain defines revoked certificates in the revocation list and system checks these lists in pre-defined time intervals. When user wants to reach resource, system first checks basic validity and these Certificate Revocation Lists (CRL) for certificate validity and activeness status.

In addition to model description two different scenarios are developed for showing model usage. The first scenario includes access control of users for university services. Resources in the first scenario are printers and online services in library. Both home domain and other domain (different university) users request access to local resources and system evaluates these requests according to user context and pre-defined rules.

Second scenario offers company discounts for telecommunication users. Telecommunication companies are certificate providers and define certificates to their users. The system controls users' location context according to which it offers discounts from companies, and also grants these discounts to them when they provide their certificates.

The idea of the model by (Yortanli, 2011) is used in the gateway modules of this study. However his model does not support mobile device usage in access requests, and therefore needs to be enhanced in order to be suitable for mobile and ubiquitous computing environments. Details are provided in Chapter 3.

## **CHAPTER 3**

### **PROPOSED MODEL**

This chapter provides the conceptual description of a certificate based authentication control model via smart mobile devices for ubiquitous environments. Proposed model offers a certificate based access control model for context aware systems and uses certificates to achieve authorization process, when a client (user) requests access for available local resources or services.

The major contribution of the proposed model is using smart mobile devices for connecting to the service, sending certificate information, requesting access and receiving responses processes. By the nature of ubiquitous and pervasive computing, mobility is one of the key issues for such systems. In pervasive systems especially in context aware systems, computers are embedded into the environment and resources are distributed. To reach these resources, use of mobile devices provides more advantages than fixed devices.

The proposed model is also an inter-domain access control model, meaning that, when a user from another domain wants to access a local resource, most other models demand the user's credentials from the user's home domain. This causes both management problems, including maintaining, updating rules, roles, and some network problems on inter-domain communications such as propagation delay, interruptions. However, the proposed model offers a synchronization method between different domains on defined time intervals. By using this methodology, when a user from another domain requests access to resources or services, the system obtains user's privileges in the local domain for decision instead of asking the user's home domain.

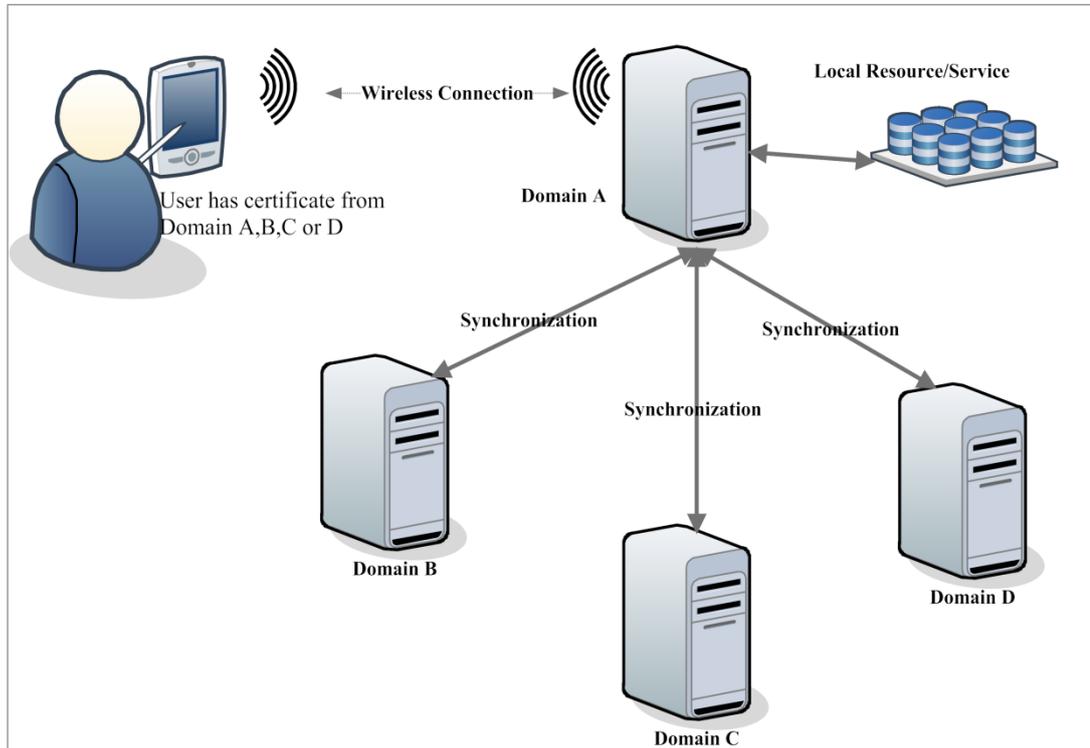


Figure3.1 General Structure of Proposed Model

The general structure of the proposed model is shown graphically in Figure 3.1. There are three main components of this model, first is the user processes running on a mobile device, second is the main gateway, performing duties such as certificate control, applying rules etc. and third is local resources or services. These components will be explained in this chapter in detail.

Gateway modules of the proposed model are designed based on an already offered certificate based access control model (Yortanlı, 2011). Similar to the model offered by Ahmet Yortanlı, a central and coordinator decision module is used in the proposed model. However, in the proposed model, rule and data service offered by Yortanlı are merged into a single module referred to as Context Engine, in order to decrease inter-module communication workload and traffic. Also different from Yortanlı's model, Data Receiver module is provided in the prototype implementation in order to retrieve sensor data. Major difference and focus of this thesis study rely on enhancing Yortanlı's offered model by adding new improvements and finally developing a new framework that uses mobile devices for access control and real sensor data as representative local resource. The most significant

contribution of this study relies on developing a real implementation based on the proposed model to show its applicability and feasibility.

The model primarily covers certificate based authentication control using mobile devices, applying policy rules for requests according to the user, location and time contexts, creating suitable responses to the user and performing synchronizations between other domains. Hence, other issues such as wireless and wired communication techniques between domains and local resources and context acquiring techniques are out of the scope of the thesis.

### **3.1. Main Characteristics of the Proposed Model**

The proposed model uses certificates for authorized access to resources. After connecting to the local service wirelessly, the user sends his/her certificate and resource/service type such as temperature/light sensors values or printer usage to gateway by using his/her smart mobile devices. After that, the gateway gathers required information and performs actions accordingly and finally produces a request result. This result is received by the user via his/her mobile device again. Certificate based authentication minimizes inter-domain communication load, because certificates carry both user identity and domain identity inside them. Also, it is easy and cheap to store certificate provider or user group identity instead of storing all user credentials.

Another important characteristic of the proposed system is providing a mobile usage environment to the system users. In ubiquitous environments, computers are hidden and resources/services are widely distributed. Also people are in transition to more mobility in terms of life styles and technology trends. Therefore, people have frequent interactions with embedded computers or resources when they are mobile. In the proposed model, home or other domain users can explore local resources/services when they are in a different location and they can send access requests to gateways.

Context-Awareness is another important feature of the proposed model. In order to respond to received requests correctly according to access policy rules, the system gathers context information from the environment. Since the proposed model is a context-aware system, it senses contextual information like location, mobile user id, time, resource type and it performs required actions according to this gathered contextual information.

The proposed model can perform authorization and access control requests conducted by not only different domain users but also by other domain users. To do that, domains provide an access policy rules agreement for their own users when they are using different domains' resources/services. According to the agreement between domains, each domain sets access rules for both home and other domains' users' requests. After these agreements, the system checks other domain certificate lists in order to update access lists of other domains in each pre-defined synchronization time intervals.

There are two cases for users' domain status similar to the model offered by Ahmet Yortanlı (2011). When the user requests an access to a local resource/service, if s/he is home domain user, the system checks access policy rules, acquire context information, evaluate request and produce a response accordingly. If s/he belongs to a different domain, the system first checks an agreement between domains, if it is available then it checks access rules for that user, collects contextual information and finally performs an evaluation and creates a response. The illustration of these two cases is in Figure 3.2.

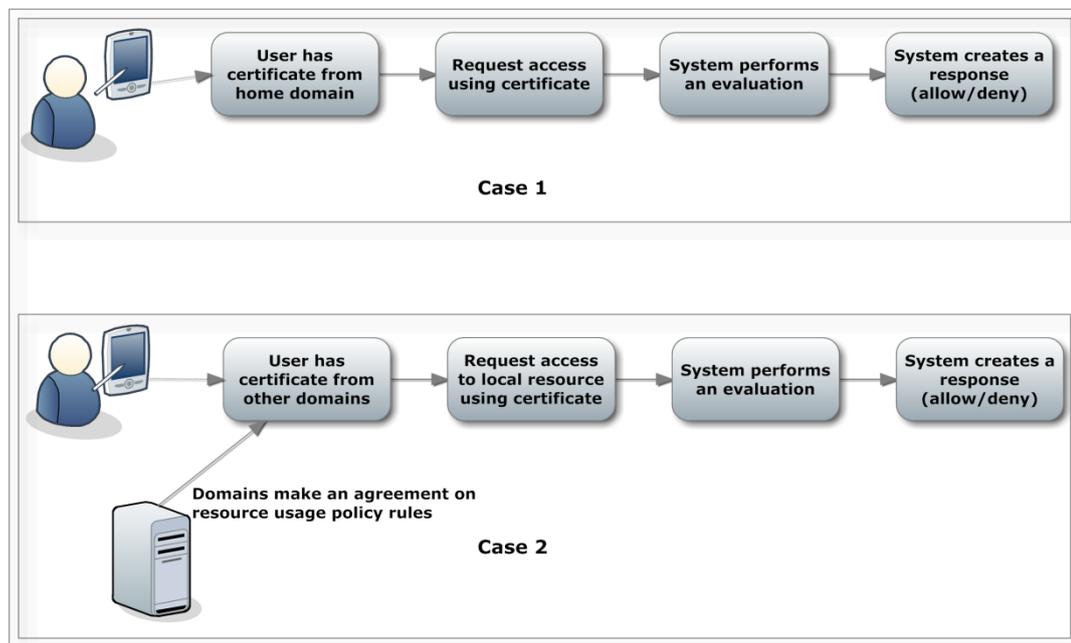


Figure3.2 Access Cases for Different Domain Users

### 3.2. Components of the Proposed Model

There are three main components of the proposed model, first is user processes running on a mobile device, second is the main gateway, performing duties like certificate control, applying rules etc. and third is local resources or services. These components and sub-components are illustrated in Figure 3.3 and will be explained in this chapter in detail.

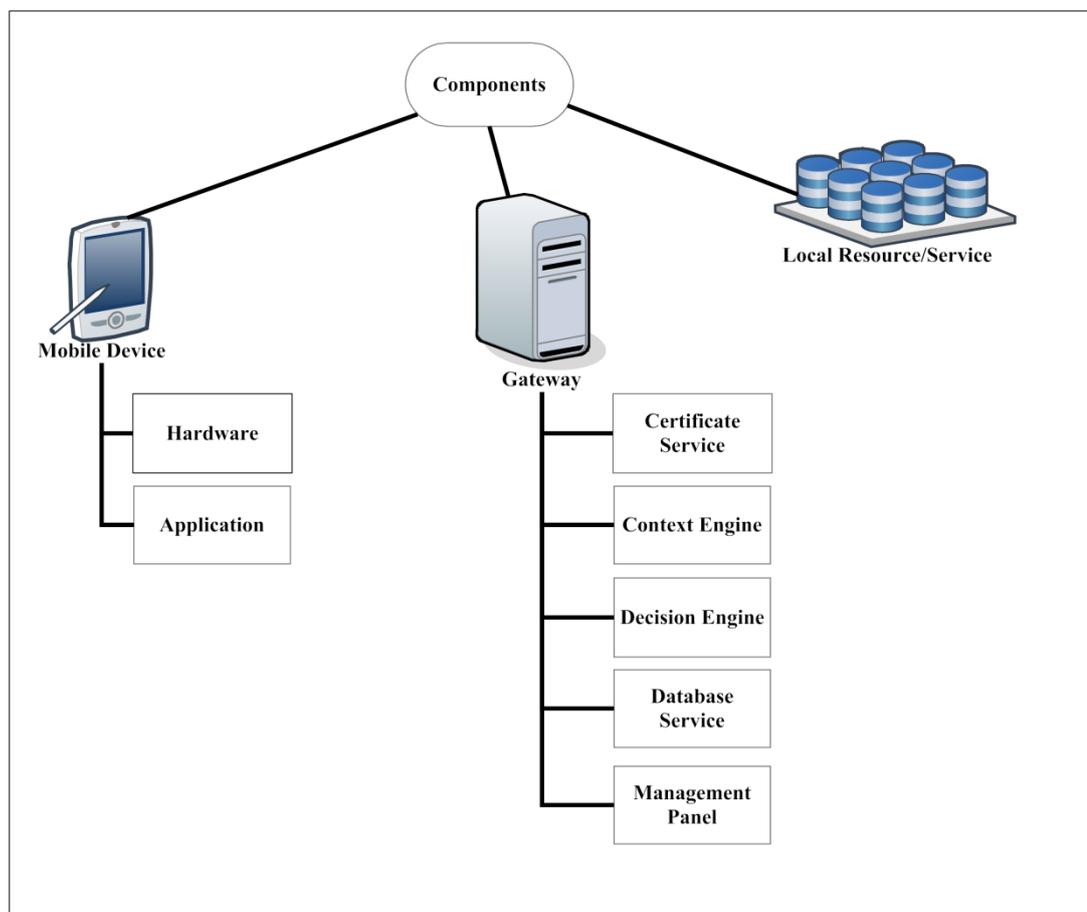


Figure3.3 Components and Sub-components of Proposed Model

The mobile device in this model –generally a PDA (Personal Digital Assistant) or a smart phone— is mainly equipped with advanced hardware units for communication, process and storing tasks. Main duties like certificate retrieval from home domain, certificate sending, access requests, and receiving system responses are performed by software (application) installed on the smart mobile device.

The gateway is the main unit of the model that accomplishes critical tasks and behaves like a bridge between the user (client) and the requested resources/services. Sub-components of the gateway are; Certificate Service, Context Engine, Decision Engine, Database Service and Management Panel. All sub-components are in a modular structure and designed according to abstraction principles. Therefore, they are all capable of working as a separate module and new modules can be added to the model as needed.

The Certificate Service is mainly responsible for checking certificates sent by the user during access request process. After certificate information reaches the gateway, Certificate Service parses it, and checks subject ID (user), provider ID (institution), validity dates (Not Before and Not After sections) of the certificate. Also the Certificate Service performs synchronization between domains. It checks domains' active certificate lists and if any change (add, delete, update) has occurred in these lists, Certificate Service updates required lists between home and other domains.

The Context Engine has mainly two duties in the proposed model. Its first task is context acquisition. Because model offers a context aware environment, context information such as location, time, group etc. about the requested resource and the user should be collected. The Context Engine collects required context information and sends them to the Decision Engine when the user demands access to the resource. Secondly, the Context Engine is responsible for managing context rules for the model. When the Decision Engine requests related rules for the defined user and resource, the Context Engine finds correct rules that will be applied for the request and sends them to the Decision Engine.

The Decision Engine is the core component of the proposed model. The user sends requests as an envelope to the Decision Engine. After receiving requests it opens envelope and defines user certificate data and resource IDs. Then, the Decision Engine demands required context information and related rules for the user from the Context Engine and sends certificate data to the Certificate Service in order to check certificate accuracy and also validity. It reaches a decision after collecting all these required data and rules.

The Database Service provides a communication infrastructure for all system modules and database. When a system module needs information stored in the database such as user group, resource ID, policy rules etc. they use the Database Service to get access to the related database.

The Management Panel allows system administrators to manage system parameters by using its interface. Administrators can perform management tasks such as add or delete rules, user groups etc. by using this panel.

### **3.3. Activity Flow of Proposed Model**

When a user requests to reach a local resource/service via his/her smart mobile device, first s/he downloads or saves his/her certificate provided by his/her host domain into his/her mobile device, then s/he establishes a wireless connection with the resource gateway. By using the application running on the mobile device, the user selects his/her certificate and requested resource type. This type may only be one such as printer usage or more than one such as sensors providing more than one resource type like temperature, light etc.. The mobile application generates a message envelope including “Certificate Data” and “Resource Type” and sends this envelope to the gateway.

After retrieving the request envelope, the Decision Engine opens it and parses the data. Certificate data is sent to the Certificate Service for validation process. The Certificate Service first parses certificate data for default validity check, also it controls synchronized active certificate lists of other domains for certificate validity and sends the result to the Decision Engine. After that, if the certificate passes the validity check, the Decision Engine requests related contexts and rules from the Context Engine. According to the user and resource type, the Context Engine finds related rules from the database and contexts and this information are delivered to the Decision Engine. During this process, the local resource sends required data to the gateway. This data type may vary according to the designed application. If it is a file access control system, data may be up-to-date version of files, if it is a printer access control system, data may be printer current status, if it is a sensor data access control system, data may be values read by sensors.

Finally, the Decision Engine collects required information from related modules and performs an evaluation. According to the results of the evaluation, the resource/service access is allowed or denied by the system. The model activity flow is illustrated in Figure 3.4.

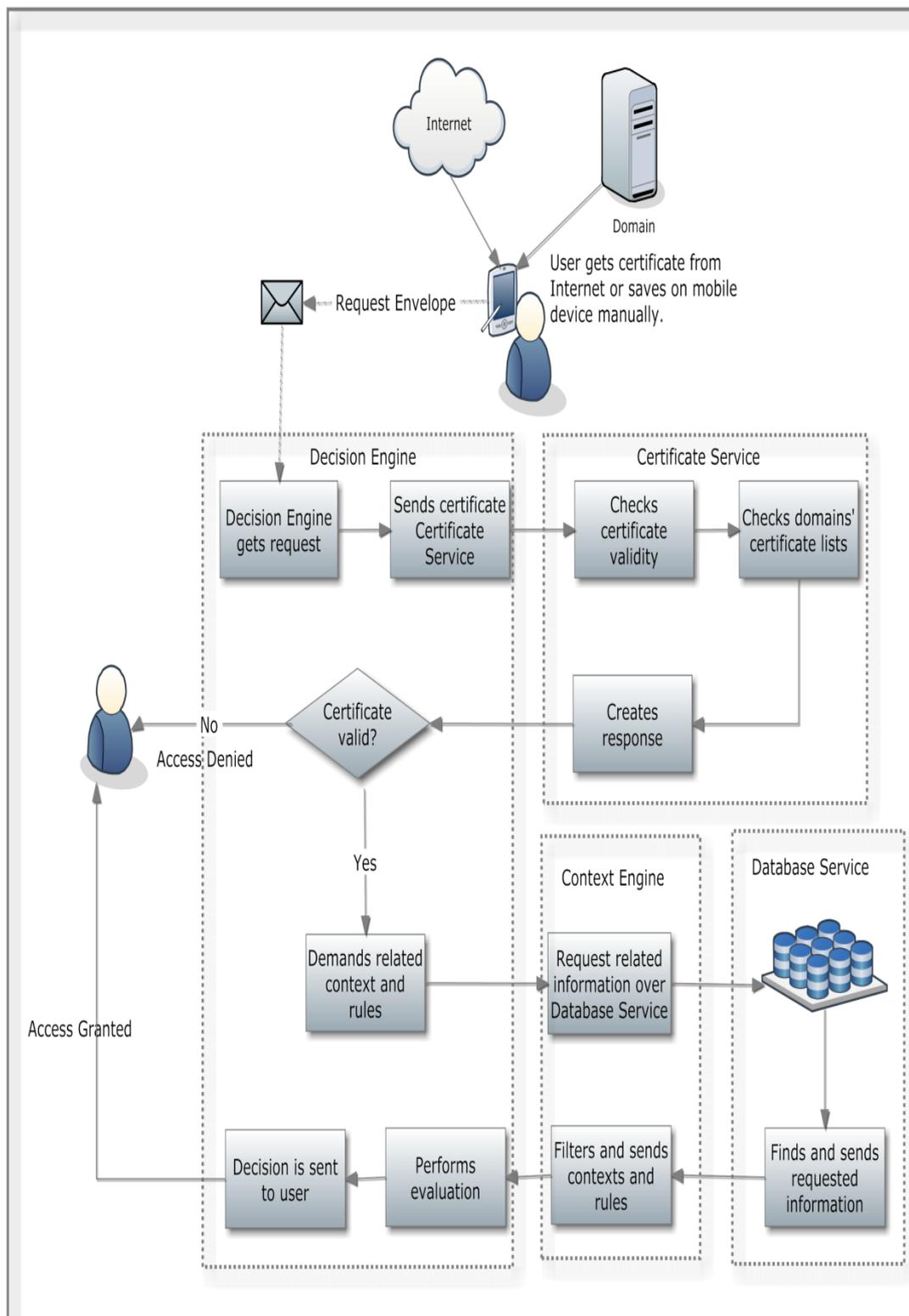


Figure3.4 Activity Flow of Proposed Model

### **3.4. Mobile Device**

The proposed model uses a mobile device for authorization and access control for reaching local resources. With the 3<sup>rd</sup> era of computing (Marc Weiser 1991), computers have become smaller and distributed into environment. In addition to computers, resources are also widely distributed into environment. Mobility provides a great advantage to reach these distributed resources.

Because the proposed model offers a context aware system for Ubiquitous Computing environments, smart mobile devices (PDA) that are planned to be used in this model allow creating contexts such as location, time, subject etc.

#### **3.4.1. Hardware**

Mobile devices that are used in the proposed model are equipped with functional hardware components. First of all, they have Global Positioning System (GPS). This feature enables to detect location of the user and create location context for the system. The device can also use Assisted GPS by using not only satellite signals but also network cellular signals to find locations faster.

Other hardware components of the mobile device are wireless connection adaptors which are Wireless LAN (WLAN) and Bluetooth adaptors. This equipment is essential for wireless connection to the gateway. To reach a local resource, the user needs to get authorized by the system. For authorization, the user needs to send his/her certificate after connecting to the gateway wirelessly. The adaptor type can vary, according to the system design, WLAN or Bluetooth connections are preferred.

#### **3.4.2. Software (Application)**

The mobile device mainly accomplishes its tasks by software (Mobile Application) installed on it. The application has the main role of mobile component in the proposed system.

After successful wireless connection to the gateway of resource, the user performs all tasks via this application. First, the user needs to have a certificate to get authorized. The certificate can be possessed in two ways, either downloading from home domain via Internet or directly transferring into the device's internal storage.

Second step of the application provides certificate selection capability to the user in case the user has more than one certificate. In this step, the user selects the certificate to be used in authorization from device storage. In the final step, the requested resource/service type is selected and then the application merges certificate and resource information as an envelope in the background.

After sending the request, the response of the gateway is received and shown via the application. Activity diagram of the process is illustrated in Figure 3.5.

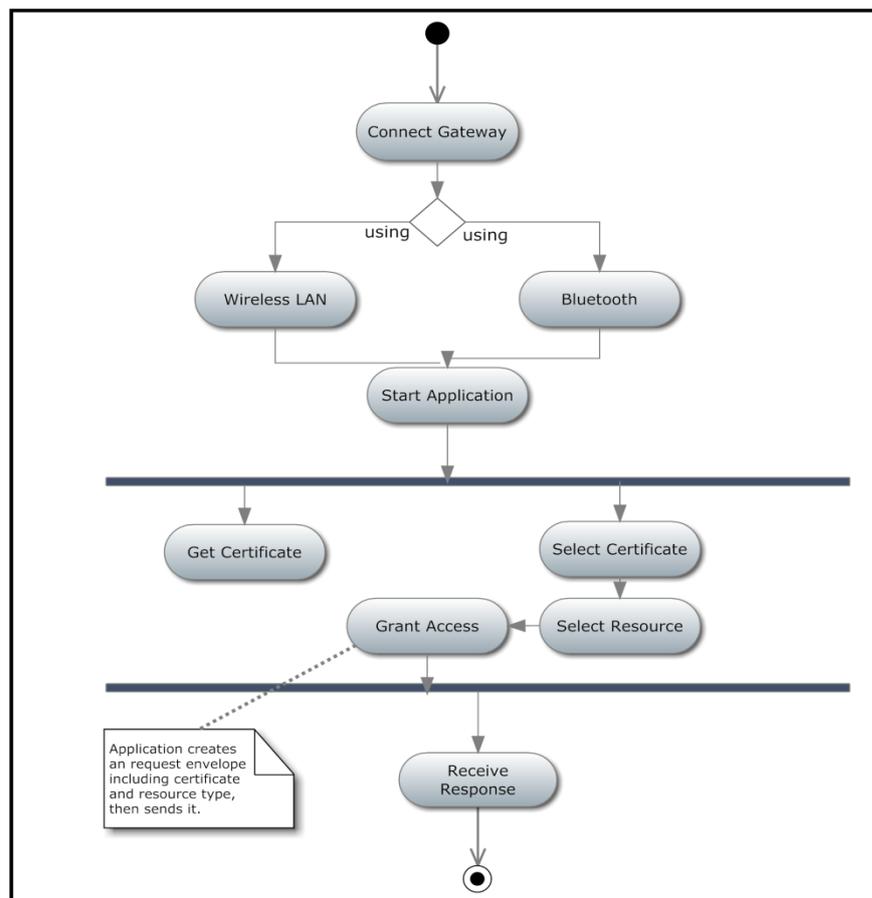


Figure3.5 Activity Diagram of Mobile Device

### **3.5. Components of Gateway**

Gateway is the central unit of the proposed model; it performs the most important required tasks. Mainly, the gateway is in communication with both mobile device and resource/service. Therefore, it receives access requests from mobile devices and data from resource/service. Gateway accomplishes its basic functions via associated sub-components. They are named as: Certificate Service, Context Engine, Decision Engine, Database Service and Management Panel. Actually, the Database Service provides a communication infrastructure for all domains.

In this part of the chapter, Certificate Service, Context Engine, Decision Engine and Management Panel modules will be explained.

#### **3.5.1. Interactions of Components of Gateway**

Each component of the gateway has different duties to complete in the proposed model. However, their cooperation results in more correct decisions and results than working separately. The Decision Engine is the central component of the Gateway and has an interaction with the Context Engine and Database Service. It uses them for collecting required information about the user and the requested resource for evaluation process. The Certificate Service and Context Engine use the Database Service in order to query needed information from the system database. Also the Management Panel has an interaction with the Context Engine and the Database Service for add, delete, update operations of rules and users' records. Interaction of these components of the gateway is illustrated in Figure 3.6.

#### **3.5.2. Certificate Service**

Proposed model provides a certificate based authorization control mechanism when accessing local resources or services. Clients that use the system need to have a valid certificate from a trusted certificate provider. System checks certificate owner ID and certificate provider ID during request evaluation process.

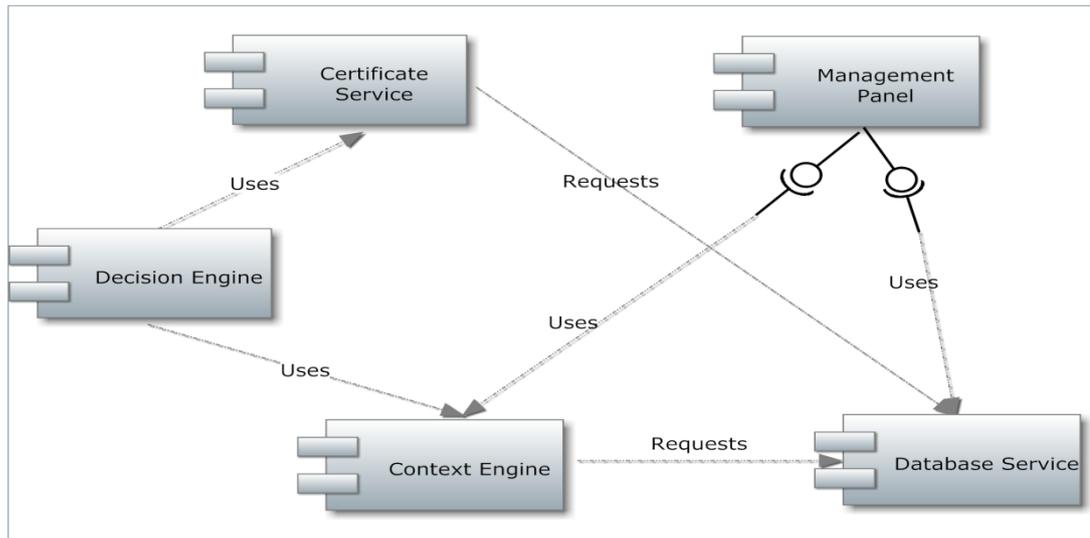


Figure3.6 Interactions of Components of Gateway

Certificate Service has two main tasks in the proposed system. Firstly, it checks certificate validity according to some pre-defined metrics by system administrators. Secondly, Certificate Service performs process of synchronization of domains' active certificate lists. This operation is needed, because there is a possibility that a certificate can be valid for user and provider, however, this certificate can be cancelled by domain administrators for any kind of reasons. In this type of situations, other domains should be aware of these cancelations in order to restrain access of owners whose certificate is cancelled by providers.

### 3.5.2.1. Validity Check of Certificate

Validity check of certificate process contains two steps. These steps are; checking basic parameters of certificate such as validity time period etc. and second step checks certificate provider whether it is trusted and known provider or not and certificate integrity.

- In the first step, the Certificate Service parses certificate data that is sent by the Decision Engine after user access request. It first checks user and issuer sections of certificate and then checks certificate time period (not after and not before) sections and compares it with the current time. If certificate passes this validity check; it is forwarded to the second step.
- Certificate provider validity is checked in the second step of certificate validity check process. Certificates have some encrypted data coded with the provider's private key.

Encoding operation of this coded data with public key of provider needs to result in the same data as provided inside the certificate. This process guarantees certificate provider's validity. Also in this step, certificate integrity and certificate's owner (user) validity are checked in order to prevent certificate modification and prevent usage of another user's certificate with coding users private key and decoding with public key that is provided with certificate. (Jim DeRoest 1997).

Proposed model primarily focuses on designing an access control mechanism using certificates in pervasive environments and checking certificate validity. However checking certificate integrity processes are another research and implementation area. Hence, these integrity control processes are out of the scope of this study.

### **3.5.2.2. Synchronization of Domains' Certificate Lists**

In the proposed model, Certificate Service performs a synchronization task in a pre-defined time period in order to make all agreed domains' active certification lists up-to-date. Each domain should be aware of certification cancellations in order to prevent access of unauthorized users into local resources. Synchronization between domains can be done by three methods.

First method provides a pull-based control method for checking cancellation lists instantly and manually. When a user from a different domain requests to reach a resource, Certificate Server asks his/her certificate cancellation status to the user's home domain immediately. This method seems unsuitable for this model, because these instant queries cause massive network traffic and latency.

In the second method, domains store whole active certificate list for other agreed domains and if any certificate is cancelled, it will be removed from active certificate list. Each domain copies others active certificate lists periodically and tries to make it up-to-date. When a request occurred, home domain refers to synchronized home certificate list for validity decision.

Third method is based on broadcasting Certificate Cancellation List (CCL), a similar approach in the offered model by Ahmet Yortanlı (2011). Domains get other domains' Active Certificate List (ACL) and then each domain broadcasts its CCL in some pre-defined time intervals via its Certificate Service. The usage of CCL between domains is illustrated in Figure 3.7.

All three methods have both advantages and shortcomings. First method tries to obtain the status of a certificate instantly. However it may cause communication related congestion over the network. Second method provides whole active certification list (ACL) every pre-defined time periods to all domains, this also causes network load and some list management problems in the home domain. Third method broadcasts certificate cancellation list (CCL) and domains only need to check this list to find out whether a user's certificate is in this list or not.

Using CCL based synchronization method seems to be more suitable for the proposed system. However it also has some problems; if synchronization time periods is too long, this may cause unauthorized user access. Administrator of one domain may cancel one certificate, however, it may still seem to be active in the home domain due to the fact that there is time to the synchronization process. Synchronization time intervals should be set as minimum as possible according to the network communication traffic load.

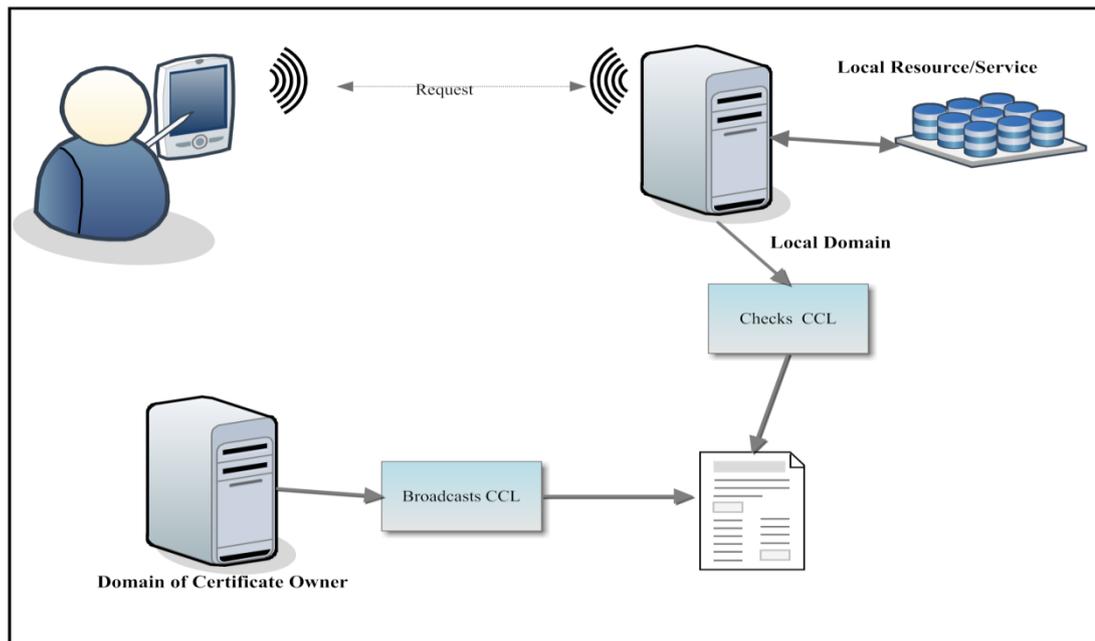


Figure3.7 Certificate Cancellation List Based Synchronization Method

Another problem of the method is that when a request occurs from a different domain user, Certificate Service checks user domain's CCL, this may also cause network traffic. To

overcome this problem, CCL lists of domain can be stored regularly in the home domain. These specifications and functions can be adopted according to the system environment and network conditions.

To sum up, Certificate Service has two major duties. It controls certificate validity when a user wants to access a resource with his/her certificate and also it checks CCLs in order to control certificate cancellation status. Second duty of Certificate Service is to synchronize CCL lists between domains in pre-defined time periods. The activity diagram of these two duties of Certificate Service is illustrated in Figure 3.8.

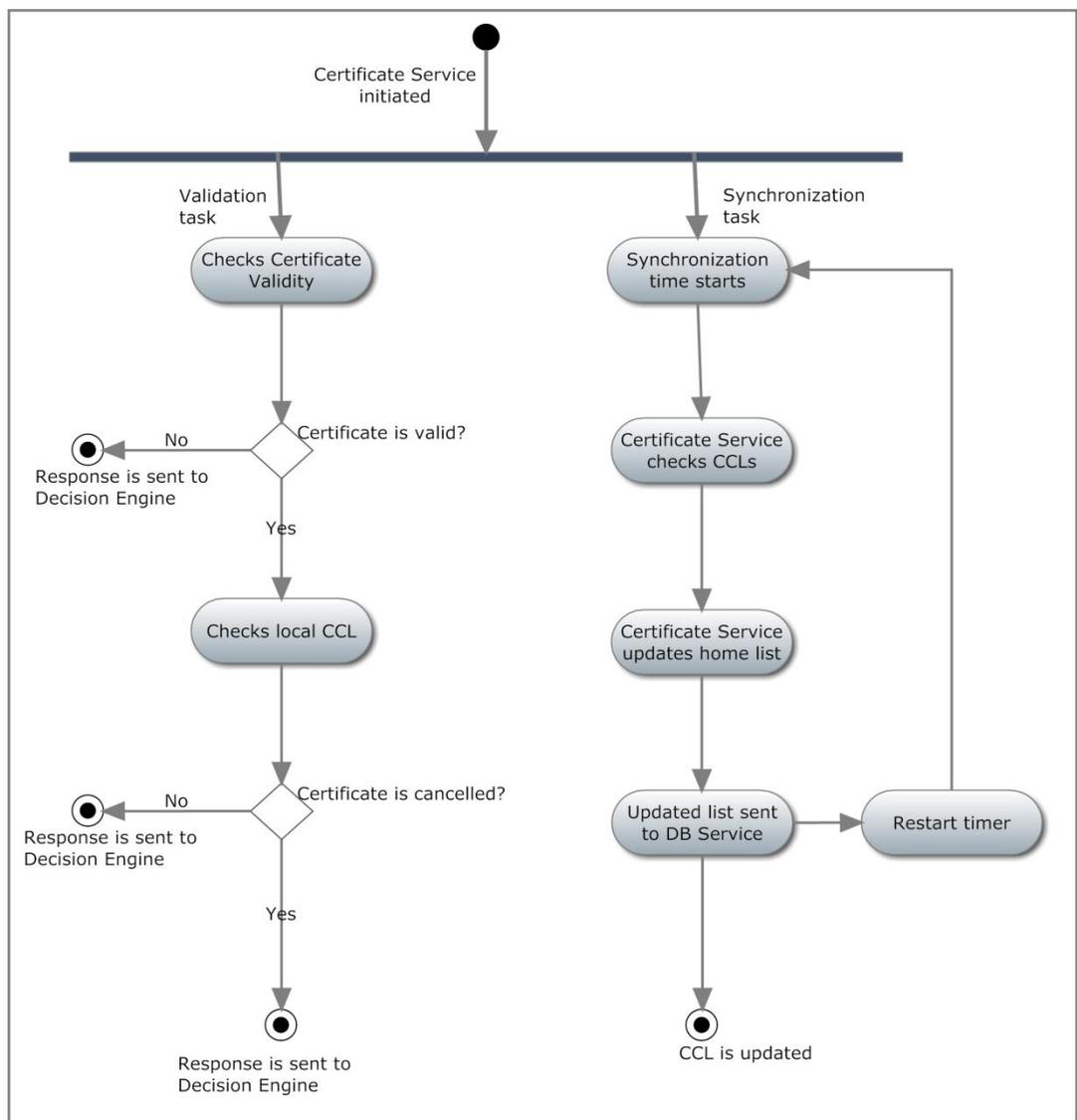


Figure3.8 Activity Diagram of Certificate Service

### 3.5.3. Context Engine

When a client requests an access to a resource, the Decision Engine needs context information about the user and related rules about the requester in order to perform an evaluation and make a decision. The Decision Engine demands this information from the Context Engine.

In the proposed model, the Context Engine has two major duties. Finding, retrieving and matching related context upon request is the first task of the Context Engine, and finding, managing and controlling access rules (AR) about related requests is the second main duty of this component. Components of Context Engine are illustrated in Figure 3.9.

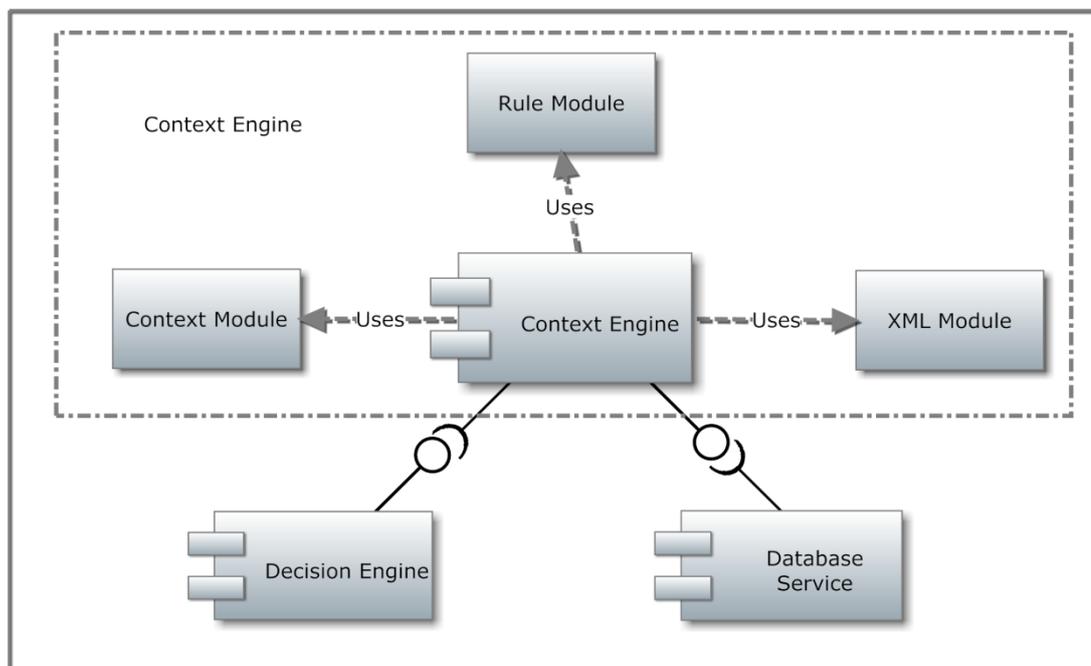


Figure3.9 Components of Context Engine

#### 3.5.3.1. Retrieving Context Information

Since retrieving and matching contexts contain different research topics and studies, these are out of the scope of this thesis. It is assumed that context retrieval and matching are done by the system successfully. However, the model uses location, user and time contexts.

Therefore, the Context Engine sends location and time contexts to the Decision Engine when an access is requested.

The proposed model provides mobile access to resources/services and in order to access a local resource, a user needs to be in range of the resource as location. Otherwise, it is not possible to reach resources. As it is mentioned in this chapter, the location context can be defined by mobile devices' hardware equipment such as GPS or AGPS and sent to the gateway, after that the gateway evaluates this context.

Another method to fulfill location condition is that of examining user location before or while connecting to the gateway. This validation can be done in such ways; when users go to a location, gateway broadcasts a wireless connection as an access point and users can only send access requests as long as successfully connected to this gateway, otherwise it is not allowed to send even an access request. Other way of validation is that; users read Quick Response (QR) codes that are located on the gateways or resources via their mobile phone and mobile devices send QR messages with requests, this method also verifies users' location.

User context information is retrieved from the Certificate Service after certificate parsing and validating process. Time context information is retrieved from gateways' system time.

### **3.5.3.2. Management of Access Rules (AR)**

The proposed model requires Access Rules in order to make decisions toward requests. The Context Engine is responsible for management, retrieving and sending required rules to requester component of the proposed model.

The system stores Access Rules (AR) in a database and the Context Engine administers coordination of the database and system modules. System Administrator defines ARs and adds them into the database. When a user requests an access, the Decision Engine asks for the related rules from the Context Engine, then the Context Engine gets ARs using the Database Service. After getting ARs, it controls requested rules and sends them back to the Decision Engine.

In the proposed model, ARs are transferred between the Decision Engine and the Context Engine in Extensible Markup Language (XML) format. XML is a platform and

programming language independent notation format, therefore, this usage provides flexibility for future module addition and deletion or structure change.

When a rule is requested by the Decision Engine, the Context Engine receives user's certificate/provider and resource id and then, it queries related rules with these identifiers. The Database Service finds and sends all related rules to the Context Engine. Sent rules are checked by the Context Engine and they are converted into XML structure. After that, XML based rules are sent to the Decision Engine for evaluation process.

Examples of Access Rules are given below:

Example 1:

```
<Access Rule>
<subject type ="certificate_id"> davut </subject>
<resource type ="resource"> all_sensors </ resource >
<context type ="time "> all_times </ context >
<decision > allow_send_data </ decision >
</Access Rule>
```

Example 2:

```
<Access Rule>
<subject type =" certificate_provider "> METU </subject>
<resource type ="service"> II/printers </ resource >
<context type ="time "> week_days </ context >
<decision >allow</ decision >
</Access Rule>
```

### Example 3:

<Access Rule>

<subject type ="certificate\_id"> serhat </subject>

<resource type ="resource"> light\_sensor </ resource >

<context type ="time"> evenings </ context >

<decision >allow\_send\_data</ decision >

</Access Rule>

ARs have four data sections. These sections are decision, context, resource and related subject.

Decision section of AR states the behavior type of the system when a user requests an access with stated context and resource type in access rule. Decision style can be as allow/deny format. In printer usage case; access rule indicates in what conditions printer usage is allowed or denied. However if the requests are for resources such as sensor data instead of printer usage, decision can be defined as “allow\_send\_data” meaning that access is allowed and requested data will be sent to the user. With this logic, decision type can be adapted to the designed system.

Context section can carry two different types of data; these are “location” and “time”. Because proposed model provides mobile access to resources, location context filtering can be used in case of having more than one resource in different locations under one gateway system, for example: “first\_floor\_printer” or “outside\_temperature\_sensor”. Time context type specifies when this access is applicable.”all\_time” can be defined for applying access rules for all times, “evenings” or “weekdays” can be other alternatives for time contexts. Date and time format and parsing procedure are explained in the (Oracle, 2010) in detail.

Resource type is defined for both users and groups. For a file access control model, resource types can be defined as “all\_files”, “documents”, “word\_files” etc. or for a sensor data access control model, resource type can be defined as “all\_sensor\_data” or “temperature\_data”.

Subject fields represent owner or member of owner group of request. Like as the resource section of AR, subject type can also defined for just a user or user group. For all students of a university, students from a defined department or a single user, different access rules can be defined by the system administrator.

#### **3.5.4. Decision Engine**

The Decision Engine is the central and coordinator component of the proposed model. Access requests are collected by the Decision Engine and after an internal evaluation process, decision is sent over the Decision Engine again.

When a client wants an access, a request envelope is created with certificate data and resource type and sent to the Decision Engine via smart mobile device by the user. After getting the envelope, the Decision Engine opens it and starts evaluation process. To reach a decision, the Decision Engine needs required information about the request.

First, certificate data is sent to the Certificate Service for validity and cancellation check. The Certificate Service first controls certificate's validity by checking its valid time periods, subject and provider ID. If certificate passes first validity check, it is forwarded to cancellation check. If it cannot pass first control, access request is not allowed without any necessity to check and evaluate other process. After first validity control, Certificate Service checks Certificate Cancellation Lists (CCL) in order to determine certificate's activeness status. Finally, the Certificate Service creates a response about the certificate and sends it to the Decision Engine.

The Decision Engine receives certificate status response, if it is a valid certificate; Decision Engine demands context and rule information about the request from the Context Engine. First; related context information such as time and location about the request is retrieved and sent to the Decision Engine. Context Engine also tries to find the most suitable rule for the request. It queries desired rule by using subject id, resource id information via the Database Service. Related rules are sent to the Context Engine and it selects the most suitable rule. In case of any conflicts in rules that are applied in the evaluation phase, the Context Engine sends only one rule to the Decision Engine.

After retrieval of all these required information, Decision Engine generates a decision about the request and sends it to user.

Components of the Decision Engine are illustrated in Figure 3.10.

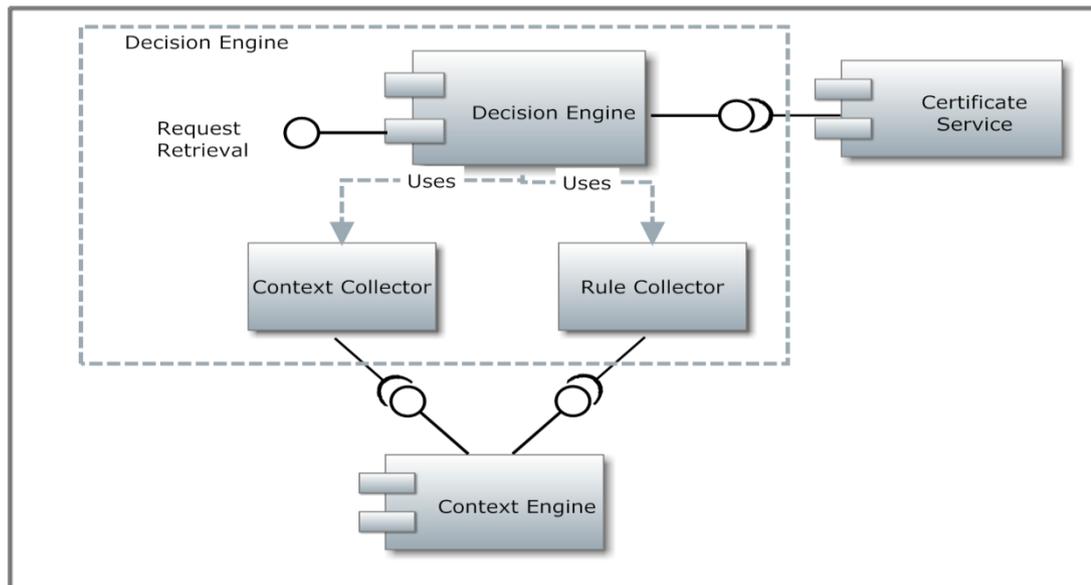


Figure3.10 Components of Decision Engine

#### 3.5.4.1. Permission Evaluation Method

Decision Engine evaluates requests of users with the consideration of collected rules and contexts. Steps of decision methodology are;

1. It is supposed that; Context Engine sends most suitable one rule to Decision Engine in order to avoid conflicts. If there is a “deny” rule among queried rules, it has superiority over other “allow” rules for the same user and same contexts.
2. If there is no rule for requested access, Decision Engine sends “deny” response to the user.
3. If rule has time “deny” definition for current time context, Decision Engine sends “deny” response to the user.

4. If rule has time “allow” definition for current time context, however it has location “deny” definition for current location context, Decision Engine sends “deny” response to the user again.
5. If rule has “allow” definition for both time and location and if these parameters are “true” for current time and location context, Decision Engine sends “allow” response to the user and performs required operations.

The activity diagram of Decision Engine is illustrated in Figure 3.11.

### **3.5.5. Management Panel**

The Management Panel provides a graphical interface to the system administrator for performing management duties. The system administrators mainly carry out add/delete/update operations for the system parameters.

First, the system manager can add/delete/update Access Rules (AR). After rule definition via the Management Panel, the Context Engine converts rules into XML files when they are requested, however administrator defines rules via standard graphical interface instead of XML based.

The administrator also can define the system user groups for both user and rule context. Resource types and resource groups can also be defined by system managers using the Management Panel.

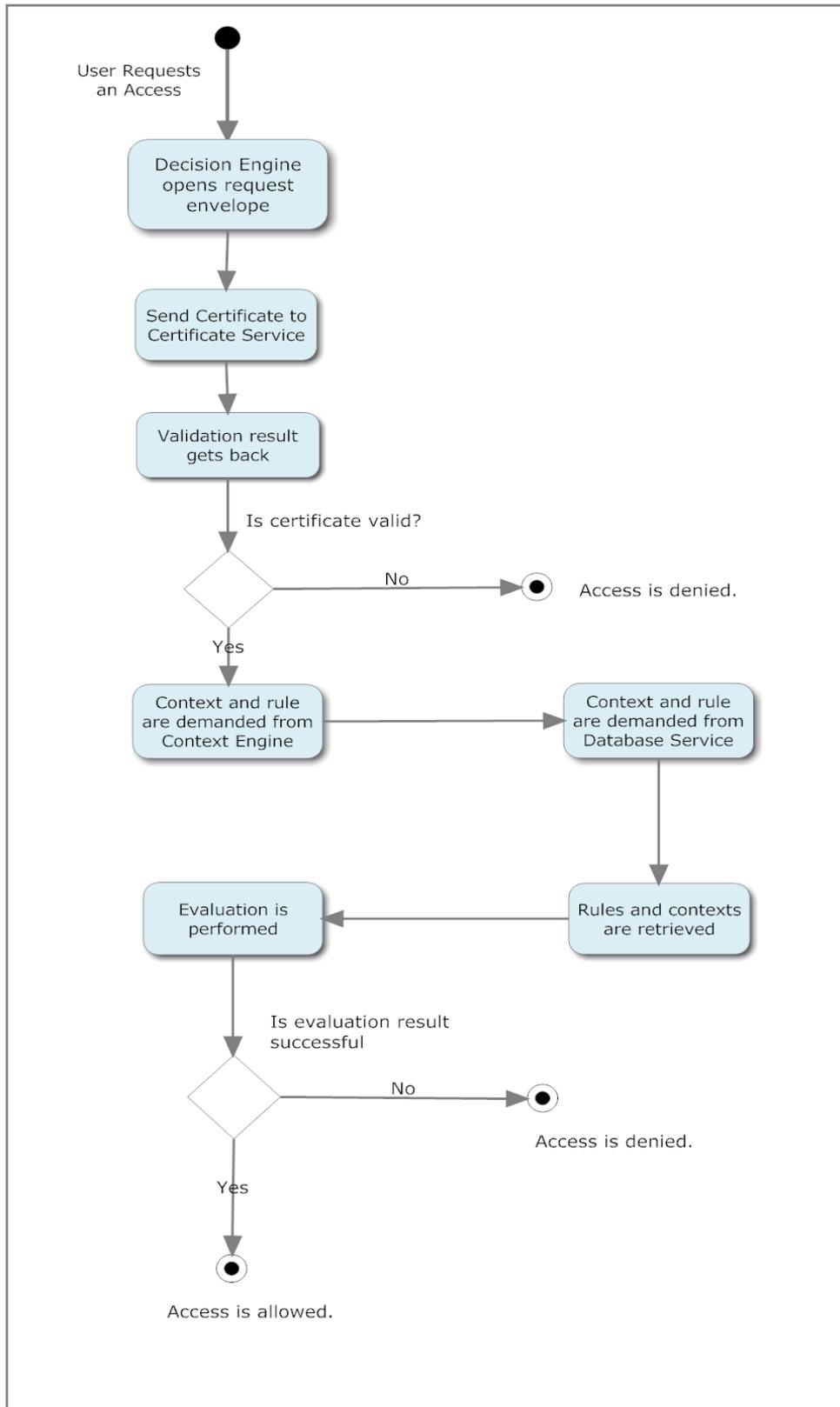


Figure3.11 Activity Diagram of Decision Engine

## **CHAPTER 4**

### **PROTOTYPE IMPLEMENTATION**

In this chapter, the prototype implementation of the certificate based authentication control model via smart mobile devices for ubiquitous environments will be described. The conceptual description of the proposed model is presented in Chapter 3 and the real prototype application will be described and explained in this chapter.

First, components of the prototype will be described, and then the activity flow will be analyzed. After that, the system design will be explained; finally the prototype scenarios and cases will be introduced in this chapter.

#### **4.1. Description of Prototype Implementation**

In order to show feasibility of the proposed model and demonstrate its applicability, a working system is developed based on the proposed model. The prototype mainly consist of an Android based mobile application that works on a mobile device, a gateway software that works on a personal computer and temperature/light sensors that work on an electronic board.

In the prototype, the mobile application represents mobile domain of the proposed model, software installed computer represents the gateway of the proposed model and temperature/light sensors represent the resource/service domain of the proposed system.

For gateway component of the proposed model; a model titled as “A Certificate Based, Context Aware Access Control Model” (Yortanli, 2011) has been offered and our study

uses this conceptual idea offered by Yortanlı (2011) only in the gateway component, however, main focus and major contribution of our study and prototype are; unlike the model offered in Yortanlı (2011), using mobile devices for user requests based on developed mobile application and using temperature and light sensors' real data as local resource. With the nature of ubiquitous computing, mobility is one of key parameters for designing future systems; therefore, this study mainly focuses on proposing an access control system based on smart mobile devices. In brief, this study contributes and creates a major difference on showing its applicability and converting the conceptual model into a real prototype implementation by adding mobile device enhancement on access requesting and receiving service by it, and also using sensors as resources.

The prototype implementation is named as “ARS” referring to the abbreviation of “Access Resource System” and the main steps of ARS involves the following: temperature and light sensors send their data to the gateway periodically, the gateway stores in specialized format for submitting them to authorized users. A user comes close to the local gateway of a resource and connects to the WLAN that the gateway broadcast as an Access Point. The user should connect to this WLAN; otherwise it is not possible to send any access request and receive any responses from the gateway. This process also gives an idea about the user's current location and ensures that the user is close to the local resource. After connecting to the gateway, the user selects a certificate from the internal storage of the mobile device and sends it to the gateway with resource type using the mobile application. The gateway basically controls certificate validity and collects required information about the request and performs an evaluation. These implementation steps will be described in detail in this chapter.

## **4.2. Components of Prototype**

The prototype implementation based on the proposed model includes 3 main components. Android based mobile domain including mobile application, gateway software and sensors located on electronic board are main components of the prototype implementation. There are also sub-components under these main components. Components of the prototype are illustrated in Figure 4.1.

### 4.2.1. Mobile Device

The proposed model includes mobile devices for connecting to local resource and services. In the prototype application Android Operating System running smart mobile device is used for sending access requests and receiving response.

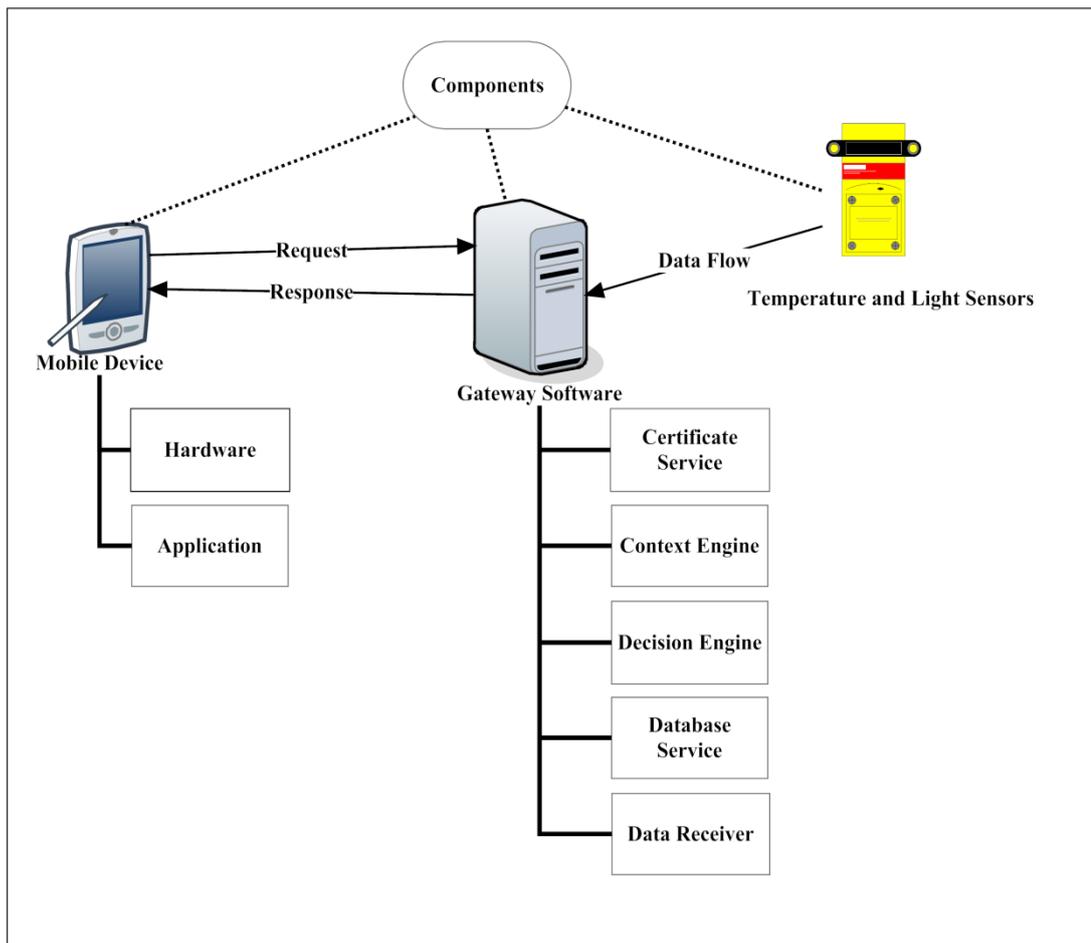


Figure4.1Components of Prototype

Smart mobile devices also can be named as Personal Digital Assistant (PDA) having sophisticated hardware components such as sensors, wide screen, GPS module, WLAN adaptor, camera, Bluetooth adopter etc. Because of these capabilities, usage of these devices provides convenience and functionality, and also increases the system's flexibility in design.

The critical hardware component of the mobile device used in the prototype is the WLAN adaptor. The gateway broadcasts wireless connection and the device establishes connection to this wireless network. The user should connect wirelessly to the gateway via the broadcasted connection, if it is not connected or tries to connect to another connection, access requests cannot reach the gateway.

#### 4.2.1.1. Android Operating System

Android is an operating system that is especially designed for mobile and smart devices. Google Inc. purchased Android Inc. in 2005 and started Open Handset Alliance together with software, hardware and telecommunication companies in order to define open standards for mobile devices. After that, Google extended mobile operating system studies with Android Open Source Project (AOSP) and released first versions of Android under free and open software license. Also Google reveals whole source code under an Apache (Licence, 2011). Structure of Android OS is illustrated in Figure 4.2.

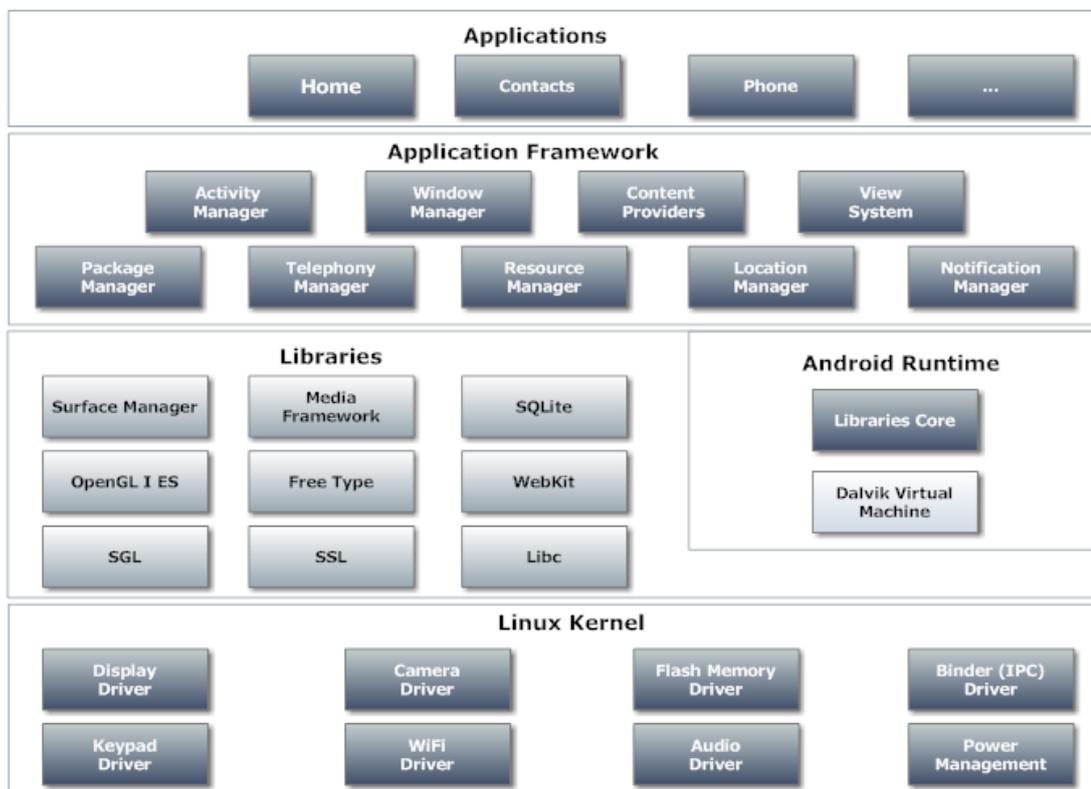


Figure4.2 Structure of Android OS

Android OS is developed over Linux kernel including hardware drivers such as display driver, Wi-Fi driver, camera driver etc. Libraries layer lies on the kernel layer; they are written in C programming language and include mainly SSL, SGL, and SQLite etc. Android Operating System includes Dalvik Virtual Machine working as a compiler and debugger, it allows to run compiled Java based application and codes. Application layer is located on top of libraries and is composed of framework and application.

#### **4.2.1.2. Basics of Android Mobile Applications**

Before explaining the prototype mobile application, a brief outline of Android mobile application basics is analyzed. In Android OS, applications mainly consists of four components, they are; Activity, Service, Broadcast Receiver and Content Provider. Activity component includes user interfaces of application such as login interface and intents that perform required actions such as opening a web page or calling a number. Service means running background services for application. Broadcast Receiver retrieves broadcasts from system such as battery status warnings. Content Provider component gets required contents for applications such as contact information from phonebook.

Graphical interfaces of applications are XML based configuration. In other words, codes and interface elements can be separated and coded, this provides convenience and functionality. Basically, graphical elements are defined in “main.xml” document and called from during coding. An example of graphical element definition in main.xml:

Example:

```
<TextView
android:id="@+id/welcome"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:textColor="#000000"
android:text="Welcome to Resource Access System! ">
</TextView>
```

Also value definitions are stored in strings.xml, such as application name, any string definitions or values of array elements are defined in this file. An example of array value definition:

Example:

```
<array name = "spinner_deger">
    <item>temp</item>
    <item>light</item>
</array>
```

The applications developed for Android OS are written in Java programming language basics, however, some special classes and methods definitions are also included. Owing to open source convenience, applications can reach mobile device hardware units such as Bluetooth, internal storage etc. however, these permissions should be defined in “AndroidManifest.xml”, an example definition is:

Example:

```
<uses-permission android:
name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.INTERNET"/>
```

Android mobile applications become Android Package File (APK) format and all related files codes, pictures, manifest etc. are accumulated into one APK file after compiling. Because of that, APK files can be moved, copied and installed into Android based mobile devices as long as their Android Software Development Kits (SDK) meets minimum SDK level of application.

#### **4.2.1.3. Mobile Application (ARS)**

For the prototype implementation, an Android based mobile application is developed for connecting to the local resource of the gateway, sending access request and receiving gateway response. The minimum Android SDK level for the application is 2.1 (Éclair).

The application has mainly four user interface screens. The program opens with the login page which is illustrated in Figure 4.3. The credentials for the login page do not affect authorization process; they are only for application usage and obtained from the domain administrators. After logging in, there are 2 buttons in the second interface; user can download his/her certificate from home domain web site and select certificate from mobile device. When user clicks “Select Certificate” button, application opens file browser and allows browsing the device’s internal storage. After selecting related certificate, application shows certificate content on the screen if it is available, then it returns to the second interface and creates an information text about which certificate selected. If correct

certificate selected, the user goes to the final stage by clicking “Next Step”. On the final stage, application shows available resources for the connected gateway; in the prototype it shows “METU/II/Wireless Lab/Temperature-Light Sensors”. The user then selects “Temperature” or “Light” from dropdown menu that is also named as “Spinner” in Android programming. Finally an access request is sent via “Grant Access” button and the response is shown in the result section of interface. These activities are described in detail and the screenshots are shown in this chapter.

The application converts access requests into Simple Object Access Control (SOAP) envelope messages. SOAP is an XML based messaging structure for communicating web services based on Web Services Description Language (WSDL). SOAP protocol has versions of 1.0, 1.1, and 1.2; in the implementation version 1.1 is preferred for stability. Because of platform independent convenience and flexibility for modularity; SOAP messaging technique is used in the prototype implementation. Simple examples of SOAP request and response envelope messages are:

Request Message:

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Header />
  <env:Body>
    <isUserSuitableToAccessResource xmlns="http://service/">
      <certificateData xmlns="">davut</certificateData>
      <resourceId xmlns="">temp</resourceId>
    </isUserSuitableToAccessResource>
  </env:Body>
</env:Envelope>
```

Response Message:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:isUserSuitableToAccessResourceResponse
xmlns:ns2="http://service/">
      <AccessRequestResult>deny</AccessRequestResult>
    </ns2:isUserSuitableToAccessResourceResponse>
  </S:Body>
</S:Envelope>
```

Android SDK does not have a built in SOAP library. In order to use SOAP messaging protocol, Ksoap2, an external SOAP library provided by (Google, 2011) is included in the

prototype implementation. SOAP envelope creating and sending in the prototype is done using this library and an example from the prototype is:

#### SOAP Message Enveloping and Sending:

```
private String isUserAuthenticated(String certificateData, String
resourceId) {
SoapObject Request = new SoapObject(NAMESPACE, METHOD_NAME);
    Request.addProperty("certificateData", certificateData);
    Request.addProperty("resourceId", resourceId);

SoapSerializationEnvelope soapEnvelope = new
SoapSerializationEnvelope(SoapEnvelope.VERSION1);
soapEnvelope.setOutputSoapObject(Request);
HttpTransportSE aht = new HttpTransportSE(URL);
    try {
        aht.debug = true;
        aht.call(SOAP_ACTION, soapEnvelope);
    }
    catch (IOException e) {}
    catch (XmlPullParserException e) {}
    String result = null;
    try {
        result = "Gateway Response:"
            + (SoapPrimitive)
soapEnvelope.getResponse();

    } catch (SoapFault e) {
        e.printStackTrace();
    }
    return result;
}
```

As it is seen from the code piece, a soap object is created with namespace and method name attributes, then two properties “certificate” and “resource type” are added to the object. A version 1.1 soap envelope is created with this soap object. In order to send this envelope, an http transport object is created with the gateway URL and then the envelope is sent using http transport protocol. Finally the message result is retrieved and shown by the application interface.

Interfaces of the mobile application and general flow of the prototype are described in the activity flow related section in this chapter in detail.

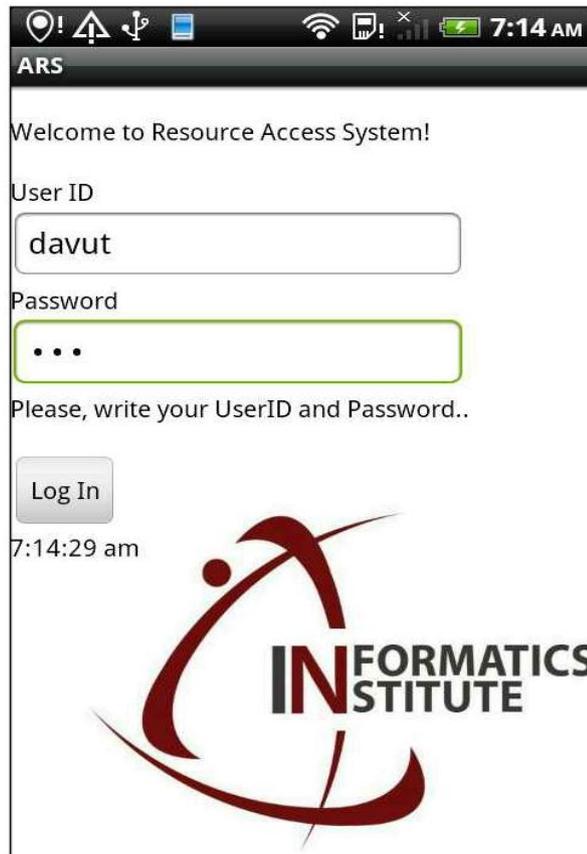


Figure 4.3 Login Page of Mobile Application of Prototype

#### 4.2.2. Gateway

In the prototype implementation, a personal computer is used as hardware and software developed in Java Enterprise (J2EE) programming environment runs on the computer. The computer has mainly WLAN adaptor for broadcasting wireless connection and serial ports for sensor serial connections.

The user should connect to the gateway wirelessly in order to send access requests and receive responses. After connecting successfully, mobile device gets an Internet Protocol (IP) address from system. Gateway runs Dynamic Host Configuration Protocol (DHCP) in order to control and distribute (IP) addresses of connected devices. Gateway takes 192.168.178.1 IP address and DHCP distributes 192.168.178.x IP addresses (192.168.178.2-

192.168.2.254) to devices. Hence, gateway supports 253 connected mobile devices in the prototype implementation.

Conceptual description of the gateway software is described in Chapter 3 and gateway software of the prototype implementation is developed based on an offered certificate based access control model (Yortanli, 2011), however, the major focus of this study and prototype is that; enhancing offered model by adding new improvements and finally developing a new framework that uses mobile devices for access control and real sensors data as local resource. The most significant contribution of this study relies on developing a real implementation based on the proposed model to show its applicability and feasibility.

The software in the gateway consists of five modules and these are Certificate Service, Context Engine, Decision Engine, Database Service and Data Receiver. Except the Data Receiver, other modules communicate with each other, and also with the mobile device using web services based on Web Services Description Language (WSDL) structure.

WSDL provides an XML based web service description. In the prototype 1.1 version of WSDL is used and main elements of a WSDL messages are; “Service”, “Port” that defines address of a web service, “binding” that defines the SOAP structure, “PortType”, “Operation” and “Message”.

The working logic and interactions of software modules are described in Chapter 3 in detail. In brief; the Context Engine is responsible for finding suitable context and rule about the request, the Certificate Service controls certificate validity and performs synchronization between domains, and the Decision Engine is the core module and receives requests, collects required information and makes a decision. The Database Service communicates with modules and retrieves required information from the database. These four modules use web services for interactions, flexibility in adding new modules and allowing changes in modules are other reasons of using web services. In the prototype, the gateway software also has Data Receiver module. Sensors send their data to the gateway via electronic board periodically. They are connected to the gateway with serial connection on a port. Data Receiver listens to the pre-defined serial port and retrieves the sensor data. After that, it parses and interprets temperature and light data separately. These data are saved to a file in the gateway and the Decision Engine sends required data from this file if authorization of the user is successful.

The interaction diagram of modules of gateway software is illustrated in Figure 4.4.

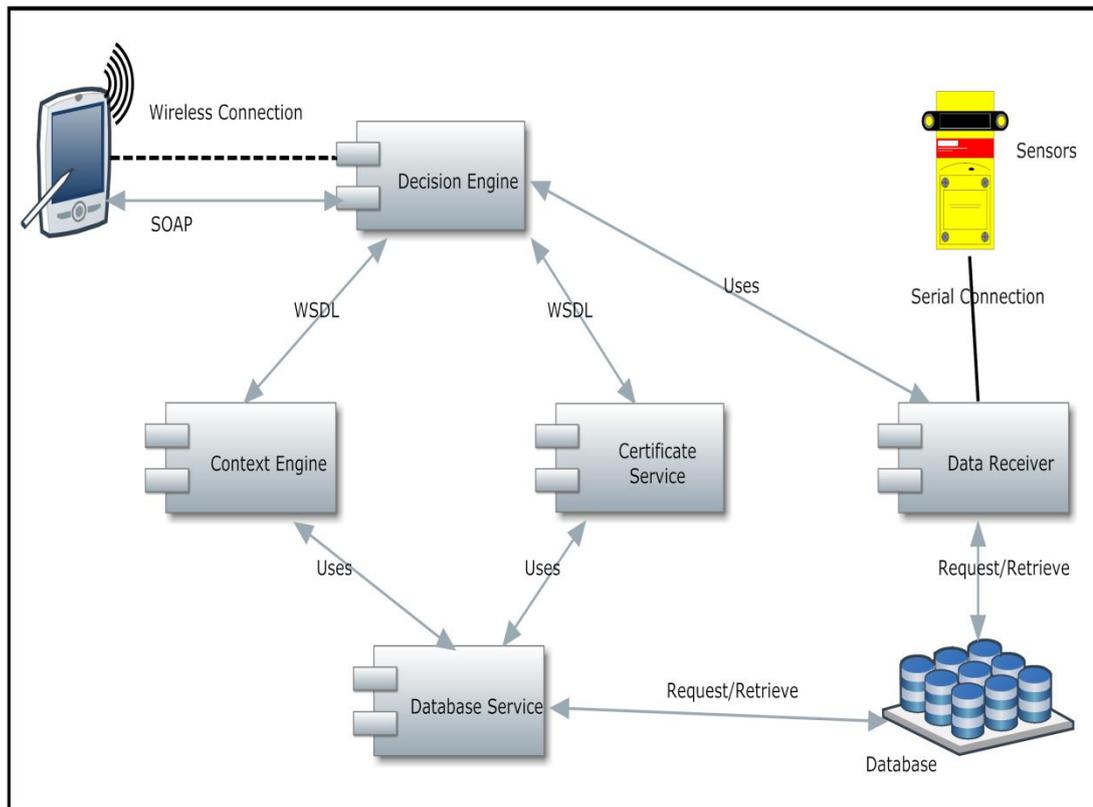


Figure4.4 Interaction Diagram of Modules of Gateway Software

### 4.2.3. Sensors

Two different types of sensors are used in the prototype implementation as local resources. Electronic temperature and light sensors measure current temperature and light values of the environment, respectively. Sensors are connected to an electronic microcontroller (Arduino Uno, 2011) both for getting power and coding environment.

Arduino Uno is an open source electronic prototyping environment supporting various electronic components for developing prototypes. It has 3.3 and 5 Volt power output for power need of hardware components and 14 digital, 6 analog input ports for getting data from installed electronic components. A photo (Arduino Uno, 2011) of microcontroller is illustrated in Figure 4.4.

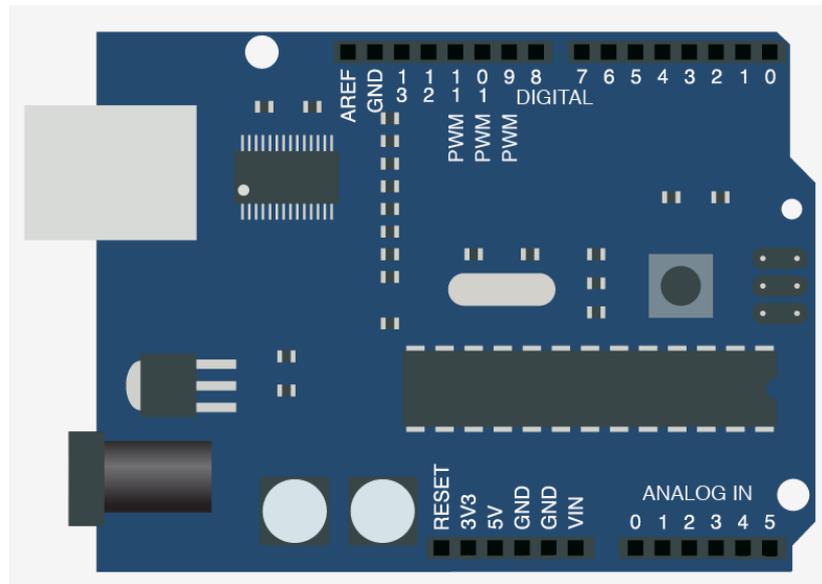


Figure4.5 Microcontroller for Sensor Connections

The microcontroller has a wired connection to the gateway. Connection is interpreted as serial connection by the gateway and the port that it is connected to is named as COMx format. This port name is defined in the Data Receiver module and it listens to that port for data flow.

In order to get sensor data and set other properties about data sending, some codes need to be uploaded to the microcontroller's memory. They are coded in special programming language similar to C/C++ using microcontroller's coding interface. Example of code for reading sensor data in prototype is:

```
void loop() {
  Wire.requestFrom(tmp102Address, 2);

  byte MSB = Wire.receive(); byte LSB = Wire.receive();

  int TemperatureSum = ((MSB << 8) | LSB) >> 4; //it's a 12bit int,
using two's compliment for negative

  float celsius = TemperatureSum*0.0625;
  float fahrenheit = (TemperatureSum*0.1125) + 32;

  Serial.print(" C :");
  Serial.println(celsius);
  int Light = analogRead(A0);
  Serial.print (" L :");
  Serial.println(Light, DEC);
}
```

```
Serial.println ("");  
delay(3000);}
```

Also it has a serial monitor to see output of the code uploaded to the microcontroller. Temperature data is shown as “C:” and Light data is shown as “L:” format in serial monitor and retrieved by the Data Receiver. The output of sensors installed on the microcontroller and used in the prototype via serial monitor is illustrated in Figure 4.6

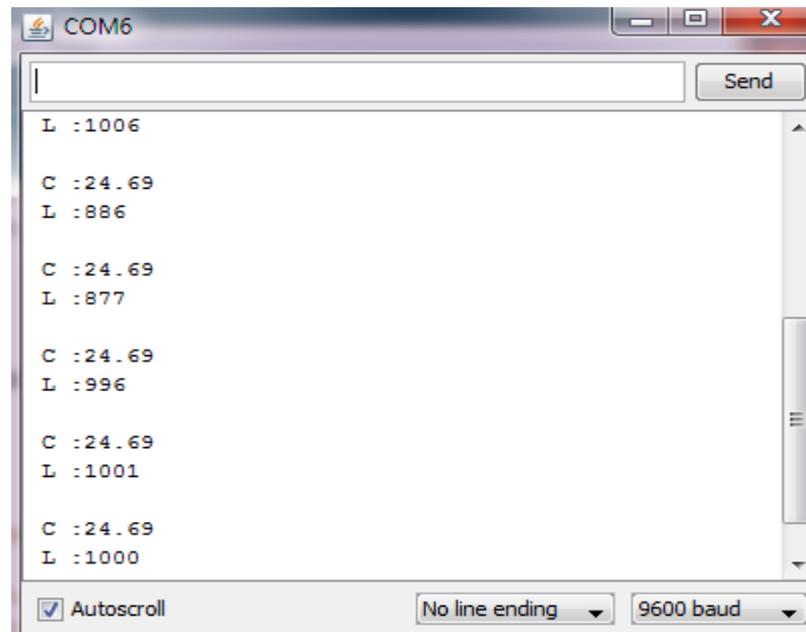


Figure4.6 Output of Sensors in Serial Monitor

#### 4.2.3.1. Light Sensor

Light sensor measures environment's light using internal resistance to light, because of that an external resistor is used in its circuit. It measures light and produces an output integer between 0 and 1024. It senses instant light, in other words it realizes instant light chances in the environment.

The light sensor gets 5 Volt power from the microcontroller and send its data via analog input port, also another pin of sensor is connected to a resistor and circuit is completed in ground port. The schema of light sensor connection is illustrated in Figure 4.7.

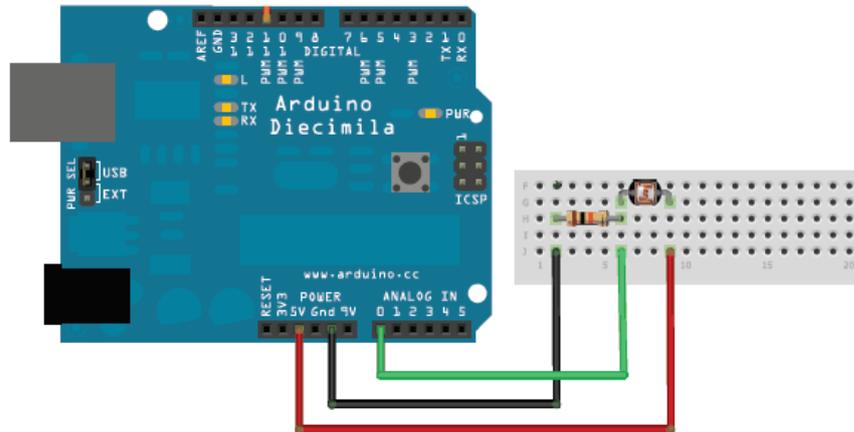


Figure4.7 Schema of Light Sensor Connection

#### 4.2.3.2. Temperature Sensor

Temperature sensor measures ambient temperature of environment. It runs on 3.3 V power and uses two analog input for sending data and two grounds for completing circuit.

The sensor includes Inter-Integrated Circuit (I2C) serial circuit inside of it and with the convenience of I2C, direct temperature values can be read as an output instead of getting analog signals. The schema of temperature sensor connection is illustrated in Figure 4.8.

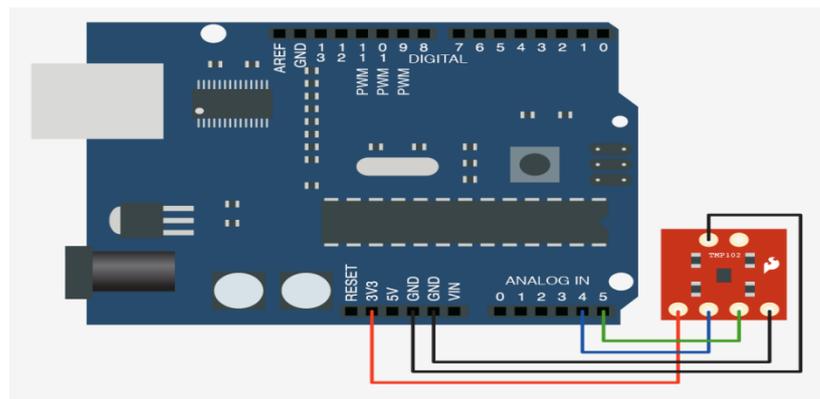
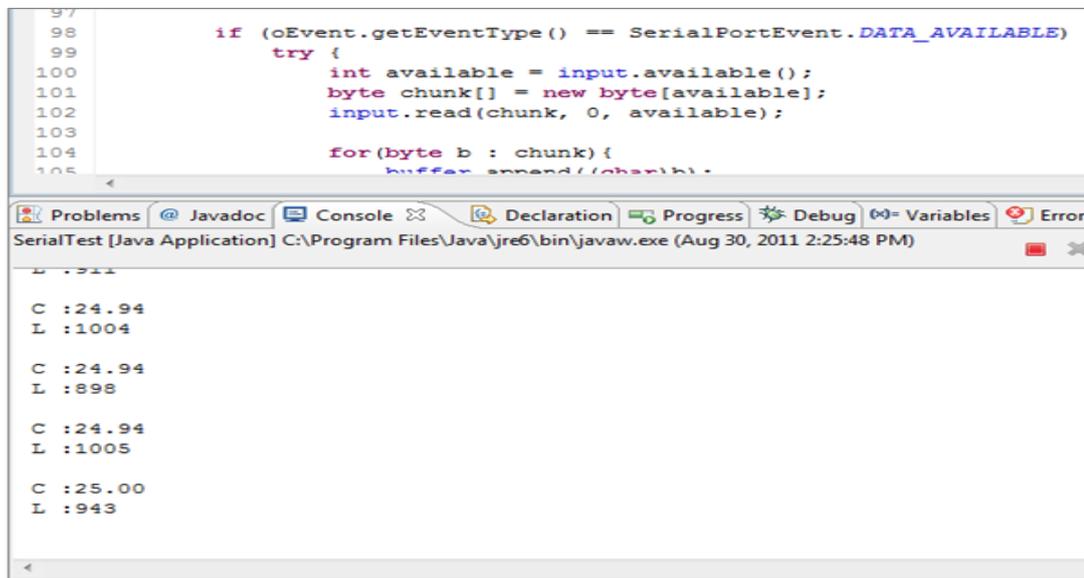


Figure 4.8 Schema of Temperature Sensor Connection

### 4.2.3.3. Sensor Data Retrieving

Microcontroller, together with the sensors mounted is connected to the gateway via a wired connection. The gateway detects it as a serial connection and gives it a serial port number such as “COM6”. This port number is defined in the Data Receiver module of the gateway software. The Data Receiver module starts to listen to this port and after data flow starts from sensors, it detects this data and shows them in the console. After detecting, it writes these values into an external file. These values are parsed and interpreted as two different sensor values when an authorized user wants to get these data. Transmitted sensor data on console is illustrated in Figure 4.9.



```
97
98     if (oEvent.getEventType() == SerialPortEvent.DATA_AVAILABLE)
99         try {
100             int available = input.available();
101             byte chunk[] = new byte[available];
102             input.read(chunk, 0, available);
103
104             for(byte b : chunk){
105                 buffer.append((char)b);
```

SerialTest [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Aug 30, 2011 2:25:48 PM)

```
L : 911
C : 24.94
L : 1004

C : 24.94
L : 898

C : 24.94
L : 1005

C : 25.00
L : 943
```

Figure 4.9 Transmission of Sensor Data

### 4.3. Activity Flow of Prototype

Sensor data retrieving continues regularly as long as sensors work and sense the environment, therefore, this process is independent from user activity flow.

In order to reach sensor values, users need to use mobile application (ARS) on their mobile devices. The Android based mobile device automatically starts installation when apk file is started by clicking on it.

After installation is complete, application starts with login page. The credentials on the login page are not used for authorization; they are only for program usage and can be obtained from domain administrators.

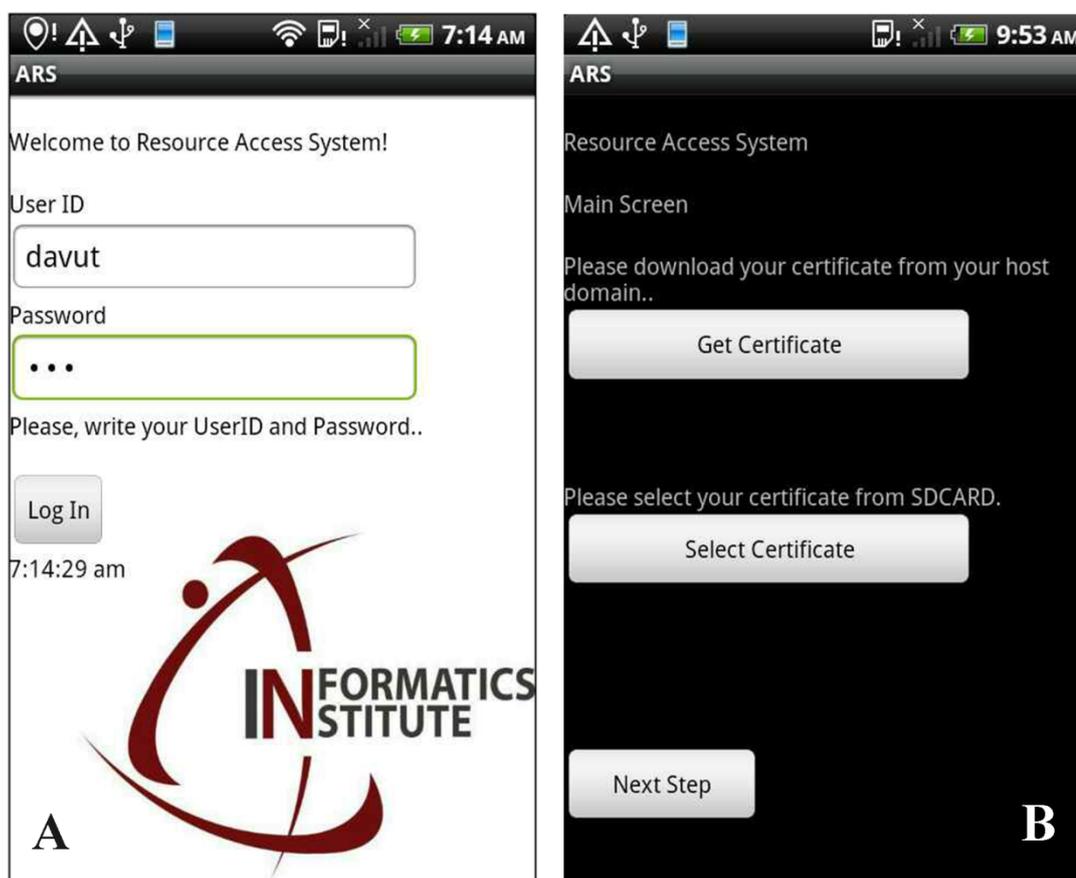


Figure 4.10 Login (A) and Certificate Selection (B) Interfaces of Mobile Application

After logging in successfully, the user is forwarded to the main screen of the application. In this screen, the user performs certificate transactions using two buttons. The first button “Select Certificate” forwards the user to his/her host domain to get a certificate, however, because of unsuitability of home domain for this action, this process is out of the scope of this thesis. The second button “Select Certificate” allows the user to select his/her certificate from the storage of the mobile device. These interfaces are illustrated in Figure 4.10.

The application browses SD Card installed on the mobile device, because external operations generally use SD Card and also Android OS does not allow using interior storage unless the user has root privileges. The user navigates files, also can go up or down within

the folders. In Figure 4.11 six certificates are shown as an example, a user generally has one certificate. After selecting a certificate, the application shows the certificate content for 3 seconds. If it is not possible, it gives an “unable to read file” warning. The certificate content disappears and application creates a text notification about selected certificate and its path on the SD Card. Browsing the SD Card and Certificate content interfaces are illustrated in Figure 4.11



Figure 4.11 Browsing SD Card (A) and Certificate Content (B) Interfaces of Mobile Application

The user proceeds to the final step by clicking “Next Step” and in the final interface: the application shows available resource types. As stated, the user needs to connect to the wireless connection that the gateway broadcasts in order to send access requests. This process also verifies the user location. According to the gateway that is connected, the application shows which resource and resource types are available.

In the prototype implementation, available resource is sensors in the Wireless Lab of Institute of Informatics. This information is shown on the screen and the user selects

temperature or light from drop down menu as resource type. This selection is illustrated in Figure 4.12. With the resource type selection, the application brings together certificate data and resource type and creates SOAP access request envelope. This envelope is sent to the gateway using SOAP mechanism via the wireless connection.

The Decision Engine of the gateway software first receives the envelope and opens it. After opening the envelope, data is parsed, and certificate data and resource type are separated. Certificate data is sent to the Certificate Service for validity check, if certificate can

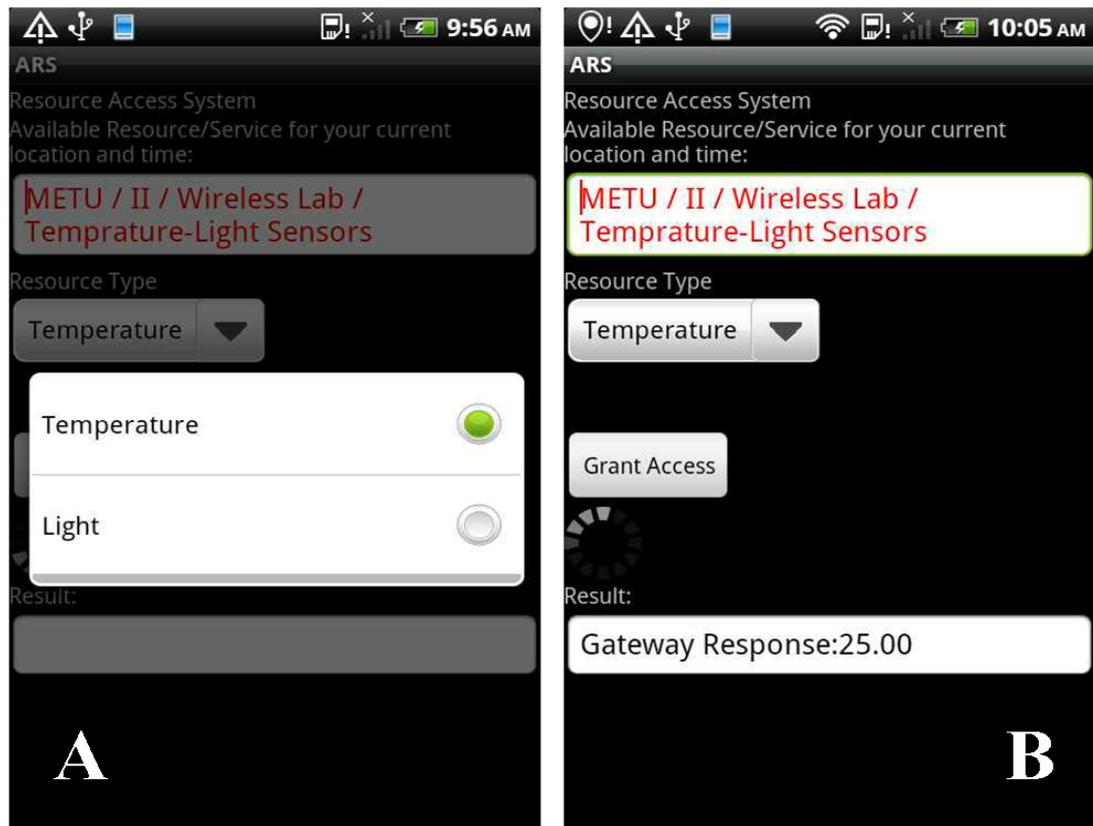


Figure 4.12 Resource Selection (A) and Temperature Sensor Data Response (B) Interfaces of Mobile Application

pass this control, then the Decision Engine demands required context data and rules from the Context Engine. After all required data are collected, the Decision Engine performs an evaluation and makes a decision. If the user is authorized to reach temperature of light sensor data, the gateway sends related data directly to the mobile device instead of sending “allow” information only. If the user is not authorized after the evaluation process, the gateway sends “deny” response to the user’s mobile device.

Temperature sensor data demonstration is illustrated in Figure 4.12; also light sensor data and access denial demonstration are illustrated in Figure 4.13.

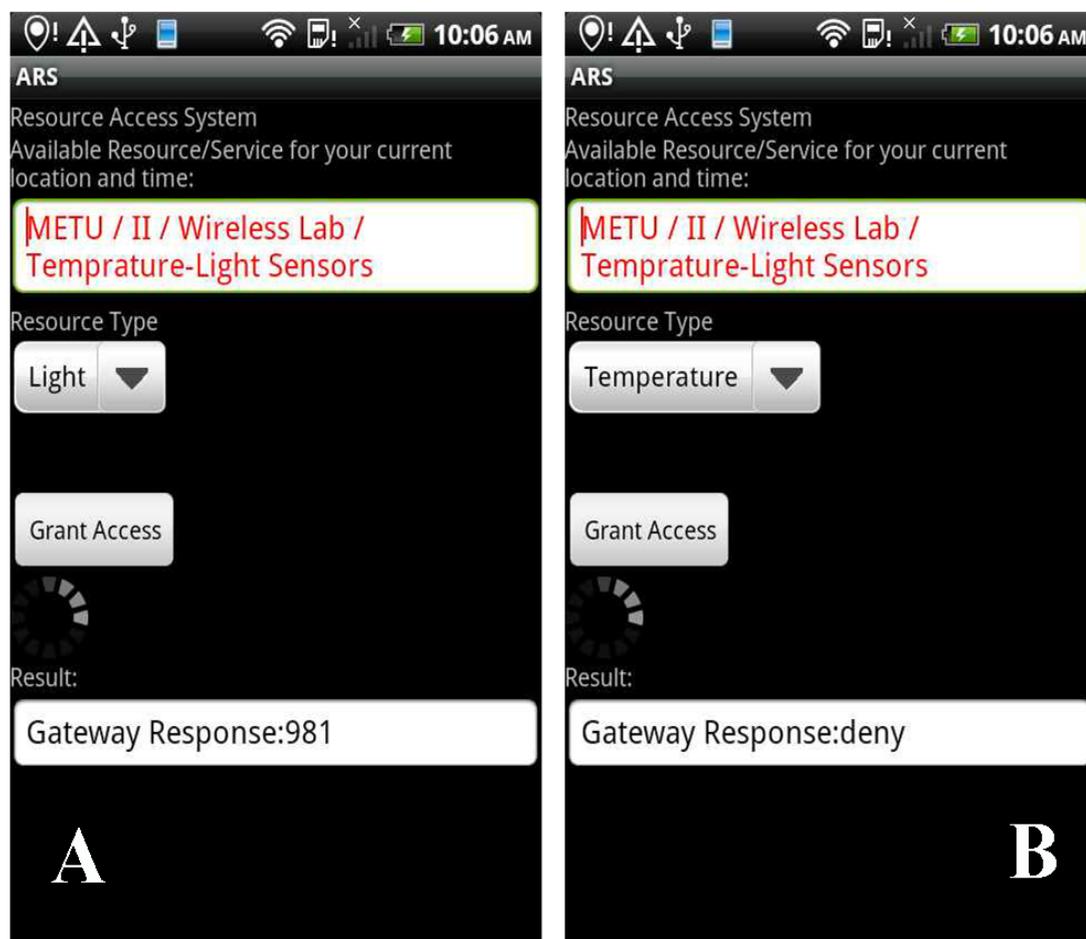


Figure 4.13 Light Sensor Data (A) and Access Denial (B) Interfaces of Mobile Application

#### 4.4. System Design

The prototype system has various components and hardware; therefore, different technologies, methods and programming environments are used in the development phase. The mobile application is developed based on Java; however, Android SDK has also different approaches, methods and structure than the standard Java programming environment. SOAP messaging protocol is implemented for communicating with the gateway, in order to do that an offered external library (Google, 2011) named KSOAP2 is used in the development.

The gateway software is developed based on Enterprise Java (J2EE) methodology. In order to provide module independency, all modules of the software are implemented as web services into the web application server. Oracle Weblogic is preferred as the application server because of its reliability and stability.

In the design of sensors as local resources, a microcontroller platform (Arduino Uno, 2011) is used together with the sensors. Sensors are installed with this microcontroller platform and send their data over it. A special programming language similar to C/C++ is used for coding the microcontroller and setting the sensor parameters.

The Access Control System Software has five modules: Decision Engine, Context Engine, Certificate Service, Database Service and Data Receiver, and also it interacts with four external components. The user being one of main components, sends access requests using the smart mobile device and receives the gateway response. The gateway periodically demands Certificate Cancellation Lists (CCL) from other domains and they send CCLs to the gateway. The sensors send their data automatically without any requests. The System Manager performs rule and user insertion transactions. The level 0 (context) data flow diagram of prototype is illustrated in Figure 4.14.

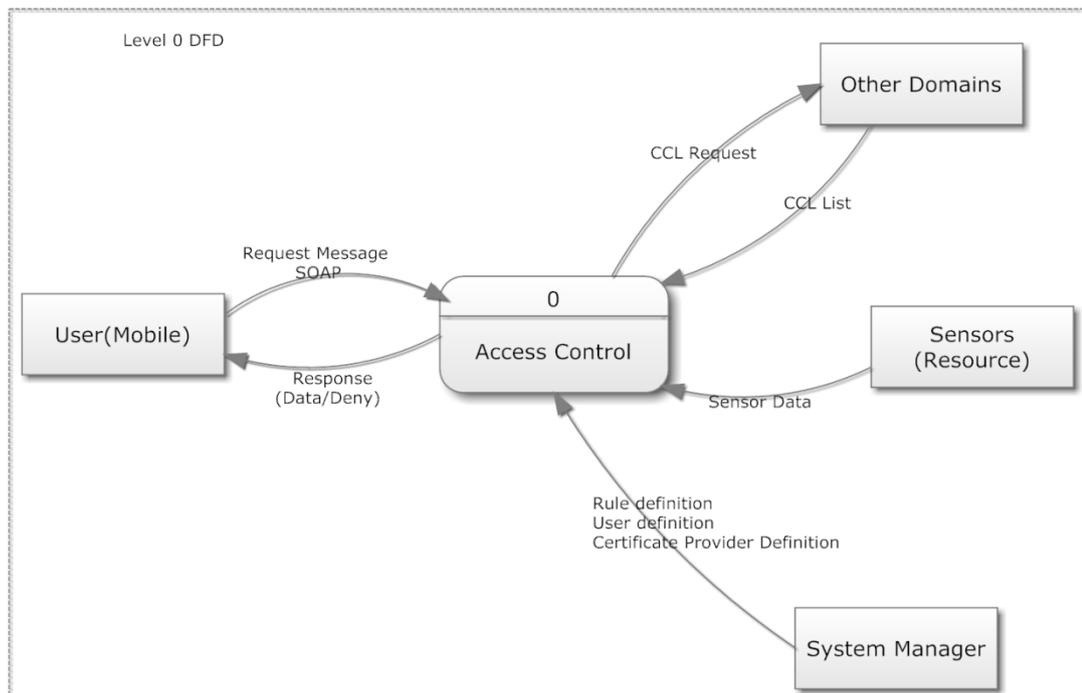


Figure 4.14 Data Flow Diagram of Prototype Implementation

#### **4.4.1. Functional Requirements**

The prototype implementation meets these functional requirements:

- Gateway shall broadcast wireless connection for mobile device.
- Mobile device shall connect to gateway wirelessly.
- Sensor microcontroller shall connect to gateway with serial connection.
- Sensors send their data periodically to gateway.
- System shall listen to serial port and retrieve sensor data.
- Mobile application shall browse mobile device storage for certificate selection.
- Mobile application shall show available resource types according to connected gateway.
- Mobile application shall send access requests as a combined envelope.
- Mobile application shall receive gateway response in a comprehensible structure.
- Gateway software shall serve with five modules: Decision Engine, Context Engine, Certificate Service, Database Service and Data Receiver
- Modules shall use web services for internal communication.
- System shall receive certificate data and resource type in access requests.
- System shall check certificate subject, provider and time interval for validity.
- System shall perform evaluation and create response including sensor data or warning “deny”.
- System shall use time and location context.
- System shall control other domains’ CCLs for synchronization of active certificate list.

#### **4.4.2. Assumptions**

Some features or functions that are mentioned in the proposed model in Chapter 3 are excluded from the prototype or assumptions are made in order to increase functionality and decrease complexity.

- The management panel is not implemented; transactions are performed throughout the Database Service.
- As it is stated in Chapter 3, certificate validity is controlled by the system, however certificate integrity is not checked.
- Gateway is assumed as having an access point for wireless broadcasting.
- All certificates are already saved in mobile device's SD Card .

## **4.5. Usage Cases of Prototype**

In this section, different cases about trying to reach resource sensor data according to related rules and context will be analyzed. Sensors are located in the Institute of Informatics (II) at Middle East Technical University (METU) and users of METU or member of user groups of METU\_II and METU\_CENG have different rules and privileges. Also it is assumed that METU and Bogazici University (BOUN) have inter-domain resource usage agreement between them and users of BOUN have access to reach sensor data according to defined rules.

### **4.5.1. Parameters of Usage Cases**

The pre-defined system parameters and details of access requests are explained in following descriptions.

- METU and BOUN are certificate domains, therefore, they can create certificates for their users.
- CCL list synchronization is performed every 10 seconds.
- Contexts controlled by the system are time and location, however, location context is controlled by the gateway during wireless connection, therefore, rules are created mainly regarding the time context. Six different time context are explained in Table 4.1
- Ten different Access Rules (AR) are defined, as indicated in Table 4.2
- Location: Wireless Lab of Institute of Informatics(II) at Middle East Technical University(METU)
- Users: Davut and Serhat (METU)

- User Groups: METU II Users (Ahmet), METU Computer Engineering (CENG) Users (Elif) and BOUN Users (Emre and Eda)
- Resources: Temperature and Light Sensors

Table 4.1: Time Contexts for Usage Cases

Context	Type	Explanation
Weekend	Time	Saturday-Sunday
Everytime	Time	All times
Evening	Time	18:00-23:59
Monday	Time	Monday
September	Time	September
FallTerm	Time	September-December

Table 4.2: Access Rules for Usage Cases

Context	User or User Group	Resource	Response
Everytime	davut	all	allow
Everytime	serhat	light	allow
Weekend	METU_II	temp	allow
Monday	BOUN	temp	deny
Evening	serhat	temp	allow
Weekend	BOUN	all	deny
FallTerm	BOUN	light	allow
Evening	METU_CENG	light	deny
FallTerm	METU_CENG	all	allow
FallTerm	BOUN	temp	allow

#### 4.5.2. Cases

Different cases on access requests, related context about requests and evaluation results will be indicated.

**Case 1:** METU domain user “davut” wants to reach temperature sensor data with the following time context.

Time of request: 15.09.2011-11:35:18 (Day time, Thursday, Fallterm)

**Result:** System allows user “davut” to access temperature data, because related user has one following Access Rule including all time access to all resources.

Context	User	Resource	Response
Everytime	davut	All	allow

**Case 2:** METU domain user “serhat” wants to reach light sensor data with the following time context.

Time of request: 22.08.2011-20:30:00 (Evening, Monday)

**Result:** System allows user “serhat” to access light data, because related user has two following Access Rules and first rule allows “serhat” to access light sensor data all time.

Context	User	Resource	Response
Everytime	serhat	light	allow
Evening	serhat	temp	allow

**Case 3:** METU\_II user “ahmet” wants to reach temperature sensor data with the following time context.

Time of request: 22.08.2011-19:30:00 (Evening, Weekday)

**Result:** System does not allow user “ahmet” to access light data and returns “deny” response to user , because related user has one following Access Rule, however, related rule indicates requests with the “Weekend” time context. The time of request is not in the “Weekend” range; therefore, Context Engine does not send any rule to Decision Engine and access is not granted to the user.

Context	User Group	Resource	Response
Weekend	METU_II	temp	allow

**Case 4:** BOUN domain user “emre” wants to reach light sensor data with the following time context.

Time of request: 21.08.2011-20:30:00 (Evening, Weekend, Sunday)

**Result:** System does not allow user “emre” to access light data and returns “deny” response to the user. Related user has four following Access Rules and user has access to all resources in Fall Term, however related time context is not in the range of Fall Term, also second rule indicates that on the Weekends access to all resources is denied.

Context	User Group	Resource	Response
Monday	BOUN	temp	deny
Weekend	BOUN	all	deny
FallTerm	BOUN	light	allow
FallTerm	BOUN	temp	allow

**Case 5:** BOUN domain user “emre” wants to reach temperature sensor data with the following time context.

Time of request: 27.09.2011-20:30:00 (Evening, Tuesday, Fallterm)

**Result:** System allows user “emre” to access temperature data. Related user has four following Access Rules and user has access to all resources in Fall Term, however, even request time is in the Fallterm range, if access was requested on Monday or Weekend, user would not access to resource because of “deny” rule’s priority over other rules.

Context	User Group	Resource	Response
Monday	BOUN	temp	deny
Weekend	BOUN	all	deny
FallTerm	BOUN	light	allow
FallTerm	BOUN	temp	allow

**Case 6:** METU\_CENG user “elif” wants to reach light sensor data with the following time context.

Time of request: 27.09.2011-20:30:00 (Evening, Tuesday, Fallterm)

**Result:** System does not allow user “elif” to access light data and returns “deny” response to user. Related user has two following Access Rules and user has access to all resources in Fall Term, however, even request time is in the Fallterm range, in the evenings access requests to light data are denied.

Context	User Group	Resource	Response
Evening	METU_CENG	light	deny
FallTerm	METU_CENG	all	allow

**Case 7:** METU domain user “serhat” wants to reach temperature sensor data with the following time context.

Time of request: 27.09.2011-09:30:00 (Morning, Fallterm)

**Result:** System does not allow user “serhat” to access light data and returns “deny” response to user. Related user has two following Access Rules and user has access to temperature data in evenings, however, request time is Morning time, therefore, access is not granted to relate user.

Context	User	Resource	Response
Everytime	serhat	light	allow
Evening	serhat	temp	allow

**Case 8:** BOUN domain user “eda” wants to reach light sensor data with the following time context.

Time of request: 19.09.2011-16:30:00 (Daytime, Weekday, Monday, Fallterm)

**Result:** System allows user “eda” to access light data. Related user has four following Access Rules and user has access to all sensor data in Fallterm. If user wants to access temperature data, she would not access to resource because of “deny” rule’s priority over other rules.

Context	User Group	Resource	Response
Monday	BOUN	temp	deny
Weekend	BOUN	all	deny
FallTerm	BOUN	light	allow
FallTerm	BOUN	temp	allow

**Case 9:** METU\_CENG user “elif” wants to reach temperature sensor data with the following time context.

Time of request: 08.03.2011-20:35:00 (Evening, March, Fallterm)

**Result:** System allows user “elif” to access temperature data. Related user has four following Access Rules and user has access to all resources in Fall Term; however, even request time is in the Fallterm range, if user wants to access light data, she would not access to resource because of “deny” rule’s priority over other rules

Context	User Group	Resource	Response
Evening	METU_CENG	light	deny
FallTerm	METU_CENG	all	allow

**Case 10:** BOUN domain user “emre” wants to reach light sensor data with the following time context; however, BOUN domain cancelled Emre’s certificate and updated Certificate Cancellation List (CCL).

Time of request: 19.09.2011-16:30:00 (Daytime, Weekday, Monday, Fallterm)

**Result:** System does not allow user “emre” to access light data. Although he has access to light sensor data according to following Access Rules, system sends “deny” response, because, his certificate is cancelled and home domain realizes that after performing synchronization.

Context	User Group	Resource	Response
Monday	BOUN	temp	deny
Weekend	BOUN	all	deny
FallTerm	BOUN	light	allow
FallTerm	BOUN	temp	allow

All of the above usage cases are applied on the prototype implementation and they work correctly according to related rules and contexts. If the user is allowed access, the mobile application presents temperature sensor data as in Figure 4.12 (B), light sensor data as in Figure 4.13 (A) and if user access is denied, the mobile application gives “deny” response as in Figure 4.13(B).

## **CHAPTER 5**

### **CONCLUSION**

In this chapter, conclusion of this thesis is provided. First summary and contribution of this study is analyzed then possible future works are discussed.

#### **5.1. Summary**

In this thesis, a certificate based authentication control model using smart mobile devices is offered for ubiquitous computing environments. Model consists of three main components which are mobile domain, gateway domain and local resource or service. Access control flow mainly starts in the smart mobile device. When a user wants to reach a local resource or service, individual certificate and requested resource type are sent to the gateway by the user using the mobile device.

The gateway validates user certificate by parsing and controlling required fields of certificate and then it collects required contexts such as location, user and time etc. about user. Evaluation process is performed according to access rules that are identified in XML format and collected contexts. If the user is allowed, related resource data or service according to application is granted to the user.

A real implementation based on the proposed model is developed as a model prototype. In this implementation, an Android based mobile application is developed for user access requests, and gateway software is developed for collecting required contexts, managing access rules and performing access evaluation requests. Real temperature and light sensors data are used as local resource and their data are retrieved and saved by gateway software for user requests.

## **5.2. Contribution**

Using smart mobile devices for access requests and reaching resources is the major contribution of this study. In the literature, offered access control models do not provide mobile device usage opportunity. In the ubiquitous computing environments, resources are distributed in the environment and in order to reach resources and use services effectively, mobile devices need to be used. Especially in the prototype implementation of this study, a mobile application that runs on a smart mobile device is developed for reaching resource.

Offered model in this study combines three main properties of ubiquitous computing environments. The model provides a context aware access control and smart mobile device usage and also the model works in inter-domain environment. In the literature, offered access control models generally are lacking in some of these three capabilities.

Another contribution is that, this study involves a real prototype implementation developed for the proposed model. The prototype implementation consists of a mobile application running on a mobile device, gateway software and also temperature and light sensors data are used as local resources. Usage cases applied on the prototype show the applicability and feasibility of the proposed model.

## **5.3. Discussion**

Because resources are distributed in ubiquitous computing environments, mobile device usage is needed in order to reach these resources. However mobile devices are limited electronic devices in terms of processing capability, memory, power and wireless connection. In addition to hardware deficiencies, mobile devices we use in the prototype also have some programming limitations such as Android OS not having standard libraries for web service (SOAP) library used for the user access request process.

Also in this study, users store their certificates in their mobile devices. Because access control mechanism is based on role based access control instead of user based access control, certificates are created for user groups. This may cause some authorization problems such as, a user having another user's certificate and accessing to a resource although they are not authorized. This problem can be solved by checking certificate integrity or pairing

certificates with users' mobile phones. By doing this, access requests can only be sent after this checking mechanism.

#### **5.4. Future Work**

In this study, resources or services and mobile devices are connected to the gateway. When a user wants to reach a resource, first s/he needs to connect to the gateway for both transmitting his/her access request and receiving data if s/he is allowed. As a future work, after successful authentication, a user can connect directly to a resource using mobile devices instead of reaching over gateway. In order to do that, a token can be given to the user after successful authentication. This type of work can decrease network traffic and also decrease gateway workload.

Another future work can be offered for communication type between mobile device and gateway. In this study wireless connection (WLAN) is used as the communication medium between mobile device and gateway. As an alternative Bluetooth can be used for communication and this method can also provide searching resources in an environment.

## REFERENCES

- Abowd, D. G., & Dey, A. K. (1999). Towards a better understanding of context and context-awareness. *1st international symposium on Handheld and Ubiquitous Computing*, (pp. 304–307.). London, UK,: Springer-Verlag.
- Abowd, G. D., & Mynatt, E. D. (2000). Charting Past, Present, and Future Research in Ubiquitous Computing. *ACM Transactions on Computer-Human Interaction*, Vol. 7, No. 1., 29–58.
- Al-Rwais, S., & Al-Muhtadi, J. (2010). A Context-aware Access Control Model for Pervasive Environments. *IETE TECHNICAL REVIEW VOL 27 | ISSUE 5 | SEP-OCT*, 371-379.
- Alsulaiman, F., Mieke, A., & Saddik, A. E. (2007). Threshold-based Collaborative Access Control (T-CAC). *International Symposium on Collaborative Technologies and Systems* (pp. 46-56). CTS.
- Arduino Uno. (2011). Retrieved August 29, 2011, from <http://arduino.cc/en/Main/ArduinoBoardUno>
- Bertino, E., Bonatti, P. A., & Ferrari., E. (2001). TRBAC: A Temporal Role-Based Access Control Model. *ACM Transactions on Information and System Security*, vol. 4, no. 3., 191-223.
- Bertino, E., Samarati, P., & Jajodia, S. (1993). Authorizations in Relational Database Management Systems. *1st Conf.- Computer & Comm. Security* (pp. 130-139). VA,USA: ACM.
- Chadwick, D. W., Otenko, A., & Ball, E. (2003). Role-Based Access Control With X.509 Attribute Certificates. *Ieee Internet Computing March- April 2003*, 62-69.

- Chen, G., & Kotz., D. (2000). A survey of context-aware mobile computing research. *Technical Report TR2000-381, Dartmouth College, Computer Science, Hanover, NH*, 1-16.
- D., G., Abowd, Anind K. Dey, P. J., Davies, N., Smith, M., & Steggles, P. (n.d.). Towards a better understanding of context and context-awareness. *1st international symposium on Handheld and Ubiquitous Computing*,.
- Fickas, S., Kortuem, G., & Segall, Z. (1997). Software Organization for Dynamic and Adaptable Wearable Systems. *IEEE*, 56-63.
- Garlan, D., Siewiorek, D. P., Smailagic, A., & Steenkiste, P. (2002). Project Aura:Toward Distraction-Free Pervasive Computing. *Pervasive Computing*, 22-31.
- Google. (2011). Retrieved August 28, 2011, from <http://code.google.com/p/ksoap2-android/>
- Han-bing, Y., He-ping, H., Zheng-ding, L., & Rui-xuan., L. (2005). Dynamic Role and Context-Based Access Control for Grid Applications. *TENCON 2005* (pp. 1-7). Region 10: IEEE.
- Kindberg, T., Barton, J., Morgan, J., Becker, G., Caswell, D., & Debaty, P. (2002). People, Places, Things:Web Presence for the Real World. *ACM MONET (Mobile Networks & Applications Journal)*.
- Koufi, V., & Vassilacopoulos, G. (2008). Context-Aware Access Control for Pervasive Access to Process-Based Healthcare Systems. *eHealth Beyond the Horizon IOS Press* , 679-684.
- Kumar, A., Karnik, N., & Chafle., G. (2002). Context Sensitivity in Role-based Access Control. *ACM SIGOPS Operating Systems*, 3.
- Lee, Y., Min, C., Ju, Y., Pushp, S., & Song, J. (2011). A Mobile Context Monitoring Platform for Pervasive Computing Environments. *5th IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2011)*, (pp. 345-348). Daejeon, Korea: IEEE.

- Licence, A. (2011). *Google*. Retrieved August 27, 2011, from <http://source.android.com/about/index.html>
- Lieberman, H., & Selker, T. (2000). Out of context: Computer systems that adapt to, and learn from, context. *IBM SYSTEMS JOURNAL, VOL 39, NOS 3&4,* 617-632.
- Lim, T., & Shin., S. (2007). Intelligent Access Control Mechanism for Ubiquitous Applications. *6th IEEE/ACIS International Conference on Computer and Information Science* (pp. 955-960). IEEE/ACIS.
- Oracle. (2010). Class Simple Date Format Retrieved August 20, 2011, from Oracle: <http://download.oracle.com/javase/1.4.2/docs/api/java/text/SimpleDateFormat.html>
- Raento, M., Oulasvirta, A., Petit, R., & Toivonen, H. (2005). ContextPhone: A Prototyping Platform for Context-Aware Mobile Applications. *PERVASIVE computing IEEE*, 51-59.
- Ravi S. Sandhu. (1993). Lattice-Based Access Control Models. *IEEE Computer*, 9-19.
- Sandhu, R., Coyne, E. J., Feinstein, H. L., & Youman., C. E. (1996). Role based Access Control Models. *IEEE Computer, vol. 2,* 38-47.
- Satyanarayanan, M. (2001). Pervasive Computing: Vision and Challenges. *IEEE Personal Communications August*, 10-17.
- Schilit, B. N., & Theimer, M. M. ( 1994). Disseminating Active Map Information to Mobile Hosts. *IEEE Network September/ October*, 22-33.
- Thompson, M., Johnston, W., Mudumbai, S., Hoo, G., Jackson, K., & Essiari, A. (1999). Certificate-based access control for widely distributed resources. *8th conference on USENIX Security Symposium* (pp. Volume 8 17-25). CA, USA: USENIX Association.

- Wang, C.-D., Feng, L.-C., & Wang, Q. (2007). Zero-knowledge-based user. *International Conference on Multimedia and Ubiquitous Engineering* (pp. 784-789). Washington, DC, USA: IEEE Computer Society.
- Want, R. (2009). *An Introduction to Ubiquitous Computing*. 1-33.
- Weiser, M. (1991). The Computer for the 21st Century. *Scientific American*, Vol.265 No.3 , 94-104.
- Weiser, M. (1996). *Ubiq*. Retrieved August 30, 2011, from Ubiquitous Computing: <http://www.ubiq.com/hypertext/weiser/UbiHome.html>
- Weiser, M., & Brown, J. S. (1997). The Coming Age Of Calm Technology. *Copernicus*, New York, NY, USA,, 75-85.
- Yortanli, A. (2011). A Certificate Based, Context Aware Access Control Model for Multi Domain Environments. *Ms.C Thesis Middle East Technical Universiy*.
- Zhang, G., & Parashar, M. (2003). Dynamic Context-aware Access Control for Grid Applications. *Proceedings of the Fourth International Workshop on Grid Computing (GRID'03)* (pp. 1-8). IEEE.