Faculteit Ingenieurswetenschappen
Vakgroep Informatietechnologie
Voorzitter: Prof. Dr. Ir. P. Lagasse

Localisatie van personen met behulp van WiFi en Smart Cards

door

Pieter Lootens
Tim De Bruyn

Promotoren: Prof. Dr. Ir. B. Dhoedt, Prof. Dr. Ir. F. De Turck
Scriptiebegeleiders: M. Strobbe, G. De Jans, D. Verslype

Scriptie ingediend tot het behalen van de academische graad
van licentiaat in de Informatica

Academiejaar 2005–2006

Faculteit Ingenieurswetenschappen
Vakgroep Informatietechnologie
Voorzitter: Prof. Dr. Ir. P. Lagasse

Localisatie van personen met behulp van WiFi en Smart Cards

door

Pieter Lootens
Tim De Bruyn

Promotoren: Prof. Dr. Ir. B. Dhoedt, Prof. Dr. Ir. F. De Turck
Scriptiebegeleiders: M. Strobbe, G. De Jans, D. Verslype

Scriptie ingediend tot het behalen van de academische graad
van licentiaat in de Informatica

Academiejaar 2005–2006

Voorwoord

Een thesis schrijven is geen gemakkelijke opdracht. Gelukkig hebben we gedurende dit jaar hulp gekregen. Een welgemeende dank-u-wel aan onze begeleiders: Matthias Strobbe, Gregory De Jans en Dieter Verslype voor hun hulp gedurende het ganse jaar, het beantwoorden van onze vele vragen en het naleeswerk. We willen ook onze promotoren prof. De Turck en prof. Dhoedt bedanken voor het uitschrijven van dit onderwerp en prof. Demeester voor het beschikbaar stellen van de infrastructuur. Daarnaast ook een oprecht merci aan Lieselotte Olders, die als niet-informaticus toch de tijd heeft genomen om de tekst grondig te lezen. Tenslotte willen we ook onze ouders bedanken die ons de mogelijkheid hebben geboden om te studeren en ons altijd gesteund hebben.

Pieter Lootens en Tim De Bruyn, mei 2006

Toelating tot bruikleen

Pieter Lootens en Tim De Bruyn, mei 2006

Localisatie van personen
met behulp van WiFi en Smart Cards

door

Pieter Lootens
Tim De Bruyn

Scriptie ingediend tot het behalen van de academische graad
van licentiaat in de Informatica
Academiejaar 2005–2006

Promotoren: Prof. Dr. Ir. B. Dhoedt, Prof. Dr. Ir. F. De Turck
Scriptiebegeleiders: M. Strobbe, G. De Jans, D. Verslype
Faculteit Ingenieurswetenschappen
Universiteit Gent

Vakgroep Informatietechnologie
Voorzitter: Prof. Dr. Ir. P. Lagasse

Samenvatting

In het verleden werden reeds verschillende systemen ontwikkeld om aan locatiebepaling te doen. De meeste van deze systemen falen echter in een indooromgeving wegens de vele externe factoren die de locatiebepaling in deze omgeving storen, of simpelweg omdat de gebruikte technologie niet indoor kan gebruikt worden (GPS).

Wij trachtten een localiseringssysteem te ontwikkelen dat zich speciaal naar een indooromgeving richt. Hiervoor werd gebruik gemaakt van Wireless Fidelity, de draadloze technologie van het moment. We implementeerden drie verschillende localiseringsalgoritmen en één totaal nieuw concept: het user-profile, een ondersteunend algoritme dat de nauwkeurigheid verbetert door realistische verplaatsingen voor te stellen. Het uiteindelijke doel is dat een bezoeker een gebouw binnenkomt en op een PDA een kaartje te zien krijgt met de route naar zijn afspraak.

Smart cards kennen de laatste jaren een enorme groei. Steeds meer klinkt de vraag om applicaties te beveiligen met deze technologie. In de beginfase hebben we verschillende kaarten en platformen bekeken. De keuze is dan uiteindelijk op de elektronische identiteitskaart gevallen. Het doel is om het systeem, voor de localisatie van personen, te beveiligen met smart cards. Om deze authenticatie te realiseren wordt gebruik gemaakt van de aanwezige PKI (Public Key Infrastructure) op deze kaart.

Trefwoorden

smart card, eID, WiFi, indoor locatiebepaling, authenticatie, radiomap, trilateratie, user-profile

Location determination using WiFi and Smart Cards

Pieter Lootens, Tim De Bruyn

Supervisor(s): Bart Dhoedt, Filip De Turck

Abstract— **We developed a system to locate a person in a building using WiFi. The system was secured through smart card technology. In this article we will present an overview of our work in 2 parts. The WiFi section analyses some algorithms to determine a location. The smart card section explains how we used this technology to add authentication to our application. In particular we used the Belgian electronic identity card and the PKI on it.**

Keywords— **smart card, eID, WiFi, indoor locatiebepaling, authenticatie, radiomap, trilateratie, user-profile**

I. Introduction

IMagine you have an appointement with someone inside an office building. How will you locate this person in a quick and efficient way? In this paper we will try to present a solution to this problem. The system we have built will locate a person in a building using WiFi and will be secured with smart card technology. The meant setup of the system is visible in figure 1:

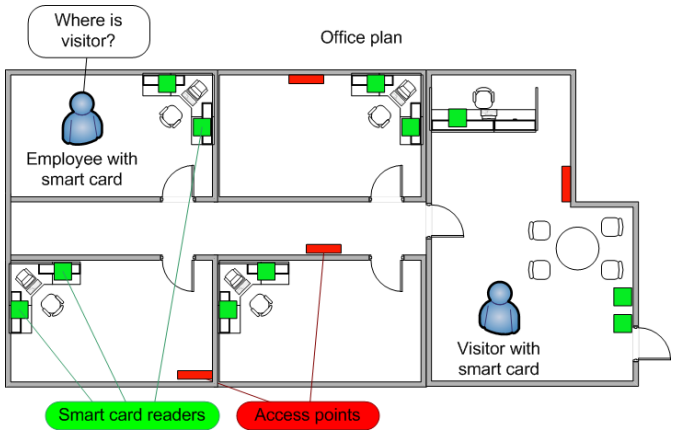


Fig. 1
SETUP OF THE SYSTEM

a visitor arrives, identifies himself using a smart-card, the employee who is going to be visited does the same, they are both located and guided to each other.

The programming language we use is Java. The setup consists of a client-server architecture. The server offers its 'services' to the client. Our client - a Personal Digital Assistent (PDA) - is a lightweight component, therefore the server has to do all the calculations. Suppose a visitor enters the building. He will get a PDA where he has to login ('authenticate himself'). The PDA connects to the server and demands some services. First he will receive an overview of his appointments with employees. The location determination module will determine the location of the visitor and the employee. The PDA will visualize a map with the route the visitor has to follow. If the visitor is dynamically in the system (when he is logged in), the route will update automatically as the visitor moves. The system can also be applied by employees. When an employee logs in, he can get an overview of all his colleagues he is allowed to see (this is done by a 'Hiërarchical User Profile'). Subsequently his colleagues are located

E-mail: PieterD.Lootens@UGent.be
E-mail: Tim.Debruyn@UGent.be

II. WiFi

The decision to choose WiFi out of the many wireless technologies to use in our location determination system was based on three reasons:

- Wifi is a relative cheap technology
- Many buildings already have WiFi access points installed which makes WiFi a cheap investment
- Wifi has a wide enough reach which other technologies, for example bluetooth, are lacking.

We developed two sorts of algorithms to determine people's location. A first sort is based on a radiomap. This sort of algorithms uses two phases: in the first phase (offline phase) reference points all over the building are scanned and stored in the database. In the second phase (online phase, the location determination phase) the environment is scanned for a second, and the obtained scannings are compared with the scannings in the database to determine the position with the highest probability to be the searched position. The algorithms we built were based on the ones in [2].

A second sort of algorithms are trilateration algorithms [3]. There location is estimated by using the structure of the building. The algorithms use only one phase. It starts with scanning the environment for one second. For every possible position in the building, it calculates wich scanning values are expected, keeping in account the locations of the access points, the walls around this position, the possible pieces of furniture in the building, etc. By searching the position with expected values the closest to the measured values, we obtain the expected position.

A new technology we developed is the so-called user-profile. The user-profile is an algorithm that helps the radiomap and trilateration algorithms to determine the right position. It does so by calculating the probability that a certain move from one position to another position is made, keeping in account the previous position, the current movement direction and the current speed. This algorithm has two big advantages:

- By giving the 'impossible' locations the lowest probability, location estimation can be quicker: no more calculations have to be done on positions that are impossible
- This algorithm makes sure the most logic positions get the highest probability. By doing this, these positions will have a higher chance to be the chosen location.

III. Smart Cards

First we were planning to use 'a smart card' and push our own applets on it (which would provide information about the user so he could authenticate himself). Therefore we had a look at different platforms (middleware) and standards and decided to choose for the Belgian electronic identity card (eID). The advantages were among others the cost saving aspect and the fact that every citizen will have his own card in the future. The eID has also a Public Key Infrastructure (PKI) included. A PKI allows binding of public keys to users. The authenticity is guaranteed by a Third Party (who has to be trusted in the system). Trust is an important factor. The eID supports the latest PKI standards e.g. X509v3 certificates, Certificate Revocation Listv2 (CRL), 1024 bit RSA keys, SHA-1 hashing algorithms, etc. The card also comes with middleware. But after having some problems with the software to acces the PKI, we changed to an (unoffical) open-source version. For the future it should be considered to switch back to the offical middleware. The idea behind the authentication is that we use the private key ('on board', the key doesn't leave the card) to generate a value. This value is verified with the public key in the application. Therefore we keep a database of visitors with their respective (authentication) certificates. The card doesn't allow sending 'random' values to be encrypted. It only allows generating (digital) signatures (using the private key) from a hash. So we extract a hash (SHA-1) from the data of a user, send this to the eID and this generates our signature needed for authentication. Finally the signature can be verified using the public key extracted from the user's certificate (after a validity check with the CRL list). The authenticion is safe because the user has to enter his PIN code when the card generates a signature. This guarantees that only the owner can use the private key on his card.

IV. PDA

As mentioned in the Introduction, we use a PDA to guide the visitor to an employee. Because of some problems with the eID's middleware on the PDA we had to design a Login module. This module runs on a 'normal' PC and performs the login procedure (smart card access). Afterwards it sends its data together with the signature to the application on the PDA which performs the authentication.

V. Test results

After developing the system, we did some tests to see which performance could be obtained. With the radiomap-algorithms, we discovered that in 60 percent of the location estimations the fault was below five meter. Only 15 percent had an error that was bigger than nine meter. Also, in eighty percent and more of the location estimations, the system located us in the right room. In most of the other estimations it located us in the room next door.

The trilateration algorithm on the other hand, showed some very poor results. It succeeded only in 35 procent of the estimations to locate us with an error that was lower than 9 meter. In eightyfive procent of the time, the algorithm located us in the wrong room. The reasons for the failure of this algorithm are the many external factors that influence the location estimation procedure. The radiomap-algorithms can deal with this by using the offline phase, trilateration can not. Trilateration is a technique that's not really suitable for use in an indoor environment.

The use of the user-profile didn't improve the testresults that much, but they were slightly better, especially in the category of the big mistakes (mistakes bigger than nine meter). Only about eight percent of the estimations had an error above nine meter. Without the user-profile we had seventeen procent 'big' errors. The explanation is that the user-profile doesn't allow moves that are unrealistic because of the size of the move. So, when a bad measurement of the environment happens, without user-profile the estimation can jump to the other side of the map, while the version that has a user-profile will be limited in error.

VI. Conclusions

Radiomap emerged to be a valuable method for location estimation systems. In most cases we got an acceptable position estimation.

Trilateration on the other hand, emerged to be a bad technique to implement a location estimation system. It couldn't deal with all the external factors inside a building.

The developed user-profile proved to have potential. Results were slightly better when using it, but improvement can be done here.

The current smart card technology is increasing fast. Because of it's mobility and features, it certainly offers a lot of opportunities for authentication purposes. The eID allows citizens to authenticate themselves, digitally sign in, etc. It can also be expanded with other data e.g. SIS. In an era were everyone has a computer and communication between people is just a stream of bytes, the eID offers a good solution for authentication problems in e.g. e-commerce. In our application we give an example of how to use smart card technology and PKI to authenticate users.

Acknowledgments

The authors would like to acknowledge, next to the (head)supervisors, our daily supervisors: Matthias Strobbe, Gregory De Jans and Dieter Verslype. Our final paper would not have been possible without their help.

References

[1] Pieter Lootens, Tim De Bruyn, *Localisatie van personen met behulp van WiFi en Smart Cards* final paper University Ghent, 2006.

[2] Moustafa A. Youssef, Ashok Agrawala, A. Udayay Shankar, Sam H Noh, *A probabilistic clustering-based indoor location determination system* 2002.

[3] Kotchakorn Maksuriwrong, Vara Varavithya, Nachol Chaiyaratana, *Wirless LAN access point placement using a multi-objective genetic algorithm*

[4] David Madigan, Eiman Elnahrawy, Richard P. Martin, Wen-Hua Ju, P. Krishnan, A.S. Krishnakumar, *Bayesian indoor positioning systems*

Inhoudsopgave

Lijst van gebruikte acroniemen

AP	Acces Point
APDU	Application Protocol Data Unit
ARL	Authority Revocation List
ATR	Answer To Reset
BelPIC	Belgian Electronic Personal Identification Card
BLOB	Binary Large Object
CA	Certification Authority
CAPI	CryptoAPI
COS	Card Operating System
CRL	Certificate Revocation List
CSP	Cryptographic Service Provider
dB	decibel
DF	Dedicated File
EF	Elementary File
eID	Elektronische IDentiteitskaart
Fedict	Federale Overheidsdient ICT
HTTP	HyperText Transfer Protocol
HTTPS	Secure HTTP
IGSO	Interactive Gateway for Service Operations
ISO	Internation Organisation for Standardization
IT	Incremental Triangulation
JAAS	Java Authentication and Authorization Services
JC	Joint Clustering
JCA	Java Cryptography Architecture
JCE	Java Cryptography Extension

JCRE	Java Card Runtime Environment
JNI	Java Native Interface
JSSE	Java Secure Socket Extension
M.U.S.C.L.E.	Movement for the Use of Smart Cards in a Linux Environment
MAC	Message Authentication Code
MACOS	Multi Application Card Operating System
MF	Master File
OCF	Open Card Framework
OCSP	Online Certification Status Protocol
OSGI	Open Services Gateway Initiative
PAM	Pluggable Authentication Module
PCA	Policy Certification Authority
PCMCIA	Personal Computer Memory Card International Association
PKCS	Public Key Cryptographic Standard
PKI	Public Key Infrastructure
PKIX	Public Key Infrastructure X.509
RA	Registration Authority
RRN	RijksRegister/Registre National
SOAP	Simple Object Acces Protocol
SSL	Secure Sockets Layer
TPDU	Transport Protocol Data Unit
UDDI	Universal Description, Discovery and Integration
UP	User-Profile
WiFi	Wireless Fidelity
WSDL	WebService Description Language

Hoofdstuk 1

Inleiding

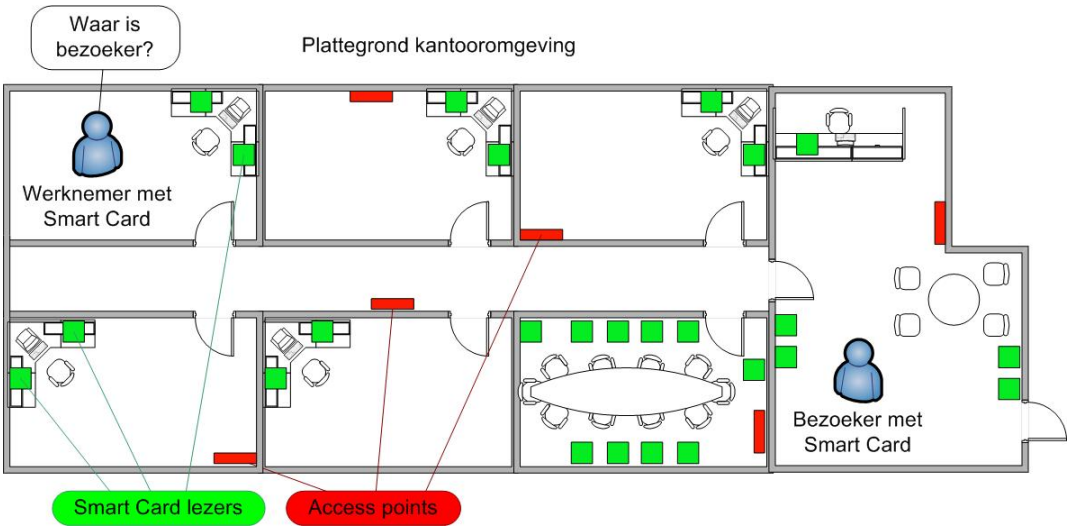
1.1 Situering

De afgelopen jaren worden gekenmerkt door een explosieve groei van draadloze communicatie-technologieën zoals WiFi en Bluetooth. Naast de klassieke toepassingen in computernetwerken kunnen draadloze technologieën ook gebruikt worden voor 'location based services'. Dit zijn diensten die afhankelijk zijn van de plaats van de gebruiker. Een gekend voorbeeld is het GPS navigatiesysteem in auto's. Een andere mogelijkheid is bv. het opvragen van een nabijgelegen tankstation of restaurant via de GSM als je je in een vreemde stad bevindt. Om de locatie te bepalen van de gebruiker wordt steeds de afstand bepaald tot verschillende zogenaamde access points. Dit kunnen satellieten zijn (GPS), GSM zendmasten of WiFi antennes. Bij WiFi wordt bv. de sterkte van het ontvangen signaal gemeten. Wanneer de afstand tot meerdere access points gekend is, kan d.m.v. trilateratie de positie van de gebruiker bepaald worden.

Daarnaast komt er vanuit de industrie ook steeds meer en meer vraag naar het gebruik van smart cards in applicaties. Deze kaarten bevatten een elektronische chip waarop men informatie of data kan opslaan die men kan beveiligen door middel van een PIN-code. De smart card technologie laat dus toe om gepersonaliseerde en vertrouwelijke informatie bij te houden op een plastic kaart. Zo kan ze onder andere gebruikt worden om medische, financiële of fysieke informatie te bewaren.

1.2 Doelstelling

Het doel van deze thesis is om personen in een gebouw te localiseren met behulp van WiFi en het systeem te beveiligen met smart card technologie. Nemen we als voorbeeld een bedrijf met enkele kantoren, een vergaderzaal en een bezoekersruimte, zoals op Figuur 1.1 is weergegeven.



Figuur 1.1: Voorbeeld van een toepassing

Indien er zich nu een bezoeker in het gebouw bevindt die een afspraak heeft met een werknemer, plugt de bezoeker zijn kaart in in de kaartlezer in de wachtzaal. Er wordt een connectie opgezet met een server die constant bepaalt waar alle werknemers zich bevinden met behulp van de access points die zich verspreid in het gebouw bevinden. De bezoeker krijgt dan zijn eigen locatie en deze van de persoon waarmee hij een afspraak heeft te zien op een kaartje samen met de kortste weg. Analoog kan een werknemer zijn kaart inpluggen in een lezer in het kantoor waar hij zich momenteel bevindt en nagaan waar bv. een collega zich bevindt. De smart card kan naast persoonlijke gegevens ook hiërarchische informatie bevatten zodat een werknemer enkel zijn collega's en bezoekers kan localiseren maar geen bazen of zodat een bezoeker enkel de persoon kan zien met wie hij een afspraak heeft.

Hoofdstuk 2

Locatiebepaling via WiFi

Er is op dit moment een overvloed aan draadloze technologieën op de markt die stuk voor stuk bepaalde voordelen en nadelen bezitten (zie tabel 2.1). Om aan locatiebepaling te doen is onze keuze gevallen op Wireless Fidelity (WiFi). De voordelen van het gebruik van deze techniek zijn de kleine kost en daarbij aansluitend het feit dat in vele gebouwen al WiFi-toegangspunten voorhanden zijn die ook gebruikt worden voor andere doeleinden. WiFi heeft ook een grote reikwijdte die andere technologieën zoals Bluetooth bijvoorbeeld niet hebben.

Techniek	*Kostprijs*	*Nauwkeurigheid*	*Indoor/Outdoor*	*Reikwijdte*
GPS	*Zeer duur*	1 *tot* 5 *m*	*Outdoor*	*Sattelietafstand*
Bluetooth	*Goedkoop*	1 *tot* 3 *m*	*Indoor*	1 *tot* 10 *m*
WiFi	*Goedkoop*	1 *tot* 5 *meter*	*Indoor*	30 *tot* 45 *m*
ZigBee	*Goedkoop*	1 *tot* 5 *m*	*Indoor*	10 *tot* 75 *m*
Infrarood	*Zeer duur*	1 *tot* 3 *meter*	*Outdoor*	*Binnen zicht*

Tabel 2.1: Vergelijkende tabel technologieën

In het kader van deze thesis werden er drie locatiebepalingsalgoritmen ontwikkeld, die te categoriseren zijn volgens twee soorten principes: radiomap en trilateratie. In deze volgorde zullen we ze ook verder bespreken in dit hoofdstuk waarna we wat meer gaan vertellen over het zogenaamde 'user-profile', een ondersteunend algoritme, dat ook door ons ontwikkeld werd. Beginnen doen we echter met een paar algemene specificaties over de WiFi-technologie.

2.1 WiFi

WiFi is de standaard voor wireless communicatie en draadloos internet. De technologie maakt gebruik van radiosignalen. De radiogolven worden verstuurd via antennes of routers en worden vervolgens opgevangen via andere antennes, zoals deze in wireless netwerkkaarten. Het formaat en de structuur van deze radiosignalen werd door het IEEE vastgelegd in een verzameling van standaarden en specificaties voor wireless internet onder de titel IEEE 802.11 (zie tabel 2.2). De meest gebruikte vorm momenteel is de 802.11g vorm. Opdat de computer de radiosignalen zou kunnen ontvangen, moet er een netwerkadapter geïnstalleerd zijn. Wij maken gebruik van een PCMCIA (Personal Computer Memory Card) netwerkkaart die wordt ingebracht in het PCMCIA slot van de laptop.

Standaard	*Frequentie*	*Snelheid*	*Reikwijdte*
802.11*b*	2.4 GHz	11 $Mbps$	$Tussen$ 30 en 45 $meter$ $binnenshuis$
802.11*a*	5 GHz	54 $Mbps$	$Tussen$ 9 en 22 $meter$ $binnenshuis$
802.11*g*	2.4 GHz	54 $Mbps$	$Tussen$ 30 en 45 $meter$ $binnenshuis$

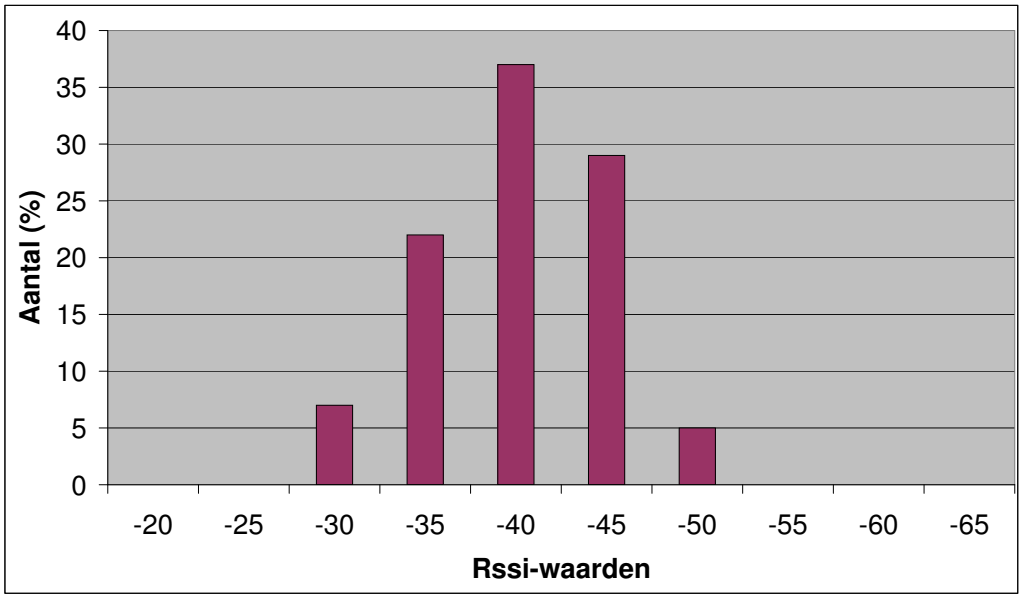
Tabel 2.2: De verschillende IEEE 802.11 standaarden

Laat ons dan even het wireless signaal zelf bekijken. De signaalsterkte hierbij wordt opgemeten in dB (decibel). Door de lage waarden wordt dit echter uitgedrukt in dBm, dit is het absoluut vermorgen in decibel ten opzichte van één milliwatt (mW). De waarde van dit signaal bleek te variëren tussen -17 dBm (beste meting, vlakbij router) en -70 dBm (slechtste meting, amper zichtbaarheid). Wanneer we het verder zullen hebben over signaalsterkte zullen we dit de RSSI-waarde (received signal strength indication) noemen, een standaard in wireless netwerken om de sterkte van een access point te definiëren.

Een belangrijk gegeven om een goede locatiebepaling te doen, is de stabiliteit van het wireless signaal waarop we ons uiteindelijk zullen baseren om te bepalen waar we staan. Als we even de sterktes bekijken die we op een bepaalde locatie verkregen na een minuut scannen (zie figuur 2.1) dan valt onmiddelijk op dat dit signaal regelmatig in sterkte varieert.

Uit ervaring blijkt inderdaad dat het WiFi-signaal heel gevoelig is voor allerhande externe factoren zoals:

- Mensen in de nabijheid die voor het signaal lopen en het zo verzwakken



Figuur 2.1: Gemeten waarden op 1 punt

- De straling van een rinkelende GSM
- Obstakels zoals muren, kasten, enzovoort
- Microgolf
- ...

Hier bleek en blijkt weinig aan te doen. De algoritmen moeten goed genoeg zijn om met dit variabel signaal te kunnen omgaan. De oplossing zou zijn om bij de locatiebepaling langer de omgeving te scannen, maar dit is natuurlijk niet verenigbaar met de performantie.

Indien u nog meer wilt weten over WiFi, kan u altijd terecht bij [1]

2.2 Radio-Map

Radio-Map algoritmen zijn locatiealgoritmen die voor het bepalen van de juiste locatie gebruik maken van een vooraf aangemaakte tabel met zogenaamde referentiepunten. Dit soort algoritmen bestaan uit twee fasen. In een eerste fase (de offline fase) worden overal, verspreid over het gebouw, deze referentiepunten ingescand en in een databank opgeslagen. In de tweede fase, de

Figuur 2.2: Een voorbeeld van een wifi-netwerk

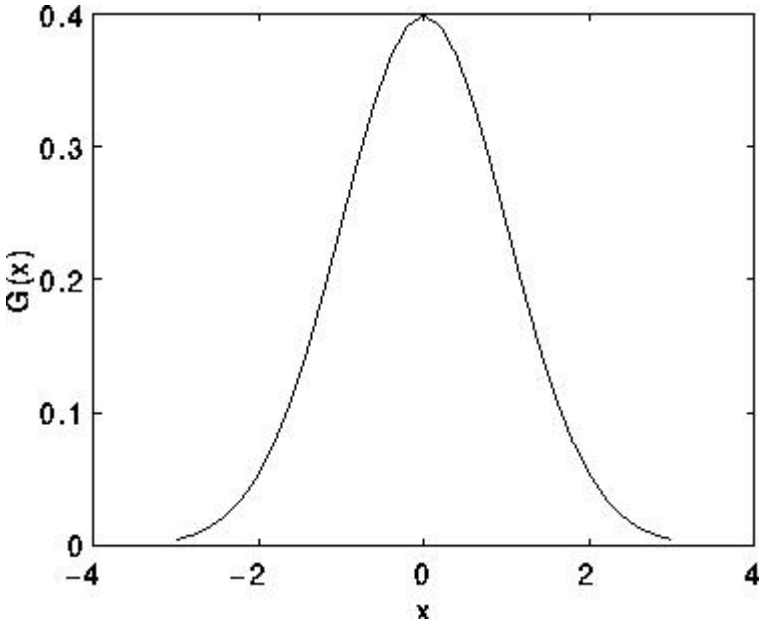
localisatie-fase zelf (online fase), worden deze punten gebruikt om te bepalen waar de te localiseren persoon zich bevindt. Door een seconde lang de omgeving in te scannen en vervolgens deze scanning te vergelijken met wat zich in de databank bevindt, wordt berekend welke positie de meeste kans heeft om correct te zijn. Voor deze algoritmen baseerden we ons op [2] en [3]

In het kader van deze thesis werden er twee radio-map algoritmen ontwikkeld: Joint Clustering (JC) en Incremental Triangulation (IT). Het verschil tussen beide algoritmen ligt in de manier waarop ze de verzameling referentiepunten doorzoeken, het algemene algoritme is bij beiden hetzelfde en zullen we eerst bespreken.

2.2.1 Het radio-map algoritme

Voor het radio-map algoritme kan gebruikt worden, is het dus de bedoeling om eerst een databank met referentiepunten te bekomen waarop we ons kunnen baseren. Dit is een vrij omslachtig en tijdrovend werk, en de afstand tussen de referentiepunten moet goed gekozen worden: hoe kleiner die afstand hoe meer referentiepunten. Vervolgens wordt op elk van die punten een tijd lang de omgeving afgescand. Wij scanden een minuut lang om de driehonderd milliseconden wat neerkomt op een tweehonderdtal metingen per positie. Deze metingen worden dan naar de server verstuurd die het hele histogram opslaat in een tabel, en als gaussdistributie in een andere tabel (een gaussdistributie is symmetrisch om het centrum: de verwachtingswaarde μ van de verdeling is het 'middelpunt' van de grafiek van de verdelingsfunctie, de 'breedte' van de grafiek van de kansdichtheid wordt gekarakteriseerd door de standaarddeviatie, voor een voorbeeld van

een gaussdistributie zie figuur 2.3). Elke rij van de tabel bevat dan de coordinaten van de positie, de naam van het access point en de gemeten waarden of de distributie. Maken we bijvoorbeeld gebruik van vier access points, dan zal elke locatie vier maal in de databank zitten, één rij voor elk (locatie/access-point)-paar.



Figuur 2.3: Een voorbeeld van een gaussdistributie

Tijdens de localisatiefase wordt dan, zoals reeds eerder vermeld, gedurende een seconde de omgeving afgescand op zoek naar bestaande draadloze toegangspunten. De gemeten RSSI-waarden die hierbij bekomen werden worden dan in de vorm van een histogram aan het algoritme gegeven, waarna deze uitzoekt wat de meest waarschijnlijke positie van de persoon is. Allereerst worden uit het verkregen histogram de access points gefilterd die we nodig hebben. In het onderzoek werd gebruikt gemaakt van access points in een gesloten netwerk die enkel operationeel waren voor locatiebepaling, waardoor we enkel deze willen gebruiken, en de andere access points uitsluiten.

Vervolgens doorzoeken we de databank op zoek naar een startverzameling van locaties waaruit we uiteindelijke de goeie locatie zullen bekomen. De manier waarop het selecteren van deze startlocaties gebeurt, is wat het IT-algoritme en het JC-algoritme van mekaar onderscheidt en zullen we later verklaren.

Nu bekijken we één voor een de metingen in het gefilterde histogram, en berekenen voor elke positie uit de startverzameling wat de probabiliteit is dat we op die positie deze gemeten RSSI-

waarden zouden kunnen bekomen, rekening houdend met de waarden die we in de eerste fase op die locatie gemeten hebben. We gaan met andere woorden in de databank kijken hoeveel keer we deze waarden in de eerste fase op die plek gemeten hebben en delen dit door het totaal aantal waarden. We doen dit voor elk koppel gemeten waarde-startpositie en bekomen op het einde voor elke plek een waarde die aangeeft wat de kans is dat we ons daar bevinden. De positie met de hoogste waarde (probabiliteit) is namelijk de gezochte positie.

Samengevat hebben we de volgende formule:

$$max_{sk}(\prod_{i=1}^{n}\prod_{j=1}^{m} P(\alpha_j|\beta_i, S_k)).P(S_k) \tag{2.1}$$

met S_k de locatie, α_j de j-de meting van access point i, β_i het i-de access point, n is het aantal access points, m het aantal samples en $P(S_k)$ wordt door het user-profile berekend (zie later), dit is de kans dat we ons naar deze locatie verplaatsen vanaf de huidige locatie.

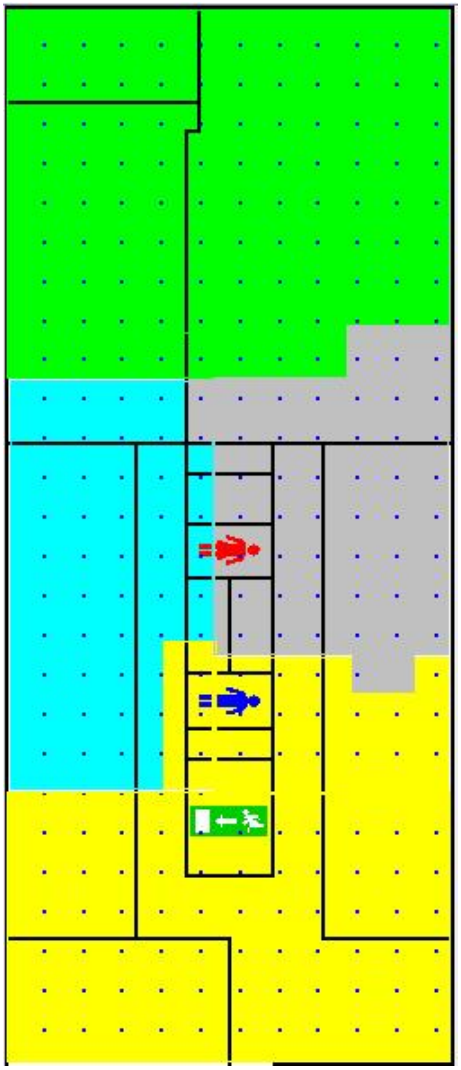
In de praktijk maken we gebruik van de gaussdistributie in plaats van de letterlijke probabiliteit van een waarde op te zoeken omdat het anders kan voorkomen dat we bijvoorbeeld in de databank opzoeken hoeveel keer we waarde -37 gemeten hebben en dan blijkt dat we die exacte waarde nooit gemeten hebben.

2.2.2 Joint-Clustering

Het Joint-Clustering algoritme maakt, zoals de naam al doet vermoeden, gebruik van clusters van access points om te bepalen welke verzameling van posities in aanmerking komt om de gezochte positie te worden (zie figuur 2.4). De reden hiervoor is dat enkel zoeken in bepaalde clusters veel sneller gaat dan zoeken in de hele verzameling van punten. Door enkel die clusters over te houden die een reële kans bezitten om het gezochte punt te omvatten wordt heel wat tijd uitgespaard. We ontwikkelden een algoritme waarbij we ons baseerden op [4]

Het clusteren gebeurt in twee delen: in de fase waarin de referentiepunten worden ingescand, wordt berekend in welke cluster dit punt moet gestopt worden. Elke cluster wordt gekenmerkt door de drie sterkste access points die alle locaties in deze cluster gemeen hebben. Wanneer dan een referentiepunt ingescand wordt, worden alle ingescande access points gerangschikt van sterk naar zwak, waarna het punt in de cluster die bepaald wordt door AP1, AP2 en AP3 gestopt wordt, waarbij AP1, AP2 en AP3 de sterkste gemeten access points zijn.

Wanneer dan in de localisatiefase het ingescande histogram verkregen wordt, wordt er gekeken wat de drie sterkste access points in dit histogram zijn, en vervolgens worden alle locaties die



Figuur 2.4: Een grafische voorstelling van een aantal clusters

in de cluster zitten die gekenmerkt wordt door deze access points als startverzameling van het algoritme gekozen.

In de praktijk bleek dit regelmatig mis te gaan, doordat bepaalde access points heel dicht bij elkaar lagen wat betreft sterkte, waardoor de ene keer het ene access point en de andere keer het andere access points sterker was en bijgevolg vaak in de verkeerde cluster gezocht werd. Dit is opgelost door, indien AP3 en AP4 slechts een bepaalde drempelwaarde van elkaar verschilden, zowel cluster (AP1, AP2, AP3) als cluster (AP1, AP2, AP4) te doorzoeken, en dit voor alle access points die niet veel verschillen in waarde.

De pseudo-code van het ontwikkelde algoritme ziet er zo uit:

```
APS = tabel van alle gemeten access points
voor i = 0 tot CLUSTERGROOTTE
    ti = tabel
voor i = 0 tot CLUSTERGROOTTE
    avg = gemiddelde sterkte APS[i]
    voor j =0 tot APS.length
        als abs( avg - gemiddelde sterkte APS[j]) < 5
            voeg APS [j] toe aan ti
CLUSTERS= alle combinaties van clusters met AP1 uit t1, AP2 uit t2 etc
voor i = 0 tot CLUSTERS.length
    voor alle locaties in CLUSTERS[i]
        bereken prob(locatie)
        sla op
sorteer locaties van groot naar klein volgens prob
geef locatie [0] terug
```

2.2.3 Incremental Triangulation

Het Incremental Triangulation algoritme vertrekt initieel van de volledige verzameling van punten die opgeslagen zijn in de database. Vervolgens worden één voor één de meetwaarden in het bekomen histogram overlopen en wordt de verzameling van dat ogenblik aangepast. Stel dat we bijvoorbeeld beginnen met access point Y, en we hierbij de gemiddelde RSSI-waarde x opgemeten hebben. Dan beginnen we met het overlopen van de verzameling van locaties en we weerhouden enkel die locaties waarvan de absolute waarde van de gemiddelde gemeten sterkte voor Y op die locatie min x kleiner is dan een bepaalde drempelwaarde. We selecteren met andere woorden enkel die posities die een reële kans hebben om de gezochte positie te zijn, rekening houdend met het profiel dat het verkregen histogram van die locatie schetst. In de praktijk bleek dat de verzameling van locaties die opgeslagen waren in de databank voldoende klein was om de drempelwaarde op het maximum te leggen en toch een voldoende grote performantie over te houden. Een drempelwaarde van vijfenzeventig betekent dat in de verzameling van punten enkel die punten toegelaten worden die het access point kunnen zien (vijfenzeventig of daaromtrent is meestal de slechtst mogelijke waarde).

De pseudo-code van dit algoritme gaat als volgt:

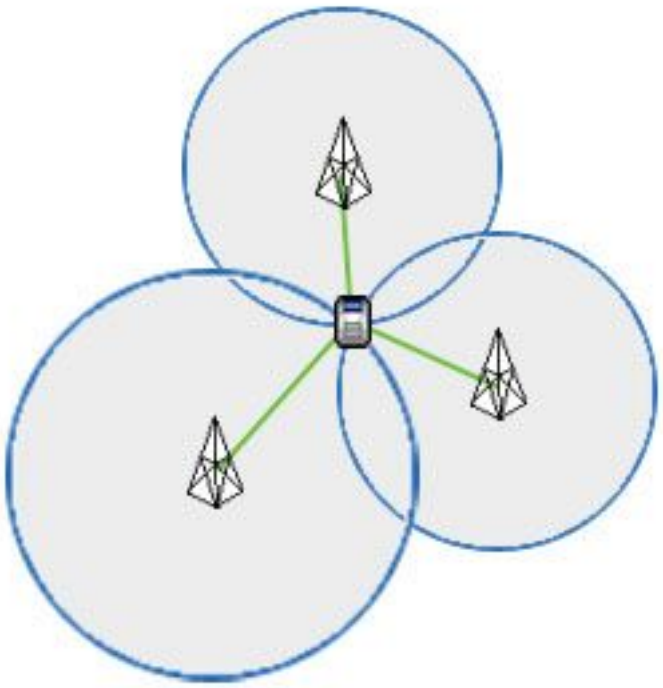
```
APS = tabel van alle gemeten access points
orden APS van sterk naar zwak
X = tabel van alles locaties waarop APS[0] zichtbaar is
voor i = 0 tot X.length
    bereken prob(X[i])
    if(prob > 0)
        bewaar
voor i = 1 tot APS.length
    voor j = 0 tot X.length
        bereken prob(X[j] voor AP[i] * X[j].prob)
        if(prob > 0)
            bewaar
        else
            verwijder X[j]
```

Dis algoritme is een variant van het algoritme beschreven in het tweede deel van [4]

2.3 Trilateratie

Trilateratie is een techniek om de locatie van een persoon te bepalen, wanneer de structuur van de plaats waar men zich bevindt gekend is. Aan de hand van de plaatsing van de muren, access points etc tracht men dan te achterhalen wat de gezochte positie is.

Het algoritme werd zelf ontwikkeld en gaat als volgt te werk. Eerst wordt er een initiële verzameling van punten aangemaakt die in aanmerking komen als gezochte locatie. Indien er reeds een eerdere locatiebepaling gebeurd is, wordt de verzameling gelijkgesteld aan de verzameling van alle punten die binnen een bepaalde straal van de vorige positie liggen. Indien er nog geen eerdere positie bekend is, wordt de verzameling gelijkgesteld aan alle bestaande punten. Vervolgens wordt voor elk punt in de bekomen verzameling berekend welke RSSI-waarden op die locatie verwacht worden, gebruikmakend van de locatie van de verschillende access points, de afstand tot die access points en het aantal muren en hun dikte tussen de bewuste locatie en de



Figuur 2.5: Trilateratie

access points. De locatie waarvan de verwachte meetwaarden het dichtste in de buurt komen van de effectief gemeten waarden, wordt geacht de gezochte locatie te zijn.

De formule die gebruikt kan worden om te berekenen welke meetwaarden we op een bepaalde locatie verwachten is de volgende:

$$SNR = SNR_0 - L - \sum_i n_i w_i \tag{2.2}$$

Hierbij is SNR de verwachte meetwaarde, SNR_0 de gemeten waarde vlakbij het access point, n_i het aantal obstakels tussen de plaats en het access point, w_i de attenuatie factor van het obstakel (dit is de waarde die de grootte voorstelt waarmee de WiFi-signalen geabsorbeerd worden) en

$$L = \begin{cases} 40 + 20 \log d \\ 40 + 20 + 10\gamma \log (d/10) \end{cases} \tag{2.3}$$

met d de afstand tussen het access point en de plaats. Deze formules komen uit [5]

In pseudo-code krijgen we:

APS = tabel van alle gemeten access punten

if(client.vorigePositie==null)

X=tabel met alle locaties

else

X=tabel met alle locaties waarvoor afstand(locatie,vorige positie) < 5

voor i=0 tot APS.length

voor j=0 tot X.length

rssi=berekende rssi-waarde op X[j] voor AP[i]

fout=gemeten waarde - rssi

X[j].fout+=fout

sorteer X van grootste fout naar kleinste fout

X[0] is gezochte locatie

2.4 User-Profile

Het user-profile is een nieuwe, door ons ontwikkelde, manier om aan betere locatiebepaling te doen: een algoritme dat de radio-map en trilateratie-algoritmen helpt bij het bepalen van de juiste locatie. Het doet dit door de probabilistische kans te berekenen dat een bepaalde verplaatsing gemaakt wordt, uitgaande van de vorige positie. De locatiebepalingsalgoritmen kunnen dan deze probabiliteit mee gebruiken. In alle, tot nu toe ontwikkelde, locatiebepalingssystemen werd elke nieuwe locatiebepaling onafhankelijk van de vorige behandeld. Dit user-profile zorgt echter voor een vernieuwing door met de voorgaande geschiedenis van posities rekening te houden.

2.4.1 Werking

In het user-profile worden een aantal dingen bijgehouden: elke keer een nieuwe locatie bepaald is, moet dit aan het user-profile gemeld worden, waarop deze de huidige locatie, de locatie daarvoor en de huidige snelheid van de client aanpast en wegschrijft.

Wanneer nu een algoritme aan het user-profile vraagt wat de kans is dat de persoon zich verplaatst heeft naar positie x, zal deze beginnen met het pad te berekenen van de laatst gekende positie naar locatie x. Belangrijk hierbij op te merken is dat dit pad maar een bepaald aantal meter lang mag zijn (in dit geval ingesteld op vijf meter) en dat een verplaatsing van kamer naar kamer vanzelfsprekend enkel door een deur kan en bijgevolg verplaatsingen door muren dus

uitgesloten worden. Wanneer nu blijkt dat er zo'n pad bestaat, wordt vervolgens de probabiliteit berekend dat dit pad gevolgd werd, uitgaande van de lengte van het pad en de huidige snelheid, en uitgaande van de richting van het pad en de richting die de persoon momenteel aan het volgen was. Zo zal bijvoorbeeld een pad van vijf meter naar rechts weinig punten krijgen als de bewuste persoon zich aan een snelheid van een meter per seconde naar links aan het verplaatsen was.

2.4.2 Voordelen

- Het meest voor de hand liggende en belangrijkste voordeel van het gebruik van een user-profile is het feit dat de meer logische verplaatsingen meer kans zullen krijgen om gevolgd te worden. Wanneer er twijfel bestaat tussen het kiezen van twee locaties waaronder eentje die logisch gezien onhaalbaar is, maar enkel door een slechte meting een kans toegedicht krijgt, zal het user-profile ervoor zorgen dat de andere, correcte, locatie gekozen wordt.

- Zonder user-profile kan het gebeuren dat in een lange, smalle gang zonder deuren er toch regelmatig van kamer veranderd wordt. Het is voldoende om dicht genoeg tegen een muur aan te lopen en een meting te doen die een klein beetje afwijkt om plots aan de andere kant van de muur op te duiken. Het user-profile zorgt er voor dat locaties aan de andere kant van een muur, zonder deur in de buurt, automatisch weggefilterd worden.

- Een laatste voordeel is de snelheidswinst. Aangezien het user-profile enkel locaties binnen een bepaalde afstand van de vorige locatie aanvaardt, krijgen alle andere locaties automatisch probabiliteit nul. Hierdoor dienen deze locaties ook niet langer in het algoritme onderzocht te worden, wat de verzameling van te onderzoeken punten kleiner maakt en de iteraties sneller.

2.4.3 Nadelen

- Na meerdere testen bleek het belangrijkste nadeel samen te vallen met één van de voordelen, namelijk het feit dat het user-profile ervoor zorgt dat er geen verplaatsingen door muren kunnen gebeuren. In sommige gevallen bleek immers dat, wanneer men zich in een kamer verplaatste en de locatiebepaling niet onmiddelijk volgde, het algoritme de deur naar die kamer niet meer vond en als het ware tegen de muur bleef opbotsen. Om een kamer binnen te gaan is het nodig om eerst op de paar punten voor de deur terecht te komen, omdat de afstand van buiten de kamer tot binnen de kamer te lang is om in één

stap af te leggen. Als oplossing voor dit probleem hebben we de afstand die mag afgelegd worden in één keer wat groter gemaakt, waardoor de sprong in de kamer wel in één keer kan gemaakt worden. Dit heeft dan wel weer als nadeel dat er veel grotere (onrealistische) sprongen kunnen gemaakt worden, waardoor deze afstand zorgvuldig moet gekozen worden.

- Er dook nog een andere nadeel op verbonden met de beperking om slechts een aantal meter in één keer af te leggen. Dit nadeel treedt vooral op wanneer de allereerste locatiebepaling een heel slechte locatiebepaling is. De positie kan dan namelijk slechts hersteld worden aan x meter per keer, waarbij x de maximale afstand is. Zo kan het zelfs voorvallen dat de berekende positie de echte positie nooit inhaalt.

Hoofdstuk 3

Smart Cards

"*One card to rule them all, one card to find them,*
One card to bring them all and in the darkness bind them."
With due apologies to JRR Tolkien.

3.1 Algemeen

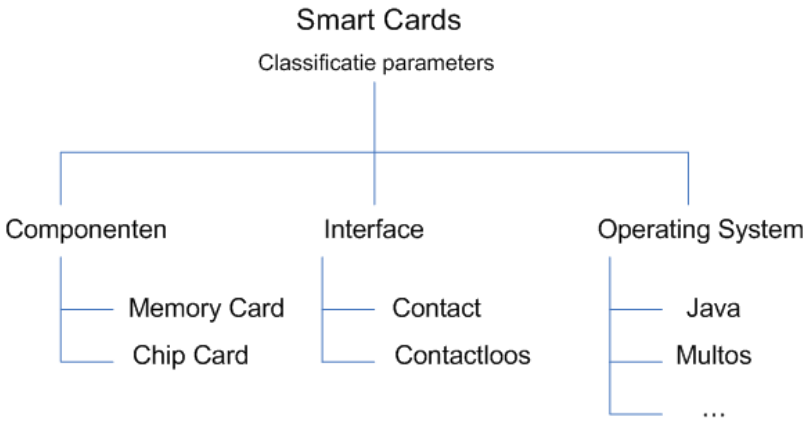
De smart card-technologie kent de laatste jaren een enorme groei in Europa. Smart cards zijn in de meeste geïndustrialiseerde landen een manier van leven geworden. Meer en meer worden ze geïntegreerd in reeds bestaande applicaties. We gebruiken ze om onszelf te identificeren, toegang tot gebouwen te krijgen, geld af te halen, te betalen. Ze worden gebruikt in telefooncellen, mobiele telecommunicatie (SIM-cards), SIS-kaarten, Protonkaarten, elektronische identiteits-kaarten, ski-pass, ... Er zijn 2 hoofdredenen om een chip op een kaart te gebruiken. De eerste is om data op te slaan. De tweede reden is security: smart cards en de geïntegreerde circuits erin hebben verschillende features die hen toelaten om niet alleen data veilig op te slaan, maar ook om gebruikers te authenticeren. Daarnaast bieden ze ook een meerwaarde op het gebied van mobiliteit. Smart cards zijn klein en gemakkelijk te dragen, maar toch is er genoeg processorkracht en data-opslag aanwezig om persoonlijke informatie op te slaan. Zo hoeft de gebruiker bijvoorbeeld geen gebruikersnamen en wachtwoorden meer te onthouden, aangezien die door smart cards kunnen worden bijgehouden. Dit kan zelfs uitgebreid worden naar gebruikersprofielen, cryptografische sleutels en certificaten om zo bijvoorbeeld e-commerce applicaties te ondersteunen.

3.1.1 Introductie

De 'Kaart' heeft een hele evolutie achter de rug: van magnetisch, over geheugen naar smart cards zoals we ze nu kennen. Een smart card wordt als volgt gedefinieerd volgens Dhar [6]:

> "*Een plastiek kaart, gewoonlijk met dezelfde grootte en vorm als een kredietkaart, die een microprocessor en geheugen bevat (wat toelaat om data op te slaan en te verwerken), conform de ISO 7816 standaard.*"

We kunnen ze classificeren op basis van verschillende parameters. Een eenvoudig overzicht wordt gegeven in Figuur 3.1



Figuur 3.1: Smart card classificatie

1. Component gebaseerd:
   - Memory Cards: Dit zijn de meest gewone en goedkoopste kaarten. Ze bestaan uit EEPROM of ROM. Het geheugen kan afgeschermd worden met een PIN code. Een toepassing is bijvoorbeeld de prepaid telefoonkaarten waar het EEPROM de tegoed kredieten kan bijhouden.
   - Microprocessor/Chip Cards: Dit zijn de echte smart cards. Ze bestaan uit ROM (bevat het OS), EEPROM, RAM en een CPU.
2. Interface gebaseerd:
   - Contact: Elke kaart heeft 6-8 contactjes die in fysisch contact staan met de lezer. Nadelen zijn o.a. slijtage, 'card tearing' (plotseling uittrekken van de kaart), ...

Ter illustratie geeft Figuur 3.2 de contacten weer volgens ISO-7816. Deze standaard wordt later besproken.

Contact No.	Contact Designation	Contact Function
C1	Vcc	Supply Voltage
C2	RST	Reset
C3	CLK	Clock
C4	RFU	Reserved for Future Use
C5	GND	Ground
C6	Vpp	Programming Voltage
C7	I/O	Input/Output
C8	RFU	Reserved for Future Use

Figuur 3.2: Smart card contacten volgens ISO-7816

- Contactloos: Deze zijn zoals de naam zegt niet verbonden met de lezer en moeten gewoon in de buurt van een antenne gehouden worden. Nadeel is dat deze kaarten duurder zijn.

3. OS gebaseerd:

   Elke smart card heeft een operating system. Het is de hardwarespecifieke firmware die zorgt voor de basisfunctionaliteit. Het OS zorgt voor alle bestands - en geheugenbeheer en datatransmissieprotocollen.

   - COS: Card Operating System: is een sequentie van instructies, permanent ingebed in het ROM. COS instructies zijn niet afhankelijk van een bepaalde applicatie maar worden frequent gebruikt door vele applicaties. Dit laat bijna geen flexibiliteit toe om één van deze componenten te veranderen zonder te moeten investeren in nieuwe software en/of hardwareimplementatie.
   - MACOS: Multi Application Card Operating Systems: Dit laat de ontwikkeling toe van verscheidene applicaties die op een kaart kunnen draaien.

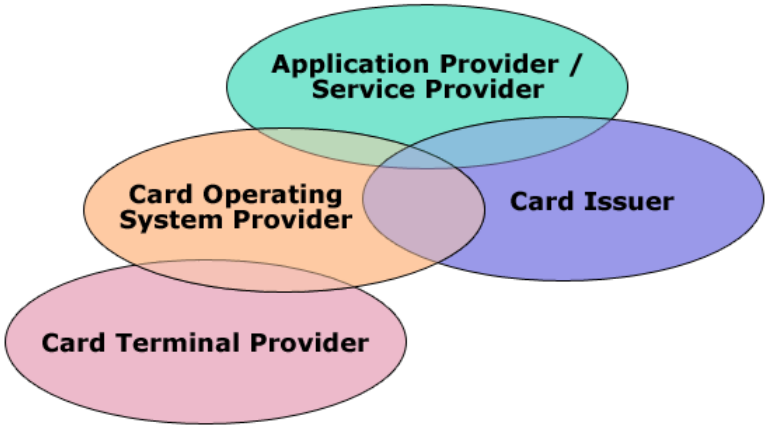
   Enkele voorbeelden:

   - MultOS
   - Java Card
   - Windows for Smart Cards

- CyberFlex
- StarCOS
- MFC

Een smart card applicatie bestaat essentieel uit twee delen: een card-resident applicatie en een card-using applicatie. De eerste bevindt zich op de kaart en bevat alle data. De 2e bevat de code die interageert met de card-resident data en functies.

Bij het inzetten van een smart card applicatie komen er verschillende partijen kijken.
Een overzicht wordt gegeven in Figuur 3.3



Figuur 3.3: Betrokken partijen bij het ontwikkelen van smart card applicaties

- Card Issuer: Degene die de kaart uitbrengt, de producent van de kaart
- Application/Service Provider: Staat in voor de applicatie op de kaart. Vroeger waren de Card Issuer en de Provider dezelfde firma maar tegenwoordeig zijn deze met de opkomst van de MACOS gescheiden. Zo kan de provider zijn applicaties op kaarten van verschillende issuers zetten. Issuers kunnen op hun beurt ook applicaties van verschillende ontwikkelaars op hun kaart zetten.
- Card Operating Systems
- Card Terminal Provider: De uitrusting en hardware drivers

3.1.2 Platformen - Standaarden

"*The nice thing about standards is that you have so many to choose from.*"
Andrew Tanenbaum

ISO 7816

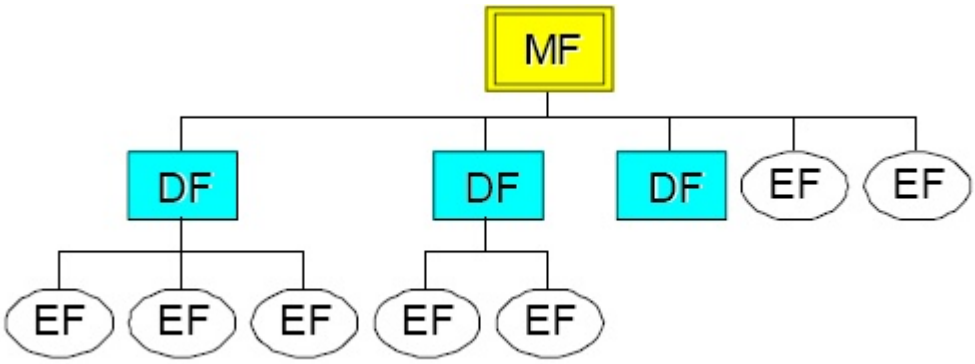
ISO 7816 beschrijft de lowest-level interface naar een smart card. Het is op dit niveau dat de databytes getransfereerd worden tussen de kaartlezer en de kaart.

Tabel 3.1: Enkele ISO-7816 definities

$7816-1$	Fysische karakteristieken
$7816-2$	Elektrische module
$7816-3$	Elektrische signalen en protocollen
$7816-4$	Inter-industriële commando's
$7816-7$	Databank
$7816-8$	Security mechanismen

- ISO7816-3: beschrijft hoe de kaart en de lezer communiceren. Er zijn twee gestandardiseerde protocollen nl. T=0 en T=1. Bijna alle beschikbare kaarten volgen de T=0 standaard. In de sectie 3.1.3 gaan we hier dieper op in.
- ISO7816-4: beschrijft o.a. de bestandsarchitectuur en commandosets (bestandsbeheer, authenticatie, ... - commando's). De bestanden zijn in drie grote groepen te verdelen (zie Figuur 3.4). De eerste is de MF (Master File). Deze is aanwezig in alle kaarten en wordt impliciet geselecteerd na het resetten van de kaart. Het bevat alle andere directories en bestanden. De DF(Dedicated File) is een directory waar andere logisch samenhangende bestanden gegroepeerd zitten (DF en EF). Als laatste heb je de EF(Elementary File). Deze bevat de data die nodig is voor de applicaties.

Voor meer informatie wordt verwezen naar de ISO 7816 website [7].



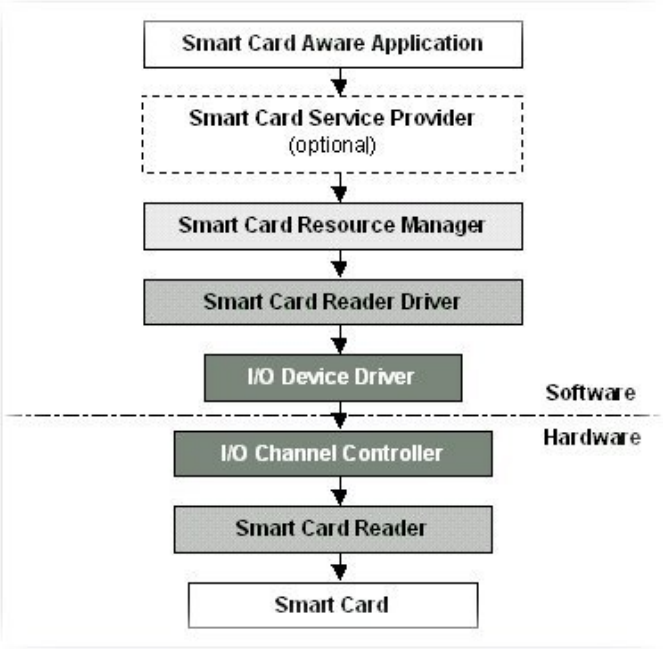
Figuur 3.4: Bestandsarchitectuur

PC/SC

PC/SC is een standaard, voorgesteld door de PC/SC Workgroup opgericht in 1996. Dit comité is samengesteld uit de 'grote' smart card producenten en computer-firma's (Microsoft, HP, Gemplus, ...). De Interoperability Specification for ICCs and Personal Computer Systems (PC/SC) beschrijft het gebruik van smart cards met de Personal Computer. De specificatie omvat de karakteristieken voor de kaarten en de lezers zodat de programmeur onafhankelijk is van de lezer. Deze is gebaseerd op en volledig compatibel met ISO 7816. Aangezien Microsoft een belangrijke rol heeft gespeeld in de uitwerking van deze standaarden, is Windows nu het enige platform dat een 'officiële' Smart Card Resource Manager aanbiedt (de Unix/Linux gemeenschap maakte implementaties voor andere platformen, zie volgend punt voor meer informatie). Deze manager is de centrale component van de specificatie. Er is ook een volledige MSDN library beschikbaar om hier mee te werken.

Figuur 3.5 toont de structuur van de PC/SC architectuur.

- Smart Card Service Provider: Deze component is optioneel maar kan zeer nuttig zijn aangezien het een standaard interface voor smart cards aanbiedt. De Provider wordt geleverd door de fabrikant.
- Smart Card Resource Manager: is de centrale component van de specificatie en moet geïntegreerd zijn in het OS van de computer. De manager laat simultane toegang tot een lezer toe, identificeert of localiseert smart cards. Het resultaat is een meer eenvoudige driver en gemakkelijkere toegang.



Figuur 3.5: PC/SC Architectuur

- Smart Card Reader Driver: vertaalt de aanvragen van de resource manager naar de taal van de lezer. Hij kiest en zet ook het gepaste I/O kanaal op om te communiceren met de lezer. Hij kan ook overweg met meerdere lezers tegelijk.

Voor meer informatie wordt verwezen naar de website van de PC/SC Workgroup [8].

PCSC Lite

Een implementatie van de PC/SC standaard voor de meeste UNIX type OS'en, Mac OS X en Windows. PCSC Lite werd gecreëerd door M.U.S.C.L.E. (= Movement for the Use of Smart Cards in a Linux Environment). Deze beweging heeft tot doel een ondersteuning te voorzien voor smart cards en cryptografie over diverse platformen. PCSC Lite laat een gemakkelijke porting toe van Windows smart card software naar andere operating systems. Ze biedt ondersteuning voor verschillende types van seriële, PCMCIA en USB smart card readers en cryptografische tokens. Het doel van PCSC Lite is een kleine afgeslankte versie van een Windows smart card interface te voorzien voor de communicatie met smart cards en lezers. Er bestaat middleware (Resource Manager voor de OS'en), drivers, applicaties en dergelijke meer voor deze omgeving.

Voor meer informatie wordt verwezen naar de website van M.U.S.C.L.E. [9].

OpenSC

Het OpenSc project is een smart card bibliotheek en applicatie met ondersteuning voor PKCS #15 compatibele kaarten (PKCS komt in de sectie 3.2.2 aan bod). LibOpenSC is de bibliotheek voor toegang tot smart card toestellen via de PC/SC Lite middleware package. Het is ook de kern van het OpenSC project. De basisfunctie zou moeten werken met elke ISO 7816-4 compatibele kaart. Encryptie en decryptie met private sleutels op de kaart zijn momenteel enkel mogelijk met PKCS # 15 compatibele kaarten. De Belgische ID kaart wordt nog niet ondersteund door OpenSC. De officiële Belgische middleware bevat wel een gemodificeerde versie van OpenSC.

Voor meer informatie wordt verwezen naar de OpenSC website [10].

JPC/SC: JNI wrapper voor PCSC

JPC/SC voorziet in een eenvoudige JNI library die toegang geeft tot de PC/SC functie vanuit Java. Het werd eveneens gecreëerd door M.U.S.C.L.E. JPC/SC biedt een API om toegang te krijgen tot de algemene functies van een smart card (select, read, ...). JPC/SC ondersteunt zowel Windows als Linux. Het biedt geen geavanceerde applicatie- of kaartspecifieke APIs (zoals OCF), maar enkel een resource-efficiënte en eenvoudige toegang tot de smart card infrastructuur. PC/SC wordt ondersteund door de aanvragen van de (Java)applicaties af te beelden op een onderliggende interne PC/SC implementatie. De software hangt af van JPCSC voor de communicatie met de echte smart cards. JPCSC heeft dus nood aan een geïnstalleerde PC/SC omgeving.

De Java Native Interface (JNI) is een API binnen Java. De JNI is een interface tussen programma's in Java en 'native' programma's (programma's die zijn opgesteld in de instructietaal van de hardware waarop ook de Java Virtual Machine (JVM)). Het hoofddoel is te komen tot binaire compatibiliteit met native bibliotheken tussen alle JVM implementaties op een bepaald platform.

JPC/SC wordt gebruikt als onderdeel van een open source tool om te communiceren met de elektronische identiteitskaart (zie sectie 3.4).

Voor meer informatie wordt verwezen naar de website van M.U.S.C.L.E. [9].

Open Card Framework (OCF)

Dit raamwerk is onstaan uit een consortium van firma's die een Java interface voor smart cards goedgekeurd hebben. Het is een object-georiënteerd raamwerk dat zich bevindt op de computer waar de smart card mee verbonden is. De service provider of applicatieprogrammeur schrijft zijn services naar de interface voorzien door het OCF, om te verzekeren dat verschillen of veranderingen in het OS of terminal geen impact hebben op de code. OpenCard is gebaseerd op ISO 7816.

Voor meer informatie wordt verwezen naar de website van Open Card [11].

GlobalPlatform (vroeger Visa's Open Platform)

GlobalPlatform is de standaard voor het beheer van de infrastructuur van een smart card. Dit beheer omvat o.a. de installatie en verwijdering van applicaties. GlobalPlatform voorziet in een reeks van interindustriële technische specificaties. Het bevat zowel kaart - als terminalspecificaties. De beheerder is de smart card issuer, die de volledige controle heeft over de inhoud van de kaart maar rechten kan toekennen aan andere organisaties om hun eigen applicaties te beheren. Het laat de smart card issuers toe om te kiezen tussen besturingssystemen en applicatie-ontwikkelaars terwijl GlobalPlatform security en kaartbeheer aanbiedt. Applicaties worden zo onafhankelijk van issuer, OS of smart card fabrikant en dat laat toe dat verschillende organisaties gezamenlijk op de kaart aanwezig zijn met elk hun eigen 'beschermd' gebied. Het was de bedoeling een API te creëeren die niet in strijd maar complementair is met gelijkaardige intiatieven zoals PC/SC en OCF. Beide initiatieven - PC/SC en OCF - focussen op specifieke doelplatformen. GlobalPlatform is ontwikkeld met het oog op een heel gamma aan devices zoals PC's, NC's, ... Oorspronkelijk gedefinieerd door Visa is GlobalPlatform geëvolueerd naar specificaties voor het complete beheer van de smart card. GlobalPlatform is nu de eigenaar en zorgt voor de verdere ontwikkeling. Het is voorgelegd aan de ISO als ISO/IEC 7816-13 voor standaardisatie.

Voor meer informatie wordt verwezen naar de website van GlobelPlatform [12].

Java Card

Java Card is een smart card die Java bytecode kan uitvoeren. Deze technologie probeert het motto van Java 'write once, run everywhere' ook hier waar te maken. Het kan gebruik maken van

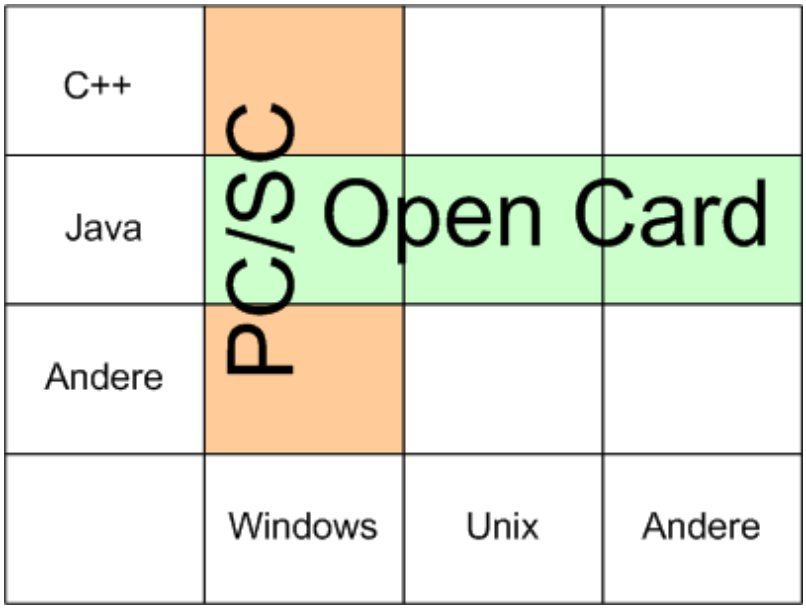
Java's ingebouwde ondersteuning voor security, object-georiënteerde technologie en hardware-onafhankelijkheid. Ontwikkelaars kunnen dus applicaties schrijven via de standaard API. Java Card omvat een multi-applicatie OS voor de smart card. Elke kaart die een Java interpreter geïncorporeerd heeft, kan dezelfde applicatie uitvoeren. Een dergelijke kaart bestaat uit 3 lagen. De onderste laag is het OS, die zorgt voor alle native functies, geheugen, I/O, ... Daarboven komt de JCVM: de Java Card Virtual Machine (die zorgt voor de platformonafhankelijkheid). De derde is het Framework dat alle API's bevat. Deze laag wordt gebruikt om de applets uit te voeren en om de zogenaamde Java Card Runtime Environment (JCRE) te ondersteunen.

Voor meer informatie wordt verwezen naar de website van Java Card [13].

Conclusie

ISO 7816 is reeds verschillende jaren de standaard, maar ze laat voldoende ruimte voor variatie en interpretatie. Dit zorgt ervoor dat componenten van verschillende producenten niet altijd compatibel zijn hoewel ze conform de standaard zijn. De 'hogere' programmeerlagen, die belangrijk zijn voor de applicatieprogrammeur, worden amper gereguleerd. Naast ISO 7816 zijn er verschillende specificaties opgedoken uit applicatiedomeinen. Voorbeelden zijn EN726 uit de telecommunicatiesector, EMV uit de financiële wereld en EU/G7 WG7 uit de gezondheidssector. De eerste 2 grote initiatieven die niet specifiek aan een zakelijk domein gebonden zijn, waren PC/SC en OpenCard. Figuur 3.6 situeert het geheel. PC/SC biedt dus heel wat mogelijkheden en wordt ondersteund door Microsoft. Anderzijds is dat een nadeel aangezien Microsoft enkel zorgt voor ondersteuning voor het Windows platform. Veel systemen die smart cards gebruiken (set-top box, smart phones, ...) draaien geen Windows, alhoewel Microsoft ook hier aan een inhaalbeweging bezig is (Windows for Smart Cards, ...). Het OpenCard Framework (OCF) werkt op verschillende platformen, maar werd helaas al enkele jaren niet meer geüpdatet. We kunnen dus stellen dat beide eerder complementair dan concurrerend zijn. Complementair in hun doelstellingen en in de omgevingen waarin ze zullen ingezet worden. Afhankelijk van de omgeving zal ofwel de ene ofwel de andere technologie adequater zijn. Als je bijvoorbeeld behoefte hebt aan een niet-java applicatie voor Windows en enkel ondersteuning nodig hebt voor een specifieke card issuer dan kan PC/SC een evidente keuze zijn door hun brede ondersteuning. Toch kun je stellen dat zelfs op Windows OCF soms te prefereren is door zijn hogere niveau APIs en OS onafhankelijkheid.

GlobalPlatform werd ontwikkeld voor een breder gamma van toestellen. Het werd ontwikkeld



Figuur 3.6: Vergelijking PC/SC - OpenCard Framework

om te draaien op elk runtime environment die meerdere applicaties toelaat. De API en services aangeboden door PC/SC en OCF zouden kunnen gebruikt worden door het GlobalPlatform. Het is niet de intentie van GlobalPlatform om de concurrentie aan te gaan met bestaande smart card specificaties, operating systems of standaarden. In plaats daarvan wil het deze inspanningen aanmoedigen door een smart card ontwikkelomgeving te voorzien die compatibel is met bestaande systemen en standaarden. In de praktijk is de runtime environment bijna altijd de Java Card Runtime Environment

Als we nu OCF en Java Card bekijken, zien we dat beiden samengaan. OCF is Java op de computer of terminal die in contact staat met de kaart. De Java Card is een speciale, uitgeklede versie van Java die op de smart card zelf draait. Java applicaties die op een PC draaien kunnen dus OCF gebruiken om toegang te krijgen tot Java Cards en standaard smart cards. Als je Java applets ('cardlets') wil schrijven dan moet je een smart card gebruiken die conform is met de Java Card standaard. De enige uitzondering is MULTOS. Hier kan je zelf Java applicaties schrijven en ze dan cross compilen naar MEL (Multo Executable Language) om ze dan te laden naar een MULTOS smart card. Deze applicaties worden opgeslagen als MEL code en uitgevoerd door de MULTOS 'Virtual Machine'. In dit geval gebruik je geen Java Card volgens de Java Card specificaties.

3.1.3 Communicatie

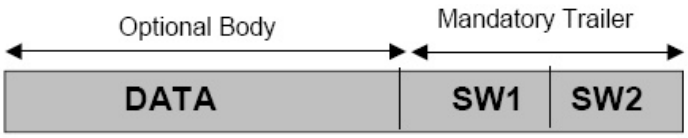
De applicatie communiceert met de lezer, die op zijn beurt 'spreekt' met de smart card, gebruik makend van de standaarden die hierboven besproken werden. De ISO standaard definieert zowel mechanische en elektrische karakteristieken als het protocol om te communiceren met de kaart. Helaas heeft de ISO groep geen standaard voorzien om te communiceren met de lezer. Om dus een commando te kunnen versturen dat de kaart ondersteunt moeten we dit commando 'wrappen' in een ISO commando-pakket. Dit moet op zijn beurt 'gewrapped' worden voor de lezer in kwestie.

De basiseenheid voor het uitwisselen van gegevens met een smart card is het **APDU** pakket. Een Application Protocol Data Unit kan beschouwd worden als een datapakket dat een complete instructie of een compleet antwoord bevat van een kaart. Om deze functionaliteit te voorzien, hebben APDU's een streng gedefinieerde structuur die beschreven is een serie van ISO documenten die behoren tot de 7816 specificatie familie. APDU's die van de terminal naar de kaart verzonden worden, noemt men Command APDU's. Hoe zo'n APDU eruit ziet, kun je zien in Figuur 3.7.

Figuur 3.7: Command APDU: indeling (bron [14])

- CLA: 1 byte klasse van de instructie (om de verschillende groepen instructies te kunnen opdelen)
- INS: 1 byte nummer van de instructie binnen de klasse
- P1/P2: 1 byte parameters voor de instructie (parameter 1 en 2)
- Lc: 1 byte lengte van de command APDU in bytes
- DATA: Lc bytes mee te verzenden data
- Le: 1 byte aantal bytes data die als antwoord terug verwacht worden

APDU's van de kaart naar de terminal zijn altijd antwoorden op instructies en zien er uit zoals Figuur 3.8



Figuur 3.8: Respond APDU: indeling (bron [14])

- Response Data: de door de instructie gevraagde informatie
- SW1: status byte (Status Word1) : status van verwerking commando
- SW2: status byte (Status Word2) : kwalificatie van verwerking commando

TPDU's: Transport Protocol Data Unit. De APDU's worden verzonden door TPDU's. De meest gebruikte protocollen zijn :

- T=0: byte - georiënteerd
- T=1: blok - georiënteerd
- T=CL: contactloos protocol

ATR: Answer To Reset. Elke smart card is verplicht om een antwoord te geven wanneer hij gereset wordt. Deze reset gebeurt bij het opstarten door de smart card terminal (meestal wanneer de kaart wordt ingeplugd, maar kan ook expliciet gebeuren door een specifiek commando te geven aan de lezer). In de reset-fase worden de kaart en de lezer aan elkaar geïntroduceerd en wordt de basis voor de communicatie-sessie gelegd. Deze basis omvat de transmissie-parameters: het ondersteunde transport-protocol, de data-transmissiesnelheid en de hardware-parameters van de kaart.

3.2 Beveiliging

Smart cards zijn zeer veilig. Informatie die opgeslagen is op de chip is moeilijk te dupliceren of te verstoren, in tegenstelling tot de 'externe' opslag op magnetische kaarten, die gemakkelijk

gekopieerd kunnen worden. De chip microprocessor en coprocessor kunnen DES, 3-DES, RSA, ECC, ... , standaarden voor encryptie, authenticatie en digitale handtekening ondersteunen.

Een snel groeiende applicatie is de digitale identiteitskaart. Hier worden de kaarten gebruikt voor authenticatie. Het meest gekende voorbeeld is de samenwerking met een Public Key Infrastructure (PKI). De Belgische elektronische identiteitskaart is hier een mooi voorbeeld van (zie 3.4). Gecombineerd met biometriek leveren smart cards twee -of drie factor authenticatie.

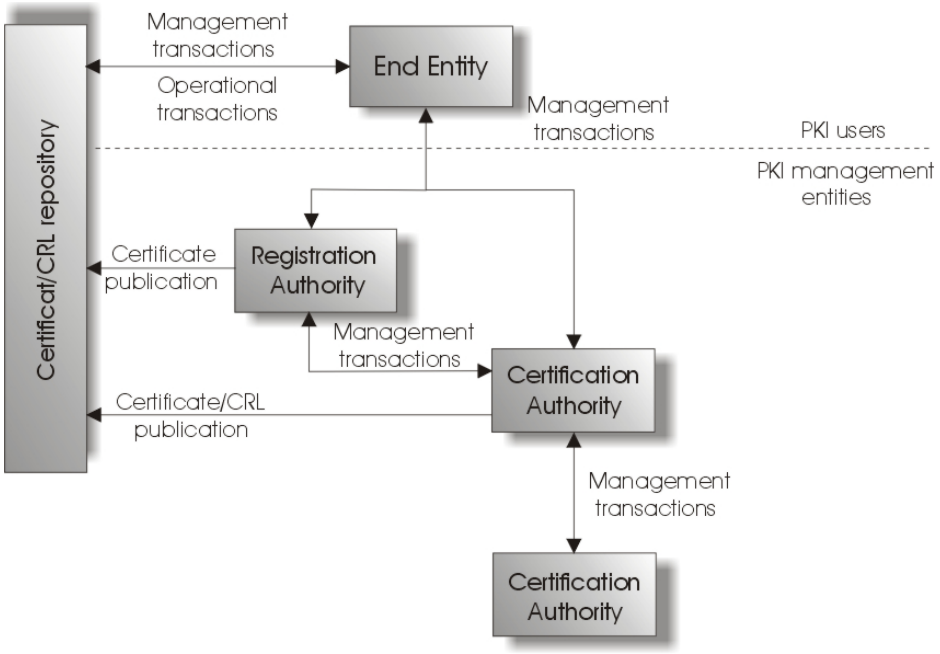
3.2.1 PKI: Public Key Infrastructue

PKI is een verzameling van technische en organisatorische voorzieningen die het gebruik van encryptie door middel van publieke sleutels ondersteunt. Centraal in de infrastructuur staat de derde partij. Deze derde partij bevestigt de identiteit van de eigenaar van een bepaalde sleutel door er een bewijs van vertrouwen, een certificaat, aan te koppelen. Dit certificaat vormt voor andere Internet-gebruikers een bewijs dat een publieke sleutel bij een bepaalde persoon hoort. De waarde die men aan dit bewijs kan hechten is afhankelijk van het vertrouwen die de uitgever van het certificaat geniet. De Public Key Infrastructure X.509 (PKIX) werkgroep heeft de basis gelegd om een architectuur gebaseerd op certificaten op het internet te ontplooien. Ter illustratie geeft Figuur 3.9 het model weer dat door deze groep ontwikkeld werd. Voor meer informatie over PKIX wordt verwezen naar [15]

Elementen van een PKI:

- Policy Certification Authority:
  De taak van een Policy Certification Authority (PCA) is het vaststellen en publiceren van de manier waarop gebruikers of organisaties geregistreerd worden. Het document waarin deze regels worden opgeschreven, wordt de policy genoemd. Deze policy kan bijvoorbeeld regels bevatten over de manier waarop personen zich moeten authenticeren (of eigenlijk legitimeren) voordat ze een certificaat kunnen ontvangen. De manier waarop deze authenticatie plaatsvindt, bepaalt de 'klasse' van een certificaat:

  1. Class 1 Individueel certificaat: garandeert dat het e-mailadres met het certificaat opgeslagen is bij de uitgevende instantie. Dit is geen bewijs van de identiteit van de certificaathouder.



Figuur 3.9: Het PKIX architectuur model

2. Class 2 Individueel certificaat: hier wordt de identiteit van de aanvrager van een certificaat bij een derde partij geverifieerd. Dit is bijvoorbeeld de onderneming waar de aanvrager werkt of een databank van een kredietkaart-onderneming.
3. Class 3 Individueel certificaat: bij deze certificaatklasse wordt de aanvrager zelf geïdentificeerd door gebruik te maken van een identiteitsbewijs.
4. Class 3 Servercertificaat: bij deze certificaatklasse worden de identiteit en de bezitter van de server vastgesteld. Hiervoor worden databases van een vertrouwde derde partij gebruikt.

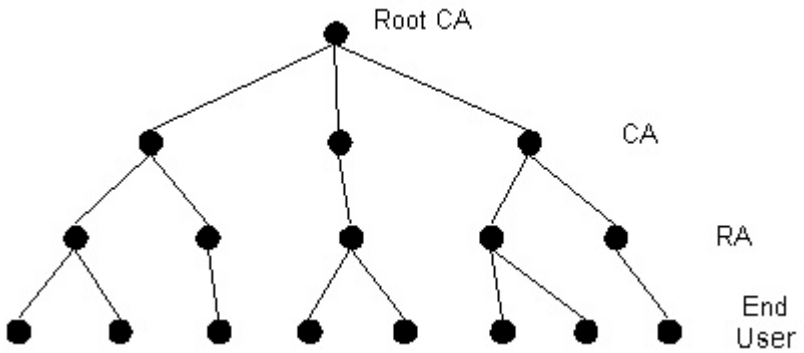
- Certification Authority:

  Het fundamentele uitgangspunt van een PKI is om twee onbekenden (personen of systemen) op een veilige manier met elkaar te laten communiceren. De meest praktische en veilige manier om dit te doen is door het gebruik van een aantal autoriteiten. Het is van cruciaal belang dat deze autoriteiten vertrouwd worden. Deze autoriteiten worden Certification Authorities (CA) genoemd. Het vertrouwen in de authenticiteit van een certificaat is gebaseerd op de handtekening van deze CA op dit certificaat. De echtheid van deze handtekening kan gecontroleerd worden bij de PCA. Dit wordt een zogenaamde **'chain of trust'** genoemd. De CA moet de publieke sleutel dus aan een identiteit koppelen en zo

garanderen dat de persoon die het unieke certificaat in handen heeft, ook daadwerkelijk degene is wie hij zegt te zijn. Daarom is het de kritieke component in datasecurity en e-commerce activiteiten.

- Registration Authority:

  Hoewel de registratie van eindgebruikers vaak door de CA zelf gebeurt, kan het nodig zijn de functie van het uitgeven van certificaten en het registreren van eindgebruikers van elkaar te scheiden om zo de CA te ontlasten. Deze kan dan bijvoorbeeld offline werken, waardoor de kans dat de CA door een netwerkaanval platgelegd wordt verminderd. Hiervoor wordt een Registration Authority (RA) opgezet. Een RA mag nooit zelf certificaten uitgeven - dit is voorbehouden aan de CA - maar kan taken uitvoeren als: het vaststellen van de identiteit van individuen; het certificatieproces opzetten met een CA in opdracht van een eindgebruiker en activiteiten die met key management te maken hebben, zoals het terugtrekken van certificaten. Een typische architectuur kan er dan uitzien zoals in Figuur 3.10.



Figuur 3.10: Hierarchie bij een PKI

Andere belangrijke componenten:

- Digitaal Certificaat:

  Een digitaal certificaat definieert data-formaten en procedures voor de distributie van publieke sleutels, waarbij de sleutels digitaal gesigneerd worden door Certificate Authorities. De publieke sleutel kan zo op een vertrouwde manier uitgewisseld worden. Een 'publieke sleutel certificaat' wordt gebruikt om de naam van een entiteit (met eventueel extra

attributen) aan een publieke sleutel te koppelen. Het is belangrijk verschillende soorten certificaten te onderscheiden, namelijk:

1. X.509 publieke sleutel certificaten
2. Simple Public Key Infrastructure (SPKI) certificaten
3. Pretty Good Privacy (PGP) certificaten
4. Attribuut certificaten

De meest gebruikte toepassing van een digitaal certificaat is om de identiteit van de zendende partij vast te stellen (authenticatie). De ontvangende partij heeft dan de mogelijkheid een versleuteld bericht terug te sturen door gebruik te maken van de publieke sleutel van het certificaat.

- Digitale handtekening:
  Een digitale handtekening is een code die aan een bericht wordt toegevoegd. De zender creëert een **Manipulation Detection Code** (hash) van de boodschap en gebruikt dan zijn private sleutel om deze hash te encrypteren. Dit is de digitale handtekening. Hiermee wordt de zender op een unieke manier geïdentificeerd. Net als bij een geschreven handtekening is het doel van een digitale handtekening te garanderen dat de zendende partij is wie hij zegt te zijn. Om effectief te zijn mogen digitale handtekeningen niet na te maken zijn. Hiervoor zijn verschillende encryptietechnieken voor handen, die verschillende veiligheidsniveaus garanderen.

Kenmerken van een PKI:

- Uitgeven van certificaten:
  In 3.4 zal uitgelegd worden dat in België de gemeente optreedt als RA en CertiPost met GlobalSign als CA.

- Vertrouwensmodellen:
  De belangrijkste vraag die gesteld moet worden wanneer we spreken over vertrouwensmodellen is: wat is vertrouwen? Vertrouwen wordt binnen een PKI als volgt gedefinieerd:

  'Entiteit A vertrouwt entiteit B als A er vanuit kan gaan dat B zich zal gedragen zoals A denkt.'

Vertrouwen is daarom niet kwantitatief uit te drukken, maar gaat altijd gepaard met een bepaald risico dat B zich niet gedraagt zoals A verwacht. Vertrouwen kan dan als volgt vertaald worden naar een PKI: een eindgebruiker vertrouwt een CA als de eindgebruiker er vanuit gaat dat de CA een certificaat op een correcte manier heeft uitgegeven, dat wil zeggen: een accurate binding van gegevens (attributen) aan een publieke sleutel onderhoudt. Verschillende vormen van PKI (zoals PGP en X.509) hebben verschillende vertrouwensmodellen. Een vertrouwensmodel bepaalt welke certificaten een entiteit (persoon of machine) kan vertrouwen, hoe dat vertrouwen tot stand gebracht kan worden en onder welke omstandigheden het vertrouwen gelimiteerd en beheerst kan worden.

- Intrekken van certificaten:
  Er zijn verschillende redenen om een certificaat in te trekken: een hacker heeft de geheime sleutel gekraakt, iemand verlaat een onderneming,... Er moet dan een manier zijn om de certificaten in te trekken. Het waarschuwingsmechanisme wordt Certificate Revocation genoemd. De teruggetrokken certificaten worden gepubliceerd in Certificate Revocation Lists (CRL's). Het is van essentieel belang dat deze informatie tijdig beschikbaar is voor personen binnen een PKI. De nieuwste techniek zorgt ervoor dat men 'online' de status van een certificaat kan checken: OCSP. In de sectie 'X.509 - certificaten' worden CRL en OCSP verder toegelicht.

- Ondersteuning van onweerlegbaarheid:
  In een PKI worden vaak acties ondernomen die onlosmakelijk verbonden zijn met de identiteit van de persoon. Onweerlegbaarheid betekent dat op het moment dat Bob een document signeert, hij dit later niet kan ontkennen. Een PKI kan op zichzelf niet zorgen voor volledige onweerlegbaarheid, omdat er altijd een menselijke factor bij betrokken is. Het opslaan van de geheime sleutel op een veilige plaats (dus niet op de hard disk van de computer), bijvoorbeeld een smart card, kan bijdragen aan de onweerlegbaarheid.

  Een ander belangrijk mechanisme bij onweerlegbaarheid is het gebruik van een tijdstempel. Zo wordt bijvoorbeeld onder een contract, naast de handtekening van de persoon, tevens een tijdstempel gezet die door beide partijen vertrouwd wordt.

Gebruik:

In het hoofdstuk 'Implementatie' zal uitgelegd worden hoe er gebruik werd gemaakt van de PKI (certificaten, CRL, ...) van de elektronische identiteitskaart (zie 3.4) om een gebruiker te authenticeren. Deze kaart maakt gebruik van het X.509 certificaat protocol. Daarom wordt er in volgende sectie wat dieper op dit protocol ingegaan.

X.509 - certificaten:

- De PKI kenmerken van X.509:

  De X.509-standaard heeft de verandering ingeluid van kleine, gesloten netwerken naar grote implementaties, geschikt voor een open netwerk. X.509 is een authenticatieraamwerk, dat gebaseerd is op een ISO-standaard. Bij de ontwikkeling van de X.509-standaard was er een grote vraag naar een vorm van certificering die platformonafhankelijk in een omgeving als het internet of een multinationale onderneming ingezet zou kunnen worden. De X.509-standaard is daarom ook een algemeen, flexibel formaat. Dat is waarschijnlijk ook de reden dat de standaard algemeen geaccepteerd wordt en technisch implementeerbaar is in verschillende omgevingen.

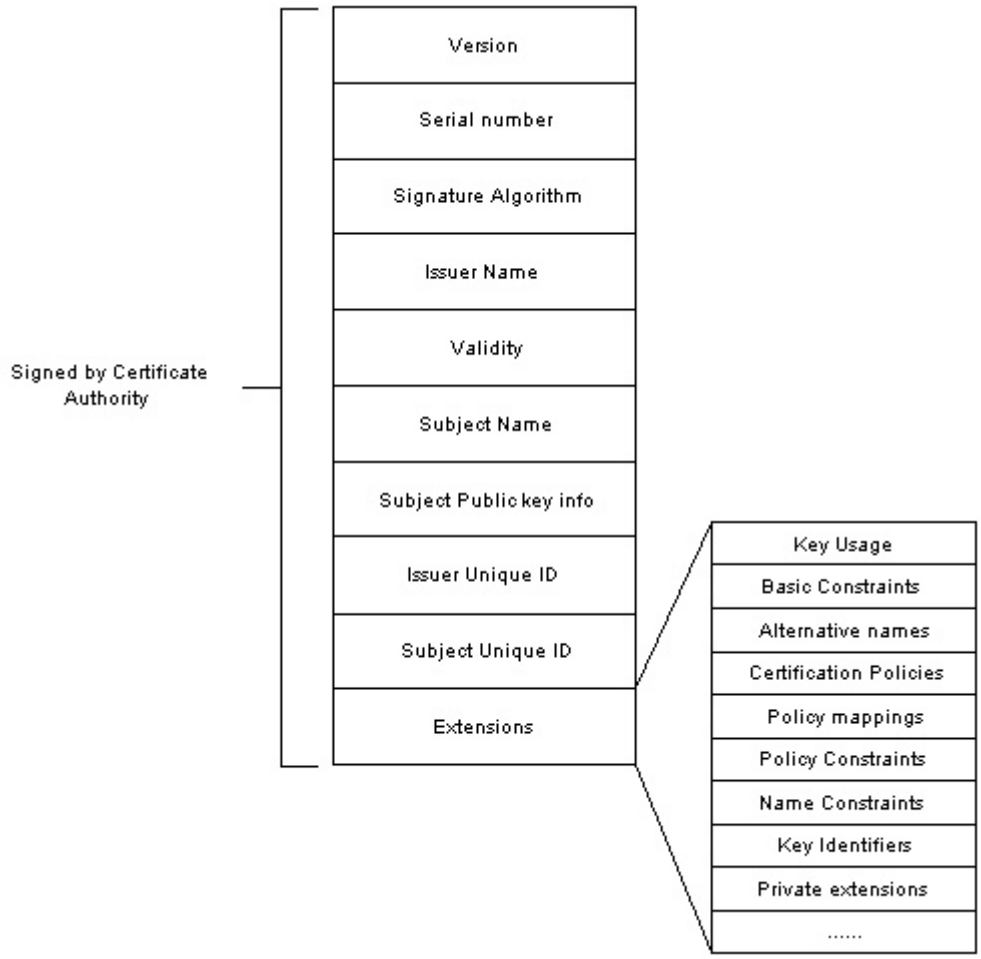
  Momenteel wordt op het internet gewerkt met versie drie van de X.509 standaard (X.509 v3). In versie drie zijn het vooral de nuttige en flexibele extensiemechanismen van certificaten, die de toepasbaarheid verhogen. Met behulp van een extensie kunnen extra kenmerken (attributen) aan een certificaat toegevoegd worden. Ook versie 2 van de Certification Revocation Lists (CRL) heeft hiertoe bijgedragen. Het mechanisme is zo flexibel dat ieder type extensie gedefinieerd kan worden en in een certificaat of CRL geplaatst kan worden; tevens kan bij iedere extensie aangegeven worden of deze deel uitmaakt van het verificatieproces. Het komt er dus op neer dat X.509-certificaten op de omgeving afgestemd kunnen worden, zonder strikte voorwaarden die de bruikbaarheid in andere omgevingen in de weg zouden kunnen staan.

- Vertrouwensmodel voor X.509:

  De X.509 PKI is gebaseerd op de gedistribueerde vertrouwensarchitectuur, waar vertrouwen in de keten doorgegeven kan worden.

- Eigenschappen van certificaten:

  Een X.509-certificaat heeft, zoals te zien in figuur 3.11, de volgende velden:



Figuur 3.11: Versie 3 Public Key certificaat zoals gedefinieerd in X.509

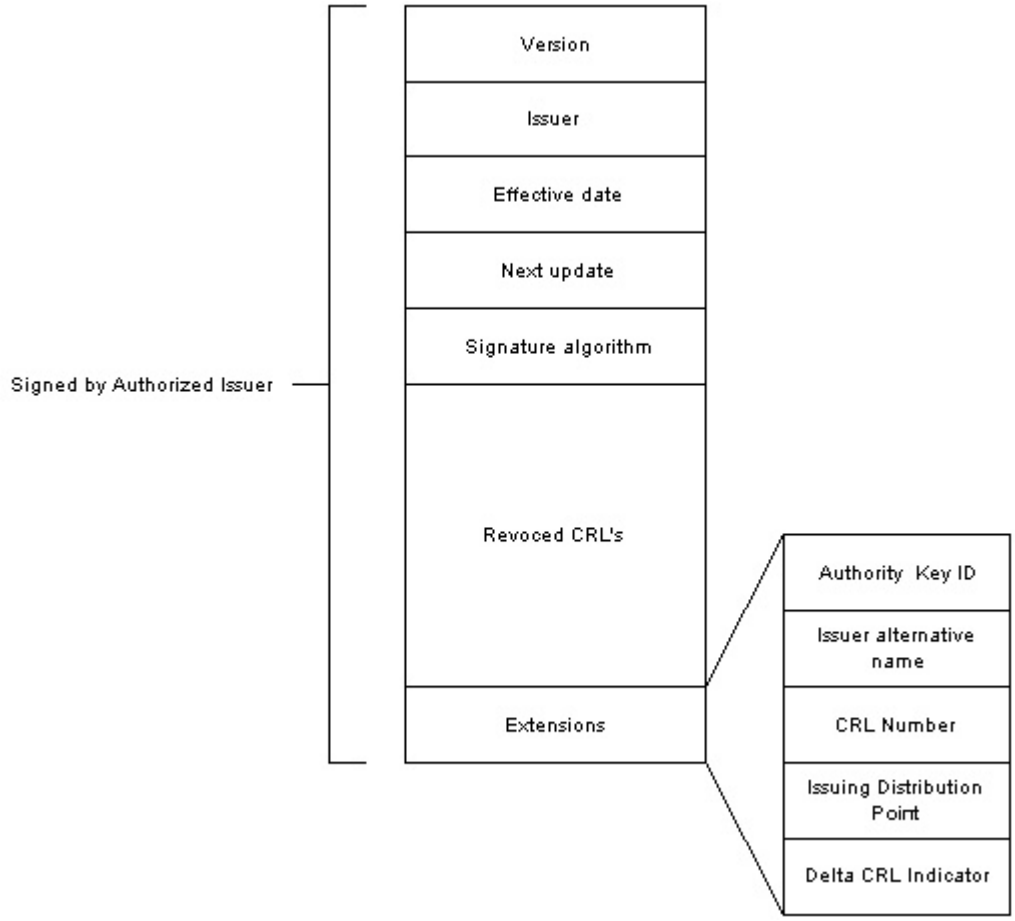
1. Versie: het versienummer waar het certificaat aan voldoet (1,2 of 3).
2. Serienummer: Een uniek nummer dat door de CA aan het certificaat wordt toegewezen.
3. CA-algoritme: Het algoritme dat door de CA gebruikt is om het certificaat te signeren.
4. Naam uitgever: De naam van de CA die het certificaat heeft uitgegeven.
5. Geldigheidsduur van het certificaat: twee velden waarin staat op welke datum en tijd het certificaat aangemaakt is en op welke datum en tijd het certificaat vervalt.
6. Naam houder: De naam van de houder van de geheime sleutel die bij de publieke sleutel hoort.
7. Publieke sleutel info houder: De publieke sleutel (en de aanduiding voor het gebruikte

algoritme) van de houder moet altijd aanwezig zijn.

8. Identificatie uitgever: Een uniek nummer van de uitgever van het certificaat (de CA).
9. Identificatie houder: Een uniek nummer van de houder van het certificaat.
10. Extensies: Hier worden optionele en standaardextensies aangegeven. De X.509-certificaatextensies worden door zowel Netscape als Microsoft Internet Explorer op een verschillende manier geïmplementeerd.

- Intrekken van certificaten:

  Een Certficate Revocation List (CRL) is een datastructuur, die certificaten bevat die niet meer geldig zijn. De integriteit en authenticiteit van de lijst worden gewaarborgd doordat deze gesigneerd is. De instantie die de lijst tekent is dezelfde als de instantie die het certificaat heeft uitgegeven. Ter illustratie geeft Figuur 3.12 de structuur van een versie 2 CRL.



Figuur 3.12: De structuur van een versie 2 CRL

OCSP(Online Certification Status Protocol) is een eenvoudig reply-request protocol dat clients toelaat om aan een 'OCSP responder' de geldigheidsstatus op te vragen van een of meerdere certificaten. De OCSP responder geeft digitaal ondertekende antwoorden terug betreffende de status van de certificaten die in de request geïdentificeerd zijn. Dit kan 'good', 'unknown' of 'revoked' zijn. Wanneer de request niet verwerkt kan worden, wordt een foutcode gegegeneerd. OCSP is ontworpen om 'real time' antwoorden te geven op queries van clients.

X509 Certificaten en CRL worden uitgebreid besproken in [16]. Voor meer informatie over PKI wordt verwezen naar white papers op het PKI Forum [17]. Dit is een organisatie die tot doel heeft de verspreiding en aanvaarding van PKI te stimuleren.

3.2.2 Standaarden

We zullen hier de PKI specifiek voor smart cards bespreken. Met de microprocessor op de smart card worden cryptografische bewerkingen uitgevoerd en op de geheugenchip worden de geheimen bewaard. Binnen een PKI is er één entiteit die niet misbruikt mag worden: de geheime sleutel. De smart card moet zo geconstrueerd zijn dat deze veilig opgeslagen en niet misbruikt kan worden.

Momenteel zijn er twee typen smart cards die een groot toepassingsgebied hebben. Smart cards die alleen het snellere encryptiealgoritme DES aankunnen: veel gebruikte sleutellengtes zijn 64 of 128 bits. Daarnaast zijn er smart cards waarmee tevens gebruik gemaakt kan worden van een asymmetrische encryptiealgoritme (gebruikt in een PKI omgeving). De meest geavanceerde kaarten kunnen op dit moment een asymmetrische sleutellengte van 2048 bits aan.

Moderne smart cards zijn in staat zijn om intern een sleutelpaar voor een certificaat te genereren. Het voordeel hiervan is dat de geheime sleutel nooit de smart card hoeft te verlaten, dit in tegenstelling tot het genereren van het sleutelpaar door bijvoorbeeld een browser.

De standaard die de interface naar de smart card beschrijft, is de PKCS#11-standaard (zie verder voor meer uitleg over PKCS). Daarnaast wordt ook de PKCS#12-standaard gebruikt, waarmee het formaat van de geheime sleutel vastgelegd wordt voor het opslaan of transporteren van de geheime sleutel, het certificaat of andere geheimen. Daarnaast zorgt de PKCS#15-standaard ervoor dat gebruikers cryptografische tokens kunnen gebruiken om zich te kunnen

identificeren bij verschillende applicaties, die voldoen aan deze standaard. Dit in tegenstelling tot het verleden, waar iedere smart card een eigen sleutel bibliotheek nodig had.

Een Cryptographic Service Provider (CSP) definieert hoe men met geheime sleutels moet omgaan, hoe sleutels aangemaakt en vernietigd moeten worden, enz. De specifieke CSP die van toepassing is voor smart cards heet Smart Card Cryptographic Provider (SCCP). Hiermee wordt gedefinieerd hoe sleutels gegenereerd moeten worden, hoe random nummers tot stand gebracht moeten worden, hoe sleutels uitgewisseld moeten worden en dergelijke.

Windows voorziet een crypografische laag die CryptoAPI (of CAPI) heet. CAPI is een API op applicatieniveau die een uniforme interface voorziet naar verschillende CSPs. Microsoft voorziet enkele CSPs in Windows en laat leveranciers toe om plug-in CSPs te leveren die CAPI toelaat om met hun toestellen te werken. CAPI is enkel beschikbaar op Windows, bijgevolg werken alle Microsoft applicaties met een CSP en andere eventueel via PKCS.

PKCS standaarden

Naarmate cryptografie steeds meer gebruikt wordt, is het nodig dat er standaarden gedefinieerd worden, zodat de verschillende technologieën kunnen samenwerken. Want hoewel leveranciers eensgezindheid kunnen bekomen over cryptografische technieken, is de compatibiliteit tussen de verschillende leveranciers afhankelijk van een strikte toepassing van standaarden. Hiervoor is de Public Key Cryptographic Standard (PKCS) in het leven geroepen. Deze is ontwikkeld door RSA Laboratories. Zij bezitten de licenties en patenten over de RSA techniek en hebben verschillende andere patenten kunnen bemachtigen. Om het gebruik van publieke sleutel technieken te promoten en te ondersteunen, ontwikkelde RSA Laboratories de PKCS standaarden. Ze behielden hier zelf de controle over met de vermelding dat ze veranderingen/verbeteringen zouden maken als het noodzakelijk leek. Zo zijn de PKCS standaarden geen echte industrie-standaarden te noemen ondanks hun naam. Sommigen zijn nu overgegaan naar 'echte standaarden' omdat ze werden overgenomen zijn door één of meerdere standaardorganisaties (vb door de PKIX werkgroep). De standaarden die binnen een PKI belangrijk zijn volgen hieronder:

- PKCS#7 : Cryptographic Message Syntax Standard (versie 1.5) : De standaard beschrijft een algemene syntax voor data waarop men cryptografie kan toepassen, zoals digitale handtekeningen en digitale enveloppen. Een afgeslankte (gedegenereerde) versie van de syntax zorgt voor een manier om certificaten en certificate-revocation lists te verspreiden.

- PKCS#10 : Certification Request Standard (versie 1.7): De standaard beschrijft een syntax voor een aanvraag tot certificatie. Een aanvraag bestaat uit een 'distinguished name' (DN), een publieke sleutel en optioneel een set van attributen die gezamenlijk gesigneerd zijn door de entiteit die de certificatie aanvraagt. De mogelijkheid om een set van attributen te bevatten is tweevoudig: enerzijds andere informatie voorzien over een gegeven entiteit of een 'challenge password' waardoor de entiteit later certificaat revocatie kan aanvragen. Anderzijds om attributen te voorzien die X.509 Certificaten kunnen inhouden.

- PKCS#11: Cryptographic Token Interface (cryptoki) (versie 2.20): De standaard specifieert een API (Cryptoki) voor toestellen die cryptografische informatie bevatten en cryptografische functies uitvoeren (= 'cryptografisch token'). Cryptoki volgt een object gebaseerde aanpak om onafhankelijk te zijn van de gebruikte technologie (elk soort toestel) en het delen van resources mogelijk te maken (verschillende applicaties die toegang hebben tot verschillende toestellen).

- PKCS#12 : Personal Information Exchange Syntax Standard (versie 1.0): Deze standaard specifieert een draagbaar formaat om de private sleutels, certificaten, ... van een gebruiker op te slaan en te transporteren.

- PKCS#15: Cryptographic Token Information Format Standard (versie 1.1): Deze standaard verzekert dat gebruikers cryptografische tokens kunnen gebruiken om zichzelf te identificeren aan verschillende applicaties (die de standaarden ondersteunen) onafhankelijk van de cryptoki provider van de applicatie.

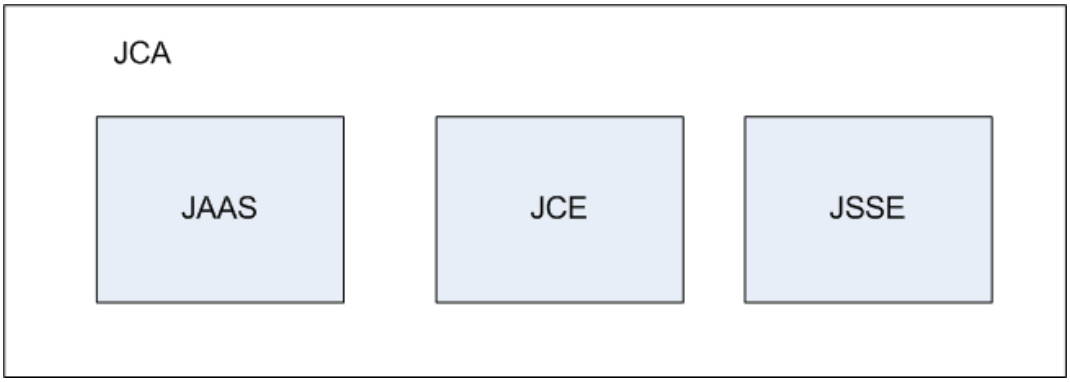
Voor meer informatie wordt verwezen naar de RSA website [18].

3.2.3 Toegepast in Java

Een overzicht van de securitygerelateerde packages in Java wordt gegeven in Figuur 3.13.

JCA

De Java Cryptography Architecture omvat de delen van de Java 2 SDK Security API die te maken hebben met cryptografie, evenals de set van conventies en specificaties zoals verder vermeldt. Het bevat een 'provider' architectuur die meerdere cryptografische implementaties toelaat. De JCA is ontwikkeld rond volgende principes:



Figuur 3.13: Java Cryptography Architecture

- Implementatie-onafhankelijkheid en - interoperabiliteit
- Algoritme-onafhankelijkheid en - uitbreidbaarheid

Algoritme-onafhankelijkheid wordt bekomen door het definiëren van cryptografische 'engines' (services) van klassen die de functionaliteit van deze engines leveren. Voorbeelden van deze 'engine' klassen zijn : MessageDigest, Signature, KeyFactory, ... Ze definiëren API methoden die applicaties toelaten een specifiek type van cryptografische service te verkrijgen. De interfaces voor applicaties die een engine klasse voorzien, zijn geïmplementeerd als een Service Provider Interface (SPI). Dus voor elke engine klasse is er een overeenkomstige abstracte SPI klasse die SPI methoden definieert die de CSP's moeten implementeren.

Implementatie-onafhankelijkheid wordt bereikt door de 'provider'architectuur.

Implementatie-interoperabiliteit betekent dat verschillende implementaties kunnen samenwerken, elkaars sleutels gebruiken of elkaars handtekeningen verifiëren.

Algoritme-uitbreidbaarheid tenslotte betekent dat nieuwe algoritmen die in één van de ondersteunde engine klassen passen, gemakkelijk toegevoegd kunnen worden.

- Cryptographic Service Providers:
  De Java Cryptography Architecture introduceert de notie van CSP ('provider'). Deze term verwijst hier naar een package (of een set van packages) die een concrete implementatie leveren voor een subset van de cryptografische aspecten van de Security API. Bijvoorbeeld: digitale handtekening-algoritmen, message digest algoritme en key conversie objecten. Een programma kan simpelweg een specifiek object (bijvoorbeeld een Signature object) aan-

vragen die een specifieke service implementeert (zoals het DSA handtekening algoritme) en een implementatie krijgen van een van de geïnstalleerde providers.

Sun's versie van de Java Runtime Environment komt standaard met een default provider, SUN genaamd. Andere JRE's hoeven niet noodzakelijk de SUN provider te leveren.

- Key Management:
  Een databank, genaamd 'keystore', kan gebruikt worden als een opslagplaats voor sleutels en certificaten. Een keystore is beschikbaar voor applicaties die dit nodig hebben voor authenticatie of signeerdoeleinden.

Voor meer informatie wordt verwezen naar [19].

JCE

De Java Cryptography Extension (JCE) breidt de JCA API uit en voorziet een raamwerk en implementaties voor encryptie, sleutelgeneratie en key agreement en MAC algoritmen (= Message Authentication Code of een hash). De ondersteuning voor encryptie omvat symmetrische, asymmetrische en blok cijfers. JCE is gebaseerd op dezelfde design principes als JCA: implementatie-onafhankelijkheid en, waar mogelijk, algoritme-onafhankelijkheid. Het gebruikt dezelfde 'provider' architectuur. Providers die gesigneerd zijn door een 'trusted entity' en nieuwe algoritmen kunnen in het JCE framework geplugged. De JCE API omvat:

- Symmetric bulk encryptie, zoals DES, RC2, en IDEA
- Symmetric stream encryptie, zoals RC4
- Asymmetric encryptie, zoals RSA
- Password-based encryptie (PBE)
- Key Agreement
- Message Authentication Codes (MAC)

Voor meer informatie wordt verwezen naar [20].

Java biedt ook ondersteuning voor X.509 certificaten, CRL's en in de laatste versie (1.5) ook voor OCSP. Er is ook een Cryptographic Provider voor PKCS#11 nl. de SUN PKCS#11 provider die eveneens pas sinds 1.5 aanwezig is.

JSSE

Java Secure Socket Extension levert een raamwerk en een implementatie voor Secure Socket Layer (SSL) en Transport Layer Security (TLS). Dit zijn 2 gestandardizeerde protocollen om een veilige verbinding over het internet te implementeren. In het hoofdstuk Implementatie wordt dit verder toegelicht.

Voor meer informatie wordt verwezen naar [21].

3.3 Authenticatie

Authenticatie is de mogelijkheid om te verifiëren en te valideren of een individu effectief de persoon is met wie men wil communiceren of een transactie uitvoeren. Er zijn slechts drie mogelijkheden om een individu te authenticeren:

- Knowledge factor: iets dat een persoon weet
- Possession factor: iets dat een persoon bezit
- Biometrische factor: iets dat een person 'is' (vingerafdruk, irisscan, ...)

Smart cards kunnen voor een combinatie van twee elementen zorgen: Een smart card bezit je en bij het invoeren kan een pincode gevraagd worden. Een PKI kan dan ook gemakkelijk geïmplementeerd worden op een smart card. Als we een smart card vergelijken met andere authenticatie tokens dan zorgt deze voor de beste combinatie van flexibiliteit, security en kosten. In het hoofdstuk 'Implementatie' zal verder uitgeweid worden over de gebruikte authenticatie in samenwerking met de elektronische Identiteitskaart.

3.3.1 Toegepast in Java

JAAS

De Java Authentication en Authorization Service kan voor twee doeleinden gebruikt worden:

- authenticatie van gebruikers: bepalen wie momenteel Java code aan het uitvoeren is (applicatie, applet, bean, servlet, ...)
- authorisatie van gebruikers: verzekeren dat ze de vereiste toegangscontrolerechten hebben

JAAS implementeert een Java versie van het standaard Pluggable Authentication Module (PAM) raamwerk. Dit houdt in dat het een architectuur ondersteunt die toelaat om gepaste authorisatie services in te pluggen volgens de nodige security vereisten.

Voor meer informatie wordt verwezen naar [22]

3.3.2 Technologie Scan

Op basis van deze gegevens werd een Technology Scan uitgevoerd (zie appendix) die een selectie bevat van vijf smart cards. Deze worden besproken en vergeleken op basis van een aantal eigenschappen. Er werd dan besloten om verder te gaan met de elektronische identiteitskaart aangezien er reeds een PKI op deze kaart aanwezig is en alle inwoners in de toekomst een kaart zullen hebben (wat de kost van de aankoop uitspaart).

3.4 Elektronische Identiteitskaart

3.4.1 Algemeen

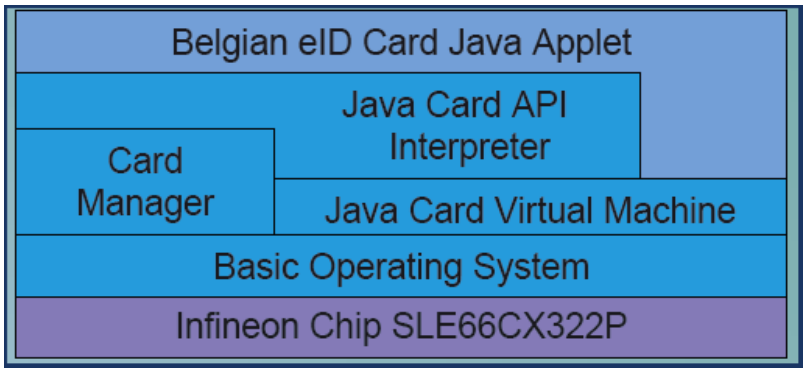
In de context van de elektronische identiteitskaart hoort men soms horen spreken van eID (Elektronische IDentiteitskaart) of BelPIC (Belgian Electronic Personal Identification Card). BelPIC verwijst naar de naam van het project dat resulteerde in de eID. Fedict (Federale Overheids Dienst ICT) was bij de realisatie van de eID vooral verantwoordelijk voor de aspecten veiligheid en gebruiksvriendelijkheid. De firma Steria heeft de hele infrastructuur geïmplementeerd zoals door de conceptstudie in 2000 werd ontworpen. Deze studie heeft ook beslist om de SIS kaart en het rijbewijs niet te integreren. Zetes werd gekozen als producent van de elektronische identiteitskaart en van de processorchip op de kaart. Het uitreiken, beheren en controleren van de certificaten voor de kaart werd uitbesteed aan het Certipost consortium, dit is een samenwerking tussen de Belgische Post en Belgacom. Certipost is hier de 'Certificate Authority'. De voornaamste onderaannemer voor dit project is Ubizen, met zijn Globalsign product voor certificaatbeheer.

Smart card specificaties

De kaart zelf is een Cryptoflex Java Card 32K van de firma Axalto.

- CPU (processor): 16 bit Micro-controller
- Crypto-processor:
  - 1100 bit Crypto-Engine (RSA)
  - 112 bit Crypto-Accelerator (DES)
- ROM (OS): 136 kB (GEOS Java Virtual Machine)
- EEPROM (Applic + Data): 32 KB (Cristal Applet)
- RAM (memory): 5 KB

Figuur 3.14 geeft een overzicht van de verschillende lagen op de eID.



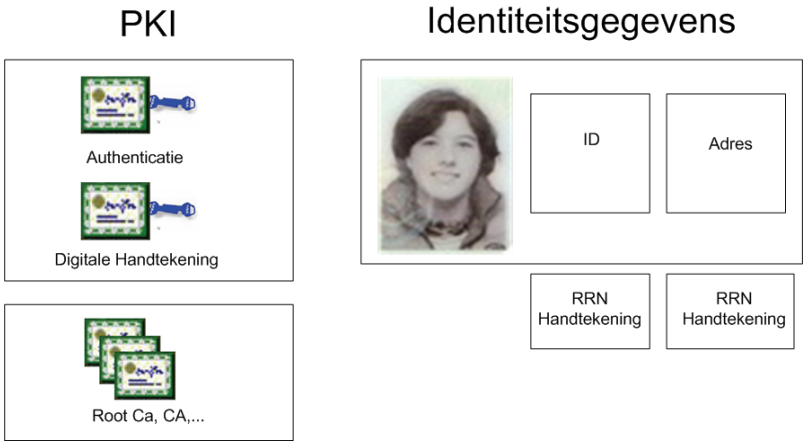
Figuur 3.14: Overzicht van de lagen op de eID

De kaart gebruikt on-board key generatie wat inhoudt dat de private sleutels de kaart niet kunnen verlaten. De bezitter moet zijn PIN code ingeven om deze sleutels te gebruiken. Dit voorkomt dat de private sleutel in verkeerde handen komt.

Zoals verder besproken, stelt de overheid gratis software ter beschikking om bepaalde elementen te beheren. De kaart zelf kan enkel beheerd worden door de Belgische overheid. Dit houdt in dat de identiteits - en adresgegevens enkel kunnen geschreven worden door het RijksRegister (RRN= RijksRegister/registre National). Ze weigert elke update poging van andere partijen. De eID heeft een geldigheid van 5 jaar.

3.4.2 Inhoud van de kaart

Een overzicht wordt gegeven in figuur 3.15

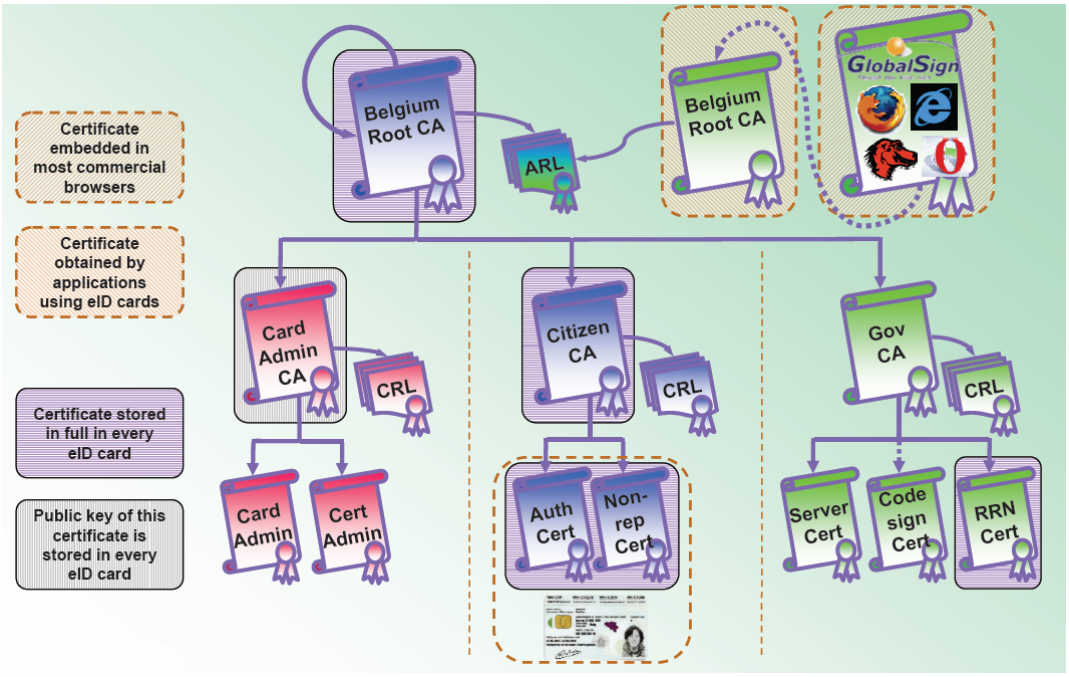


Figuur 3.15: Inhoud van de eID

Gegevensbestanden

- Identiteitsbestand (∼160 bytes)
  - Chip-specifiek: Chip nummer
  - Burger-specifiek: naam, twee voornamen, eerste letter derde voornaam, RRN nummer, nationaliteit, ... en SHA-1 hash van de foto van de burger.
  - Kaart-specifiek: Kaart nummer, geldigheidsperiode, document type, ...
- Digitale handtekening op identiteitsbestand uitgegeven door het RRN
- Adresbestand van de burger (∼120 bytes): Straat, nr, postcode, gemeeente
- Digitale handtekening op adres en identiteitsbestand uitgegeven door het RRN
- JPEG foto van burger (∼3 Kbytes)

Het adresbestand wordt afzonderlijk gehouden omdat het kan veranderen binnen de geldigheidsperiode van de kaart. Het RRN signeert de adresgegevens samen met de identiteitsgegevens om de link tussen beide te garanderen. België heeft beslist om een foto te gebruiken als biometrische feature. Deze is (indirect) gesigneerd door de RRN doordat zijn hash deel uitmaakt van de identiteitsfile.



Figuur 3.16: Hiërarchie model

PKI

De Belgische eID kaart bevat drie verschillende 1024-bit RSA private signeersleutels, waarvan twee voor de burger met bijhorend certificaat:

- Authenticatie burger: X.509v3 certificaat
- Onweerlegbaarheid: X.509v3 certificaat

De derde heeft geen certificaat en is een sleutel die gebruikt wordt om zich te identificeren bij de Belgische overheid. Dit wanneer de kaart communiceert met het RRN voor wederzijdse authenticatie (bijvoorbeeld adres updaten). De RRN houdt een databank bij met een kopie van de publieke sleutels om de handtekeningen te verifiëren. De smart card zelf is niet in staat om de cryptografische hashwaarde te berekenen op basis waarvan hij een handtekening produceert: een eID kaart signeert digitaal de 16 of 20 bytes die het ontvangt van een externe applicatie. De kaart krijgt tijdens de initialisatie van de signeersessie opgegeven wat hij mag verwachten (16 of 20) afhankelijk van het padding type (MD5metRSA of SHA1metRSA). Het is onmogelijk om de kaart een handtekening te laten genereren gebaseerd op andere informatie dan deze 16 of 20 bytes. Er zijn geen concrete plannen om decryptie-functionaliteiten te integreren in de kaart.

De kaart houdt ook enkele specifieke overheidscertificaten bij. De overheid heeft beslist om voor haar CA certificaten een 2048 bit RSA sleutel te gebruiken. De certificaten van de kaarthouder zelf en de RRN hebben 1024-bit RSA publieke sleutels. Figuur 3.16 toont dat de Root CA de andere CA certificaten certificeert bijvoorbeeld voor kaartadministratie en overheidsspecifieke servers. Het sleutelpaar dat gebruikt wordt voor de zelf-gesigneerde Belgium Root CA is ook gecertificeerd door de commeriële CA GlobalSign. Zo kan de certificaatketen gevalideerd worden (certificaat burger → Citizen CA Certificaat → Belgium Root CA). Elke CA implementeert ook CRL's zodat men dus kan controleren of een certificaat nog geldig is. De Belgium Root CA wordt gecontroleerd door een ARL (Authority Revocation List) wat analoog is aan een CRL maar voor een CA. Aangezien elke kaart een kopie heeft van de Belgium Root CA kan elke kaart gebruikt worden als een 'trusted source': elke gebruiker kan de 'chain of trust' verifiëren binnen het Belgische PKI systeem door de Belgium Root CA te laden van zijn/haar smart card. Daarom kan het hele eID project gezien worden als een PKI uitgestrekt over het ganse land met sterke gebruiker-authenticatie bij uitgifte, aangezien elke burger zichzelf moet komen presenteren op het gemeentehuis.

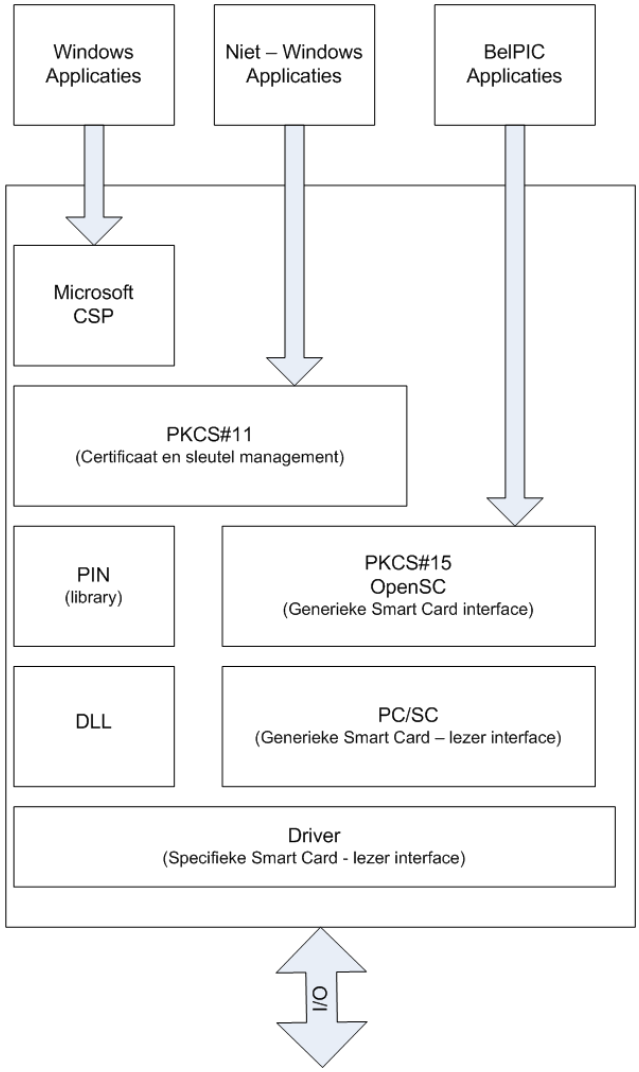
Voor meer informatie wordt verwezen naar [23] en [24]

3.4.3 Software

Middleware

Een mooi overzicht van de middleware wordt gegeven in figuur 3.17. We onderscheiden drie delen:

- PKCS#15 bestandssysteem voor ID applicaties:
  - Alle eID verwante data (certificaten, foto, adres, identiteitsbestand, ...)
  - geen sleutel management
- PKCS#11 standaard interface voor crypto tokens:
  - abstractie van de signeerfuncties (authenticatie, digitale handtekening)
  - toegang tot certificaten
  - beschikbaar voor Unix, Windows, MacOSX
- CSP voor Microsoft platformen:



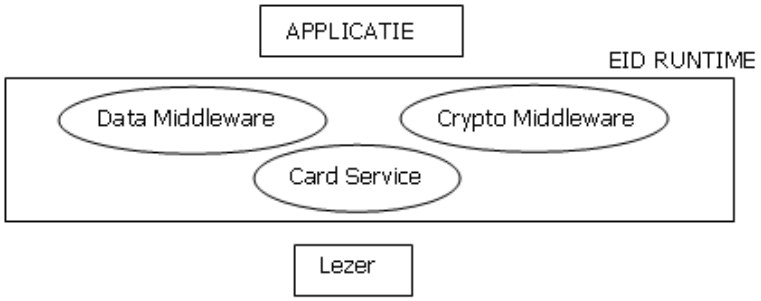
Figuur 3.17: eID middleware

- sleutels en certificaten enkel beschikbaar via CryptoAPI van Microsoft
- authenticatie en handtekening
- geintegreerd in explorer, outlook, ...

Hier moet nog een kleine opmerking gemaakt worden: normaal wordt een CSP geïmplementeerd voor een specifiek smart card type want meestal wordt de CSP door de smart card leverancier voorzien. De Belgische middleware heeft een andere aanpak gekozen waardoor de CSP niet de specifieke smart card interface implementeert, maar de implementatie via een PKCS#11 interface gebeurt. Indien later het type smart card verandert, hoeft de CSP implementatie niet aangepast te worden. Zoals reeds eerder gezegd implementeert de officiële middleware een

gemodificeerde versie van OpenSC (in de PKSC#15 laag) die een specifieke driver bezit voor de eID. Dit is de BelPIC PCSC Service (die in het geheugen geladen wordt tijdens het opstarten). Indien de overheid in een later stadium van kaart wil veranderen, moet deze driver vervangen worden. De kaart zelf wordt nog niet ondersteund in het OpenSC project. De middleware wordt uitvoerig besproken in [25].

Toolkit



Figuur 3.18: eID Runtime Software

De middleware is gebundeld in een toolkit die eID Runtime heet. Ze bestaat uit 3 delen:

- Card Service: deze component zal iedere ingebrachte kaart in de chipkaartlezer detecteren en, indien het over een identiteitskaart gaat, al de beschikbare functies aan de middleware presenteren.
- Data Middleware: deze component zal communiceren met de Card Service om de identiteitgegevens in de kaart op een gestandaardiseerde manier te presenteren aan de toepassingen die toegang tot deze informatie willen krijgen.
- Crypto Middleware: deze component zal communiceren met de Card Service om de cryptografische functies te activeren voor de authenticatie en de handtekening die de invoer van een PIN code vereisen.

De eID Runtime is enkel beschikbaar voor Windows. Voor de platformen waarvoor de eID Runtime nog niet beschikbaar is bestaat er een vereenvoudigde versie met alleen de Crypto Middleware(d.i.: Linux, MacOS, Solaris)

Meer informatie is te vinden in documenten [26] en [27].

3.4.4 Conclusie

Samengevat komen de functionaliteiten hierop neer:

- Data Capture
- Authenticatie
- Digitale Handtekening

Als we de documentatie bekijken biedt de officiële site van de overheid heel wat PDF's met uitleg voor zowel de techneut als de leek. De algemene indruk is dat de documentatie zeer verspreid is. Er is bijvoorbeeld de site van Fedict en daarnaast heb je nog certipost. Dit heeft als gevolg dat op de ene pagina oudere versies van documenten staan terwijl er verder al geüpdate exemplaren staan. Daarenboven werken sommige links niet en zijn er verschillende pagina's met zelfde uitleg. In het document voor 'softwareontwikkelaars', waar uitleg gegeven wordt hoe je de middleware kunt gebruiken, wordt er amper verwezen naar andere documenten waaruit je informatie nodig hebt. De middleware zelf staat ook nog niet volledig op punt. Naast de reeds vermelde problemen met documentatie, gaat het allemaal vrij traag. Daarnaast werken bepaalde methodes niet. Na wat opzoekwerk bleken er nog mensen te zijn met hetzelfde probleem en wist men niet direct een oplossing. In de te downloaden bestanden zitten er ook enkele voorbeeldbestanden voor verschillende programmeertalen (Java en Visual Basic) maar bijvoorbeeld in het Visual Basic bestand zit zelfs een syntaxfout. Voor iets dat de huidige identiteitskaart vervangt, is dit toch niet echt een goede reclame. Er wordt gewerkt aan nieuwere versies die de problemen zouden moeten oplossen maar er is nog geen release datum van bekend. Er werd dan overgeschakeld op een 'onofficiële' opensource implementatie ([28]). Deze implementatie is een Java versie die gebruik maakt van jpc/sc om te communiceren met de smart card. Dit is onofficieel en er wordt ook niet veel documentatie voorzien, maar er zitten enkele voorbeeldbestanden bij die veel duidelijk maken en werken.

Hoofdstuk 4

Architectuur

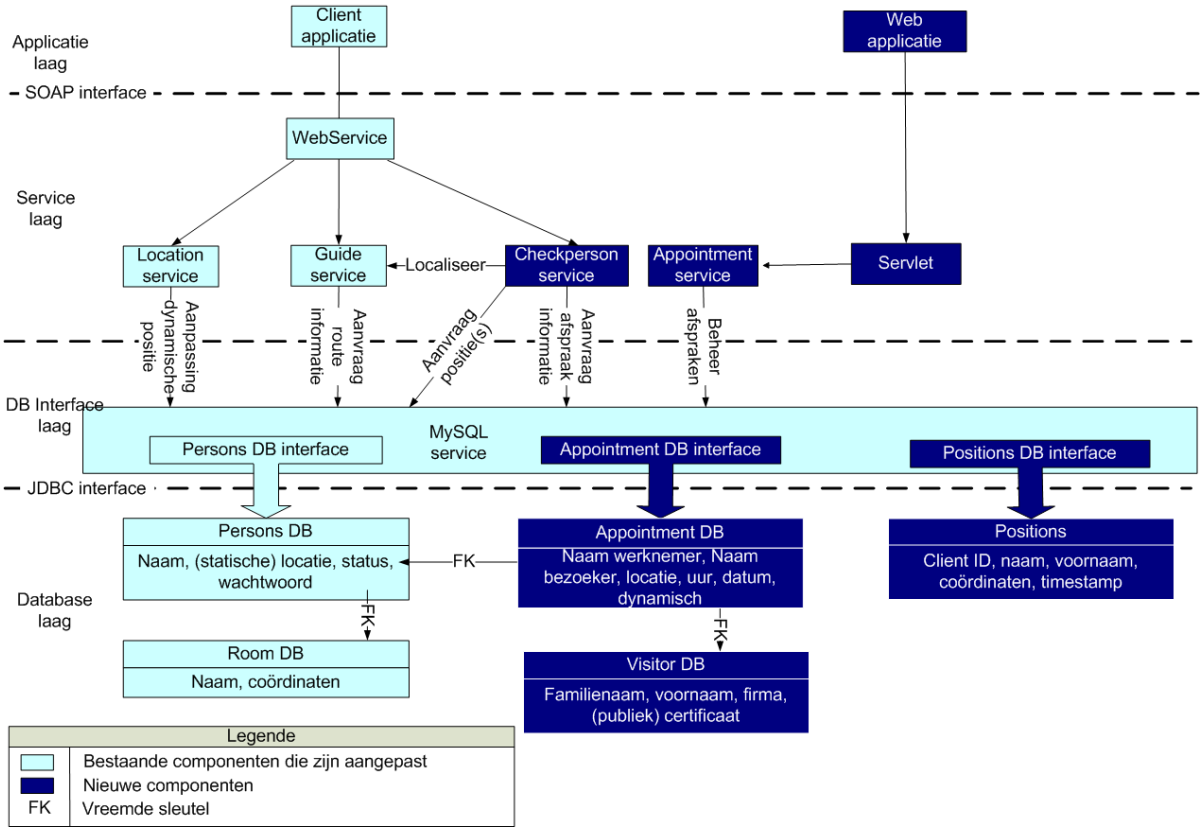
4.1 Concept

Er wordt gewerkt met een client-server architectuur waarbij de server alle 'services' aan de client moet aanbieden. Aangezien de bezoeker(=client) een mobiel toestel (PDA) zal gebruiken, gebeuren alle berekeningen zoveel mogelijk aan de serverkant. Er is hierbij gebruik gemaakt van een IBCN - versie van het OSGi platform nl. IGSO (Interactive Gateway for Service Operations) (zie 5.1.1). Het protocol dat de communicatie tussen client en server verzorgt is Simple Object Acces Protocol (SOAP) (zie 5.1.3). Om alle gegevens bij te houden, wordt een MySQL databank gebruikt. Bij aanvang van de thesis zijn al een webservice en bijhorende bundels op het IGSO platform ontwikkeld samen met een client applicatie. In samenspraak met de begeleiders werd een aanpassing van de bestaande architectuur ontworpen. Een algemeen overzicht wordt gegeven in Figuur 4.1.

4.1.1 Servicelaag - Server

GuideService

Deze service zorgt voor alle padgegevens om een route weer te geven. Een kaart in een gebouw wordt omgezet naar een graafvoorstelling. Het kortste pad wordt dan berekend aan de hand van deze graaf. Dit was reeds geïmplementeerd.



Figuur 4.1: Ontwerp architectuur

AppointmentService

Een werknemer kan met deze service (via de Servlet) zijn afspraken beheren. Elke werknemer heeft een wachtwoord (dat aangepast kan worden) en na het inloggen krijgt hij een overzicht van de mogelijkheden. Hij kan afspraken bekijken, bewerken, aanmaken, ... Er kan ook een overzicht van de reeds in het systeem ingebrachte bezoekers opgevraagd worden, zodat niet telkens opnieuw alle gegevens van de bezoeker ingebracht moeten worden.

Servlet

De Servlet maakt gebruik van de AppointmentService om de afspraken van een werknemer te beheren. De functies van de AppointmentService worden via deze service weergegeven in de vorm van een website.

CheckPersonService

De belangrijste functie is om de afspraken van een bezoeker terug te geven. Er kunnen eventueel ook personeelsleden weergegeven worden afhankelijk van de status van de gebruiker. Deze status wordt verder besproken onder 'Hiërarchisch User Profiel'. De service zorgt ook voor de certificaten indien een bezoeker nog niet geregistreerd is. Dit wordt verder uitgelegd in het hoofdstuk Implementatie.

WebService

De WebService zorgt voor het contact met de buitenwereld. Ze bevat een overzicht van de methodes van alle bundels en delegeert aanvragen naar de juiste bundel. Bundels worden toegelicht in het hoofdstuk Implementatie. Eventuele resultaten van de methodes worden ook teruggestuurd.

LocationLibrary - LocationService

Dit is een bibliotheek met nuttige klassen voor de andere bundles. Ze bevat onder andere de klassen Position, LocationService(interface), Histogram en Client (bevat alle gegevens over een client). Alle algoritmes moeten de LocationService interface implementeren. De methode calculatePosition omvat het eigenlijke algoritme.

MySQLService

Deze service handelt alle interacties met de databank af. Ze zorgt voor een interfacelaag zodat de services niet alle databankcode moeten bevatten.

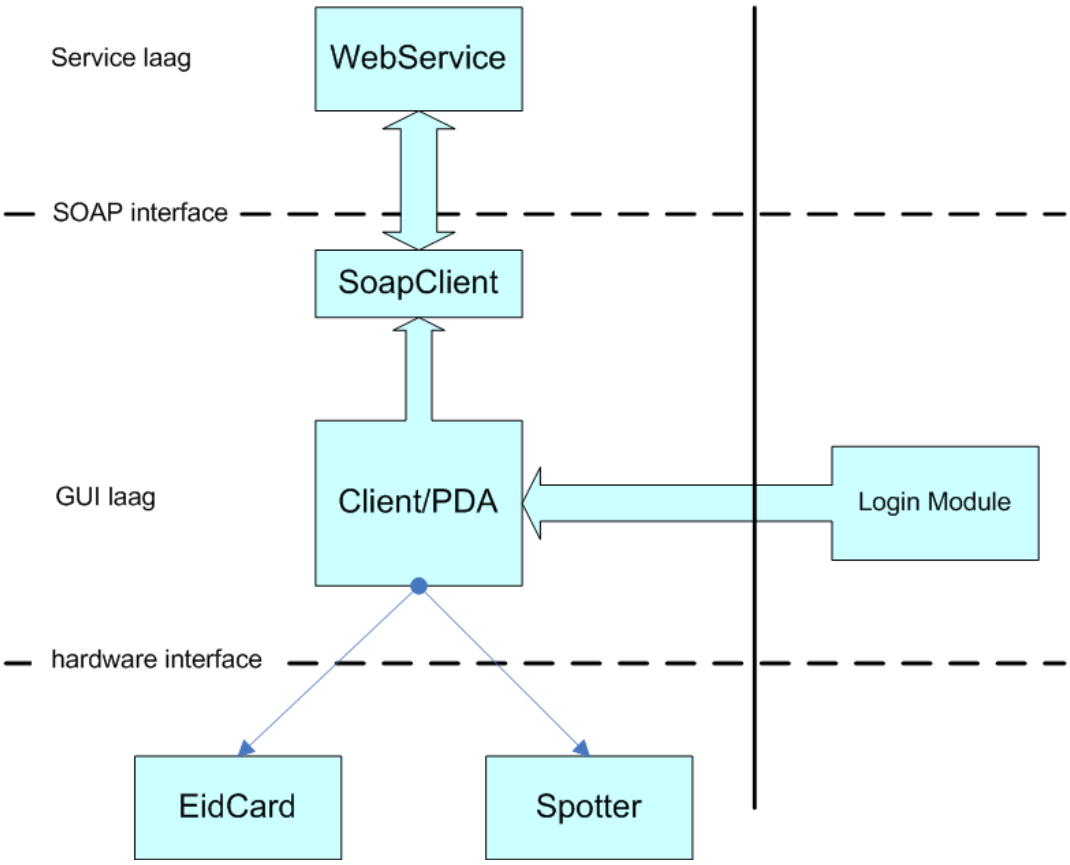
4.1.2 Databanklaag

- appointments: Bevat alle informatie omtrent een afspraak en werkt met een vreemde sleutel naar de visitor databank
- visitor: Elke bezoeker wordt hier met zijn gegevens opgeslagen. Een werknemer kan deze gegevens meegeven tijdens het invoeren van zijn afspraak.
- persons: Alle personeelsleden zijn hier in opgeslagen en er wordt gewerkt met een vreemde sleutel naar de rooms databank.

- rooms: Hier worden de room-nummers afgebeeld op de coördinaten.
- positions: De gegevens van dynamische gebruikers worden hier (tijdelijk) opgeslagen. Dit zal uitgelegd worden bij 'Dynamische gebruikers'.

4.1.3 Applicatielaag - Client

Een overzicht van de architectuur van de client wordt gegeven in Figuur 4.2.

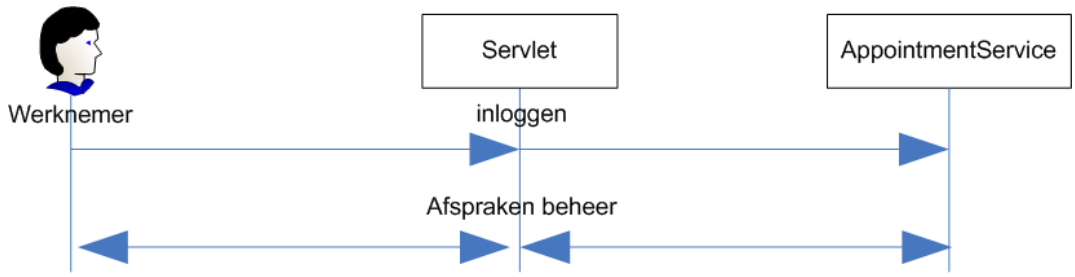


Figuur 4.2: Architectuur applicatielaag

Hiervan werden 2 versies ontwikkeld: 1 voor op PC en een andere voor op een PDA. De PDA versie is voor de bezoeker bedoeld. In deze versie zijn er enkele aanpassingen gebeurd, maar deze worden besproken in het hoofdstuk Implementatie. De client maakt verbinding via SOAP naar de WebService op de server en deze zorgt voor de afhandeling van alle aanvragen. EidCard is een abstractie van de eID. Het behandelt alle connecties van en naar de kaart. Spotter wordt gebruikt bij de lokatiebepaling en maakt gebruik van de WiFi-kaart. De Login Module is een

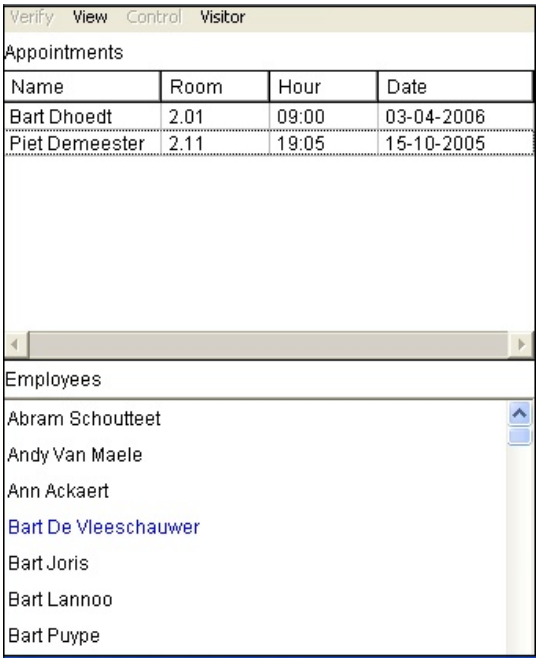
extra klasse die gebruikt wordt tijdens het inloggen met de PDA. Dit wordt, samen met de Spotterklasse, verder besproken in het hoofdstuk Implementatie.

Algemeen



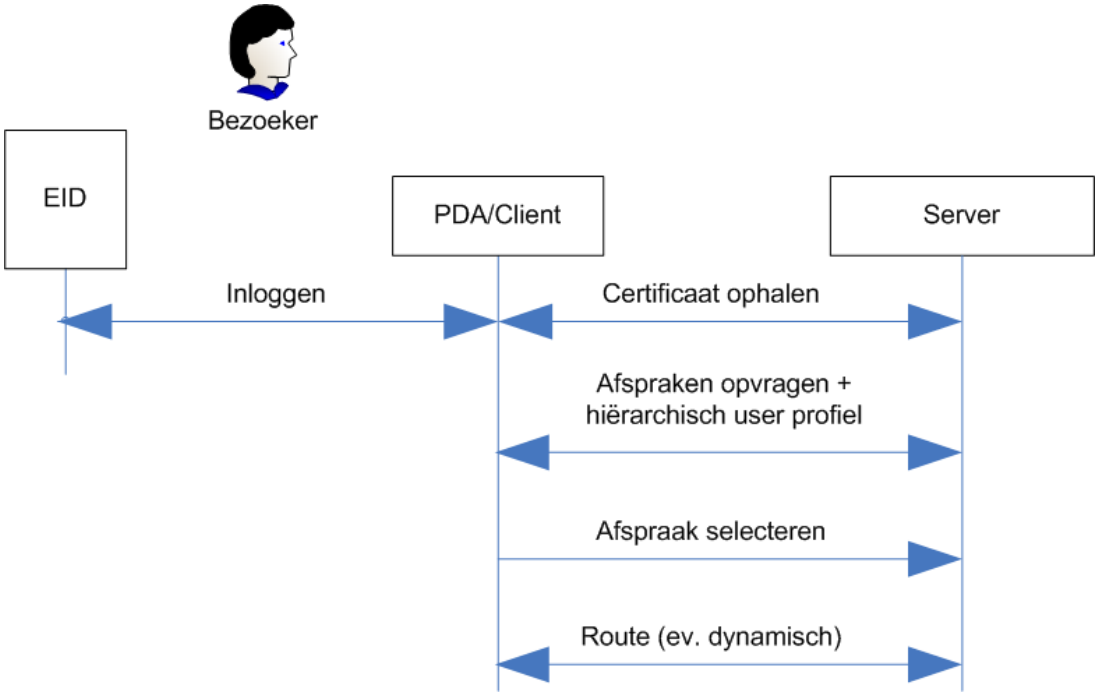
Figuur 4.3: Scenario personeelslid

Een werknemer surft naar de servlet om zijn afspraken te beheren. Het scenario voor een personeelslid wordt gegeven in figuur 4.3. De bedoeling is dat een bezoeker bij het binnenkomen van het gebouw een PDA krijgt. De bezoeker moet inloggen (meer uitleg in het hoofdstuk Implementatie). Dan krijgt hij de mogelijkheid om zijn afspraken op te vragen. Figuur 4.4 toont hiervan een voorbeeld. De bezoeker krijgt hier eveneens een overzicht van de personeelsleden die



Figuur 4.4: Een overzicht van een bezoeker zijn afspraken

hij eventueel mag 'zien' volgens het 'Hiërarchisch User Profiel'. In een volgende stap selecteert de bezoeker zijn gewenste afspraak of personeelslid en vraagt de route op. Een personeelslid kan opgeven of 'zijn afspraak' hem dynamisch mag lokaliseren. Dit wordt uitgelegd bij 'Dynamische gebruikers'. Het scenario voor een bezoeker wordt gegeven in Figuur 4.5, een voorbeeld in Figuur 4.6.



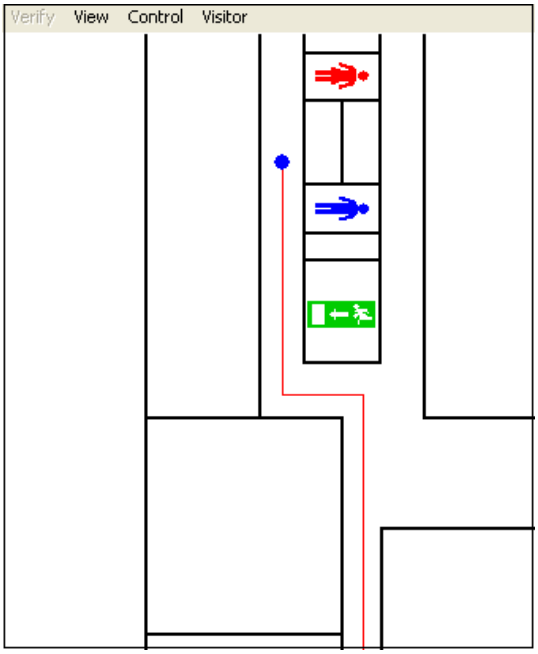
Figuur 4.5: Scenario bezoeker

Hiërarchisch User Profiel

Elk personeelslid heeft een eigen 'status'. Er wordt een onderscheid gemaakt tussen personeel en oversten: een 'gewone' werknemer heeft status 1 en iedere hiërarchisch hogere krijgt een hogere status. Zodoende mag een werknemer iedereen zien met een status kleiner of gelijk aan zijn eigen status. Anderzijds is er een onderscheid tussen bezoekers en personeel. Een bezoeker krijg status 0 en kan zo enkel personeelsleden zien met wie hij een afspraak heeft.

Dynamische gebruikers

Wanneer iemand inlogt in het systeem, wordt zijn positie opgeslagen in de databank 'positions'. Dit is een soort 'dynamische gebruikersdatabank'. De eigen positie wordt om de seconde her-



Figuur 4.6: De route naar de locatie van de afspraak

berekend en doorgestuurd. Stel, een werknemer is online en heeft opgegeven dat hij dynamisch gelokaliseerd mag worden, dan zal de route naar die werknemer om de 5 seconden geüpdatet worden. Zo wordt de route telkens aangepast aan zowel bezoeker als werknemer. Indien de werknemer 'offline' is, wordt de statische route weergegeven. Dit is de plaats van afspraak. Indien een gebruiker te lang (30 seconden) zijn positie niet meer geüpdatet heeft, wordt hij verwijderd uit de databank.

Hoofdstuk 5

Implementatie

5.1 Gebruikte technologieën

5.1.1 OSGi

OSGi (Open Services Gateway Initiative) is een op Java gebaseerd platform dat services aanbiedt. Het dient als een container voor 'bundles' die at runtime kunnen geladen, gestopt of geüpdatet worden. Een bundel is een soort plugin die functionaliteiten (=services) aanbiedt aan andere bundels en/of gebruik maakt van andere services. Wij maken gebruik van een door IBCN ontwikkelde versie van dit platform: IGSO (Interactive Gateway for Service Operations).

Voor meer informatie wordt verwezen naar de OSGi Alliance website [29].

5.1.2 MySQL

MySQL is een open source relationele database (RDB), die gebruik maakt van SQL. Het pakket dat wij gebruiken bestaat uit :

- MySql System Tray Monitor: een serverprogramma dat zich in de taakbalk installeert en kan gebruikt worden om de databanken op te starten.
- MySql Control Center: een programma dat verbinding maakt met de opgestarte databanken en waarmee de data die zich in deze databanken bevindt kan gevisualiseerd worden. Het is ook mogelijk aan de hand van SQL-query's bewerkingen zoals tabellen aanmaken, rijen toevoegen, gegevens updaten, enzovoort uit te voeren.

- MySql administrator: een administratieprogramma

Voor meer informatie wordt verwezen naar de MySQL website [30].

5.1.3 SOAP

SOAP werd gebruikt om de communicatie tussen de client- en de servercomponent te onderhouden. SOAP is een protocol dat de communicatie verzorgt door XML-boodschappen uit te wisselen, gebruikmakend van het gewone HTTP-protocol. Dankzij het gebruik van het HTTP-protocol heeft SOAP geen probleem om met firewalls te werken. SOAP wordt vooral gebruikt in het domein van de webservices. Een webservice maakt het mogelijk om vanop afstand (meestal over het Internet) vanaf een client-computer een dienst op te vragen aan een server, bijvoorbeeld het maken van een berekening, het leveren van gegevens of (in dit geval) een locatiebepaling. Services worden beschreven met behulp van WSDL (Web Services Definition Language). Een WSDL-document is een XML-document, bestaande uit een verzameling definities. Voor registratie en opsporing van webservices (een soort telefoonboek voor webservices dus) wordt gebruik gemaakt van een UDDI-database (Universal Description, Discovery and Integration) waarin de WSDL-documenten en aanvullende informatie worden opgeslagen.

Een SOAP-boodschap is samengesteld uit:

- Een header: bevat relevante informatie over de boodschap, zoals de datum waarop ze verzonden is, authenticatie informatie, ...
- De body: de boodschap zelf.

Een groot voordeel ten opzichte van andere communicatieprotocollen over HTTP zoals Servlet-communicatie is dat SOAP object-georiënteerd is. Een van de nadelen is de performantie. Door de lengte van de te versturen boodschappen ligt deze een pak lager dan de performantie bij technologieën zoals RMI of IIOP. Wanneer er enkel kleine boodschappen worden verzonden is dit natuurlijk geen probleem.

Voor meer informatie wordt verwezen naar de website van het World Wide Web Consortium waar SOAP gespecifieerd wordt. [31].

5.1.4 SSL

SSL (Secure Sockets Layer) is een protocol om een veilige communicatie over het internet op te zetten. Het maakt gebruik van asymmetrische cryptografie (publieke sleutels) om een set van gedeelde sleutels uit te wisselen en gebruikt dan symmetrische encryptie om data uit te wisselen. De gedeelde sleutels worden zowel gebruikt voor encryptie als authenticatie. Typisch zal de server geauthenticeerd zijn, terwijl de client onbekend blijft.

Voor meer informatie wordt verwezen naar de website van Netscape waar SSL gespecifieerd wordt. [32].

5.2 Locatiebepaling en padberekening

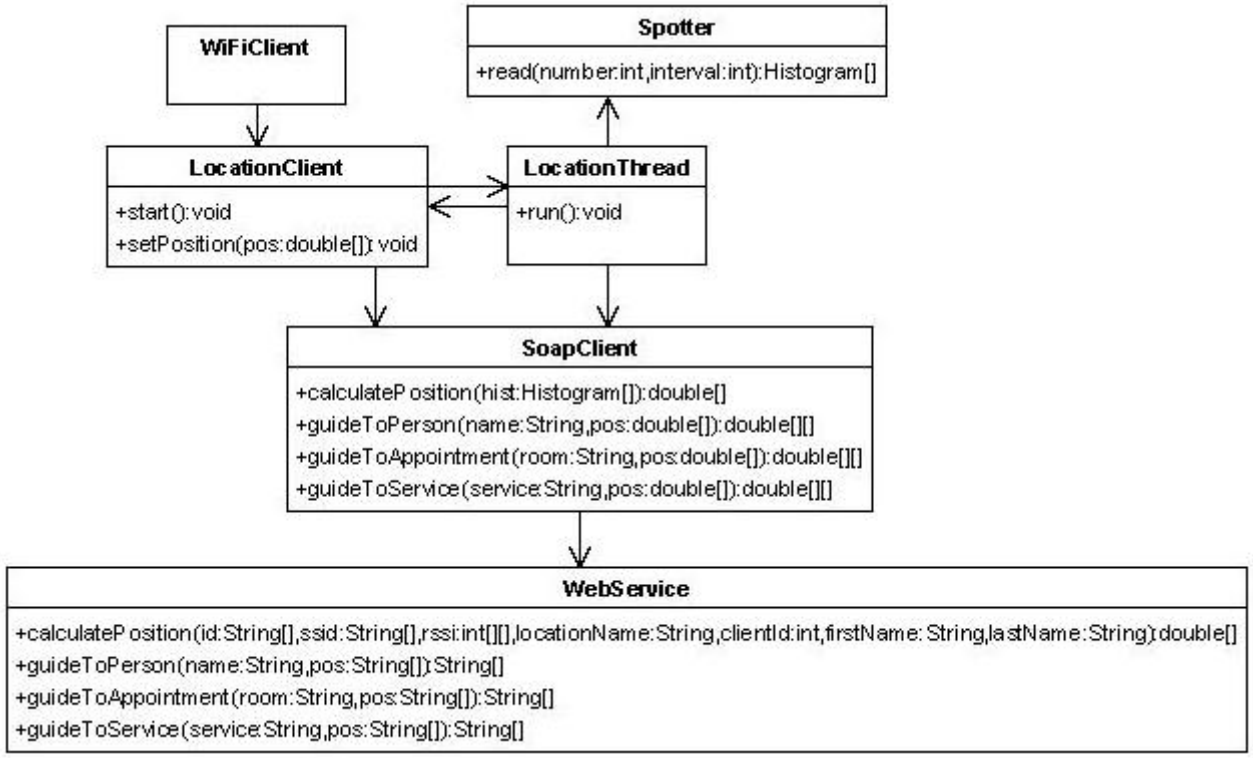
In deze sectie zullen we de manier waarop de locatiebepaling en padberekening geïmplementeerd zijn van naderbij bekijken en bespreken. We zullen dit doen aan de hand van UML-schema's die de samenwerking tussen client en server verduidelijken. We beginnen met de client-kant en daarna zullen we de serverkant van dichterbij bekijken.

5.2.1 Client

De hoofdklasse van de client is WifiClient.java. Deze initialiseert onder andere een startpanel, een menubar, enzovoort. Wanneer nu gekozen wordt om aan locatiebepaling te doen, zal de LocationClient opgestart worden. Deze klasse is een java.awt.Panel en tekent een kaart van het gebouw waarop met een bolletje zal aangeduid worden waar de gebruiker zich bevindt. Eens deze kaart geladen, zal LocationClient een LocationThread opstarten.

LocationThread is logischerwijze een aparte thread die maar één doel heeft: de omgeving meten (door gebruik te maken van de Spotter-klasse), dit in een histogram stoppen en de methode CalculatePosition van SoapClient oproepen. Deze methode geeft een positie terug aan LocationThread, waarna deze die positie zal doorspelen aan LocationClient via de methode setPosition van deze klasse, en de geüpdate positie wordt op het panel getekend.

SoapClient is een object dat alle methoden bevat om communicatie van client naar server op te zetten, door gebruik te maken van het SOAP-protocol. Binnen in die methoden worden alle argumenten omgezet (indien nodig), in de aangemaakte SOAP-enveloppe gestopt en doorgestuurd naar de overeenkomstige methode in WebService aan server-zijde. Eventueel wordt het



Figuur 5.1: Het UML-schema van de client

resultaat dat via de return van de WebService bekomen is, zelf teruggegeven naar het object dat de methode opriep.

Als SOAP-bibliotheek maken we gebruik van kSOAP. Het gebruik van deze bibliotheek zorgt er echter voor dat sommige objecten moeten omgezet worden naar andere objecten vooraleer ze verstuurd kunnen worden. kSOAP codeert bijvoorbeeld tabellen als Vectoren en ondersteunt geen doubles, zodat deze als String moeten gecodeerd worden. Zo zien we dat calculatePosition in SoapClient als argument een tabel van histogrammen krijgt en dit moet omzetten naar twee tabellen van Strings (een met de id's van de access points en een met de ssid's) en een meerdimensionale tabel van meetwaarden, elke rij voor een ander access point.

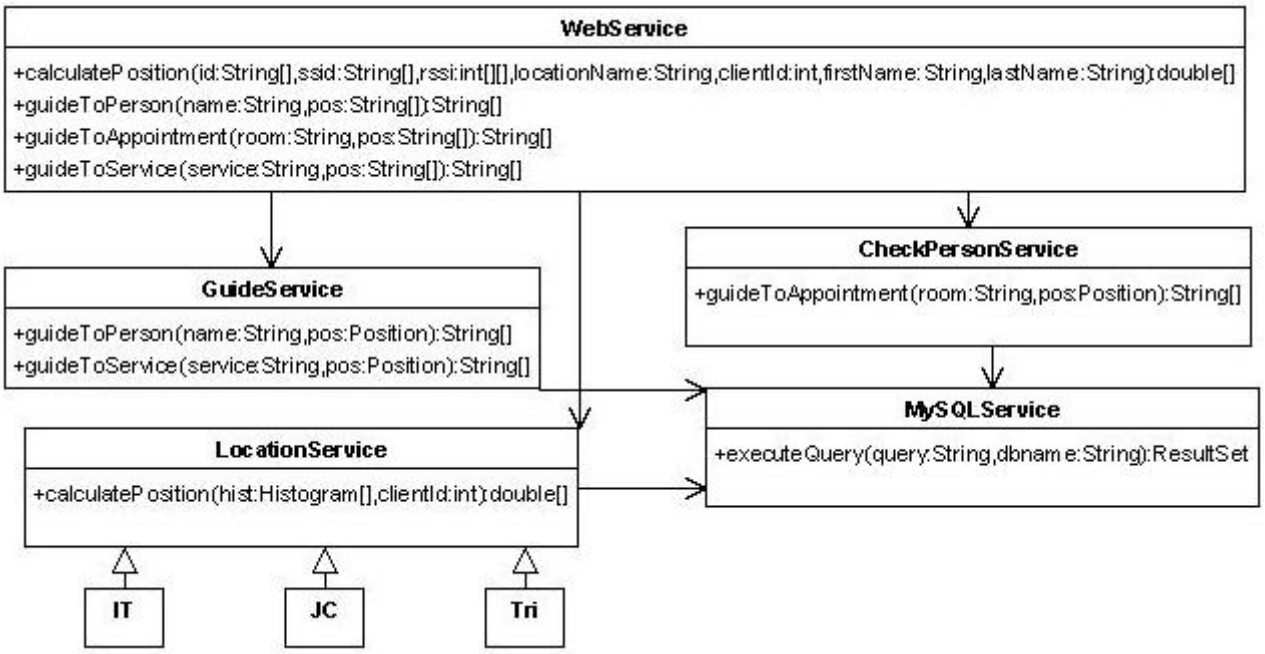
Het meten van access points in de omgeving, nodig om de positie te kunnen berekenen, gebeurt in de klasse Spotter door de methode read(int number, int interval) aan te roepen. Number is hierbij het aantal metingen dat moet gebeuren, en interval het aantal milliseconden tussen elke meting, waardoor de tijd die gespendeerd wordt aan meten number * interval aantal milliseconden in beslag neemt. Spotter, op zijn beurt, maakt gebruik van PlaceLab om deze WiFi-metingen te verrichten. PlaceLab is gratis software om in "urban environments" waar

GPS soms faalt (bijvoorbeeld door hoge gebouwen) toch aan locatiebepaling te kunnen doen door gebruik te maken van publieke access points. Wij gebruiken slechts een klein deel van deze technologie, namelijk het meten van RSSI-waarden. Om meer te weten te komen over PlaceLab zie [33].

Wat de padberekening betreft: LocationThread start bij het begin van de positiebepaling een eigen interne Thread op die om de vijf seconden bij SoapClient de benodigde guideTo-methode oproept, afhankelijk van of er een pad moet berekend worden naar een service (wc, lift, ...), een afspraak of een persoon. Deze zal dan een meerdimensionele tabel van doubles teruggeven met de x-en y-coördinaten van de opeenvolgende punten die doorlopen moeten worden in het pad, waarna LocationClient dit pad op het scherm tekent.

5.2.2 Server

De server runt, zoals eerder vermeld, op het OSGi-platform en bestaat uit een verzameling van services. Elke service is één bundle in OSGi.



Figuur 5.2: Het UML-schema van de server

Alles aan server-kant start vanaf de WebService. Deze klasse bevat alle methoden om de services die de server aanbiedt te kunnen gebruiken. Daaronder bevindt zich de laag met de

geïmplementeerde services zelf.

Wanneer de methode calculatePosition van WebService aangeroepen wordt, gaat deze na welk algoritme er moet gebruikt worden. Elk algoritme is een implementatie van de LocationService-interface waarvan de belangrijkste methode de calculatePosition-methode is die de berekeningen moet doen en de meest waarschijnlijke positie in een Position-object stopt en teruggeeft. Elk algoritme zit in zijn eigen bundle: LocationService-TRI, LocationService-JC en tenslotte LocationService-IT.

Het user-profile is geïmplementeerd als een methode in de klasse Client. Elke keer een nieuwe gebruiker inlogt wordt er een nieuwe instantie van het object Client aangemaakt. Dit object bevat de naam van de ingelogde gebruiker, zijn positie, zijn vroegere positie en nog een paar andere variabelen zoals zijn huidige en gemiddelde snelheid. Verder bevat het de methode getProbability(Position pos) die het eigenlijke user-profile-algoritme uitvoert en berekent hoe groot de kans is dat de bewuste gebruiker zich verplaatst heeft naar positie pos, rekening houdend met alle bijgehouden statistieken.

Om de drie guideTo-methoden uit te voeren, maakt de webservice gebruik van de GuideService en de CheckPersonService. Alle drie de methoden geven een tabel van Strings terug, waarbij elke String bestaat uit een x-coördinaat, gevolgd door een komma en dan een y-coördinaat. Dit zijn de opeenvolgende posities die moeten gevolgd worden in het berekende pad. Zowel guideToService (naar de plaats van de service), als guideToAppointment (naar de kamer van de afspraak) berekenen een statisch pad. GuideToPerson echter heeft de keuze tussen de exacte locatie van de gezochte persoon of zijn bureau, afhankelijk van de instellingen van deze persoon. Het Position-object dat meegegeven wordt, bevat natuurlijk de positie van de persoon die deze pad-berekening aanvraagt.

Tenslotte hebben we nog de MySqlService die alle toegangen tot de databank beheert en die gebruikt wordt door elke andere bundle.

5.3 Authenticatie

Principe

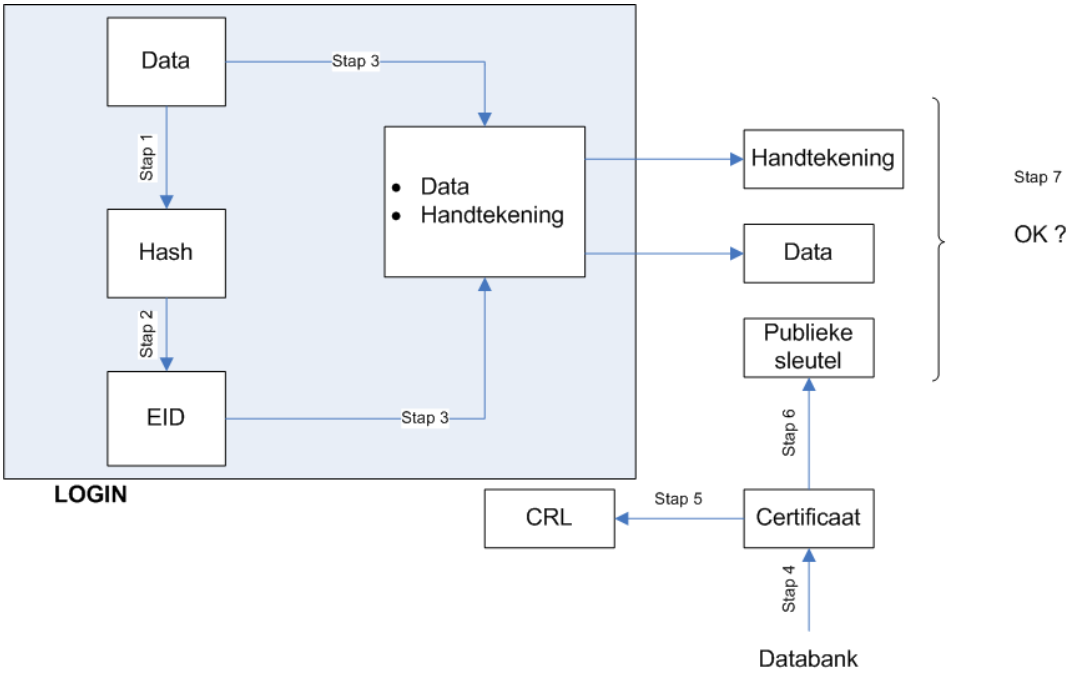
De authenticatie wordt verricht tijdens de inlogfase. Wanneer een bezoeker het gebouw binnenkomt, steekt hij zijn eID in de lezer. Zijn openbare gegevens (naam, voornaam) worden

gelezen. Via de methode MessageDigest wordt hiervan een hash berekend. Deze hash wordt opnieuw naar de kaart gestuurd en er wordt 'on-board' een handtekening gegenereerd met de private authenticatiesleutel (deze sleutel mag de kaart niet verlaten wegens veiligheidsredenen). Wanneer dit gebeurt, moet de bezoeker zijn PIN code opgeven. Zo bevestigt de eigenaar dat hij wel degelijk de rechtmatige eigenaar is. Zoals uitgelegd in het hoofdstuk Smart Cards, kan de eID enkel handtekeningen genereren van een bepaald aantal bits (en niet van volledige strings). Er wordt gebruik gemaakt van SHA-1 als hashalgoritme die een digest genereert van 20 bytes. De handtekening is 128 bytes groot. De data en handtekening worden gebruikt als authenticatie.

De databank met bezoekers ('visitor db') houdt, naast de gegevens van de bezoeker, ook voor elke bezoeker zijn publiek authenticatiecertificaat (X.509v3) bij. Dit is opgeslagen als een BLOB (Binary Large Object). Java biedt ondersteuning voor certificaten in zijn Security package.

Tijdens de authenticatie wordt het 'persoonlijk' certificaat van de bezoeker uit de databank gehaald. De geldigheid wordt gecontroleerd via CRL. De client heeft dus een Certificate Revocation List. Om correct te zijn, zou regelmatig een nieuwe versie moeten afgehaald worden. Na deze controle wordt zijn publieke sleutel geëxtraheerd. Deze RSA sleutel wordt gebruikt om de handtekening te controleren en zo te verifiëren of de persoon is wie hij beweert te zijn. Java biedt ook ondersteuning voor digitale handtekeningen en verificatie ervan. Aangezien de private sleutel op de kaart blijft en de applicatie het publiek certificaat van de bezoeker heeft, kan de authenticatie gegarandeerd worden. Een overzicht wordt gegeven in figuur 5.3

1. Bereken hash
2. Genereer handtekening
3. Het koppel (data,handtekening) wordt gebruikt bij authenticatie
4. Haal certificaat uit databank
5. Controle certificaat via CRL
6. Haal publieke sleutel uit certificaat
7. Controle of de handtekening geldig is = authenticatie



Figuur 5.3: Verloop van de authenticatie

Problemen

- Het zou gemakkelijk zijn indien er een openbare databank bestond waarin alle (publieke) certificaten van de Belgische bevolking in opgeslagen zijn. Maar aangezien dit niet bestaat, werd dit op een andere manier opgelost. Een bezoeker die voor de eerste maal inlogt, stuurt zijn certificaat door en dit wordt opgeslagen in de databank. Aangezien een certificaat ondertekend is door een CA en zijn gegevens ook in het certificaat zijn verwerkt, kunnen we er vanuit gaan dat het certificaat van de bezoeker is.

- Zoals reeds vermeld bij de eID sectie waren er problemen met de eID middleware. Data ophalen werkte probleemloos, maar bij authenticatie werkten sommige methoden niet. Om dit op te lossen werd er overgeschakeld op een open-source versie (zie 3.4). Dit is echter een tussenoplossing aangezien er niet gegarandeerd wordt dat toekomstige versies van de eID zullen werken. Wanneer er nieuwere versies van de eID software uitkomen, is het aangewezen om hierop terug te vallen.

5.4 PDA

Eerst werd er een versie van de client voor PC ontwikkeld, maar nadien is er een backport gemaakt voor de PDA. Deze versie is geïmplementeerd met IBM Websphere Studio Device Developer 5.7.1. De PDA zelf is een HP iPAQ h5450 Pocket PC met Windows Mobile 2003 waarop IBM's J9 VM geïnstalleerd is.

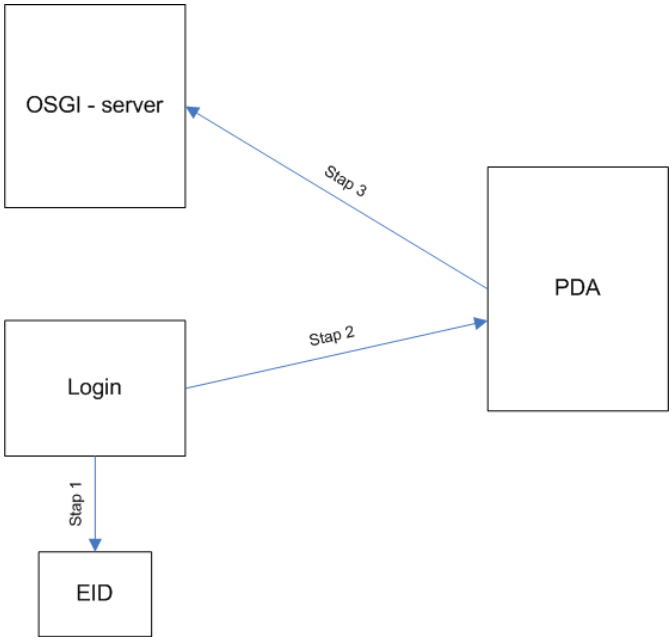
Problemen

- Het grootste probleem was hardware gerelateerd. De ingebouwde WiFi kaart bleek onjuiste metingen te verrichten. Daarom werd er een nieuwe kaart aangekocht, maar deze bleek na heel wat moeilijkheden niet samen te werken met PlaceLab. Uiteindelijk is dan toch gebruik gemaakt van de ingebouwde kaart wat resulteerde in slechte locatiebepaling. De uitgevoerde testen en metingen zijn gebeurd op de laptopclient.

- Een andere tegenvaller was opnieuw de eID middleware. Er bleek namelijk nog geen versie te bestaan die ondersteuning bood voor PDA's. De open-source variant bood evenmin ondersteuning. Er werd dan geopteerd voor de implementatie van een aparte login-module. Deze module kan op een gewone PC uitgevoerd worden. Ze staat in voor het inlezen van gegevens en de aanvraag van de digitale handtekening aan de kaart. Deze gegevens worden dan doorgestuurd naar de PDA. Wanneer een nieuwe bezoeker binnenkomt, gedraagt de PDA zich als server totdat alle authenticatie afgelopen is. Figuur 5.4 toont hoe het inloggen gebeurt.

  1. Login Module leest de gegevens in.
  2. Na het opgeven van het IP adres van de PDA worden de gegevens doorgestuurd. De PDA gedraagt zich als een server zodat deze de gegevens kan ontvangen.
  3. De PDA kan interageren met de applicatieserver voor de authenticatie.

- Aansluitend bij vorig punt is het beveiligen van deze verbinding. Er wordt gebruik gemaakt van sockets om de informatie te versturen maar deze verbinding is niet veilig. Er werd daarom een SSL verbinding geïmplementeerd. Tijdens het testen bleken de sleutels gecreëerd met de standaard Java (de login - module) niet compatibel met J9 VM (de pda). Na wat zoekwerk bleek het probleem gekend en geen oplossing voorhanden.



Figuur 5.4: Inloggen op PDA

- Een ander minpunt van het niet volledig compatibel zijn van de J9 VM, is dat er geen ondersteuning voor Swing is, waardoor sommige elementen werden herschreven naar AWT (Abstract Windowing Tookit).

- We hebben ter controle van de geldigheid van de certificaten enkel CRL geïmplementeerd. Een probleem hierbij is dat de CRL file voor heel wat vertraging zorgt tijdens de authenticatiefase (het is een bestand van 4 MB groot). Dit zou opgelost kunnen worden door OCSP te implementeren maar dit wordt maar ondersteund in Java 1.5, nog niet in J9. Indien dit in een latere versie van J9 komt, kan dit probleem opgelost worden.

Hoofdstuk 6

Metingen

In dit hoofdstuk bespreken we de metingen die uitgevoerd zijn om de verschillende localisatie-algoritmen te testen en welke resultaten we hiermee bekomen hebben.

Er zijn testen gedaan naar verschillende foutindicatoren: in het eerste deel van dit hoofdstuk gaan we bekijken op hoeveel afstand nauwkeurig de algoritmen kunnen localiseren en wat hun maximum en minimum fout is, in het tweede deel bekijken we in hoeveel procent van de metingen ze de juiste kamer vinden, zonder ons daarbij om de foutafstand te bekommeren.

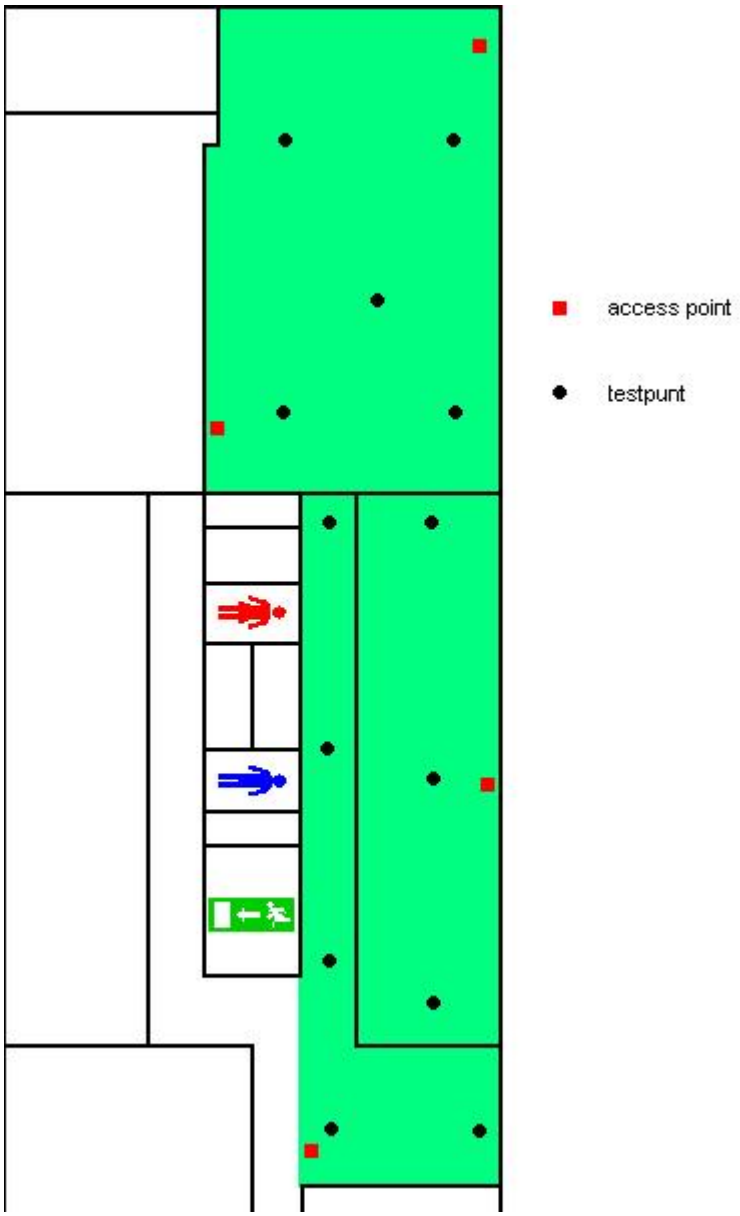
De metingen werden verricht door te klikken op de map op de locatie waar men zich werkelijk bevond, en dan een aantal localisatiebepalingen te doen en te kijken hoe deze zich verhouden ten opzichte van de de werkelijke positie.

6.1 Metingen naar algemene foutafstand

6.1.1 Opstelling

Het gebied waarin de testen gebeurden was zo'n 330 m^2 groot. Een rechthoek van 10 op 17 m naast een rechthoek van 7 op 24 m (zie figuur 6.1). We maakten gebruik van vier access points, die verspreid over het gebied opgesteld stonden.

De referentiepunten werden op anderhalve meter van elkaar gelegd, wat resulteerde in een totaal van honderdtweeënvijftig referentiepunten. Het groen ingekleurde gebied op figuur 6.1 is het ingescande gebied. Op elk punt werd de omgeving gedurende één minuut ingescand.



Figuur 6.1: Opstelling voor metingen naar foutafstand

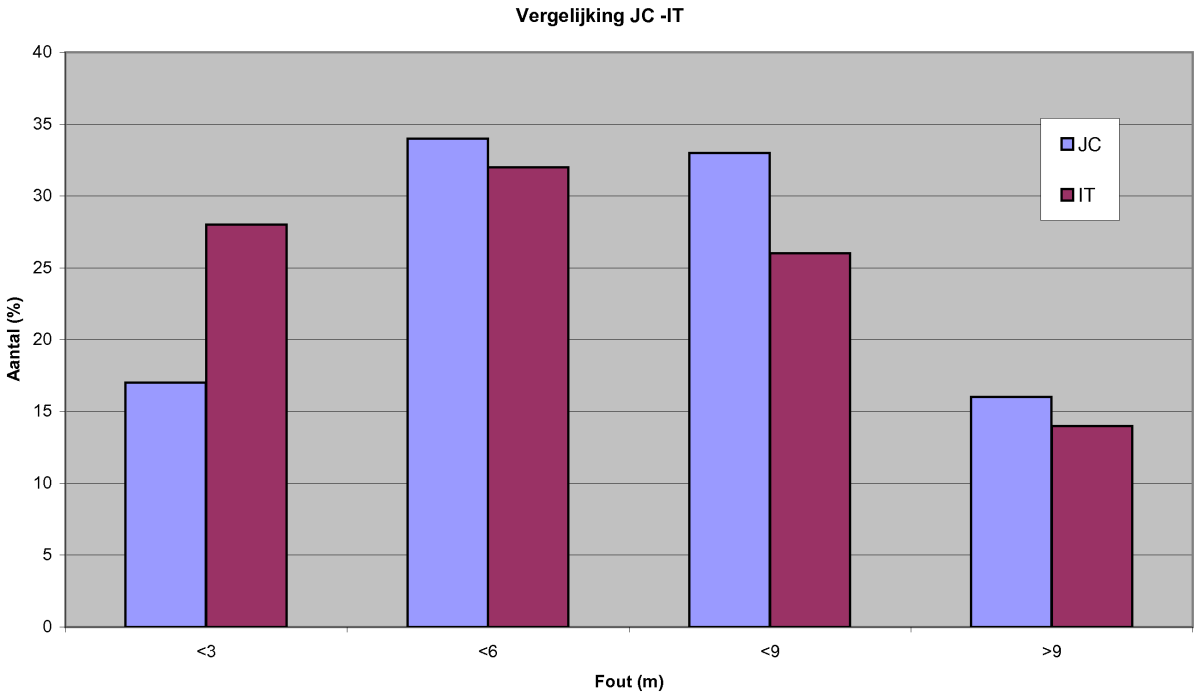
De testfase zelf gebeurde op dertien verspreide testpunten. Op elk testpunt gebeurden vijftig localisatiebepalingen, wat ongeveer neerkomt op één minuut en dertig seconden scantijd per punt.

6.1.2 Radio-map

Zonder user-profile

In figuur 6.2 staan de resultaten van de metingen met het Joint-Clustering- en het Incremental Triangulation-algoritme zonder het gebruik van het user-profile.

Wat meteen opvalt is dat beide algoritmen ongeveer even goed zijn. Het IT-algoritme heeft wel meer metingen die kort bij de werkelijke locatie komen. Dit komt vooral door het feit dat JC enkel in bepaalde clusters gaat zoeken. Vergist hij zich echter van cluster dan zit hij er onmiddelijk een aantal meter naast. IT heeft dit nadeel niet.



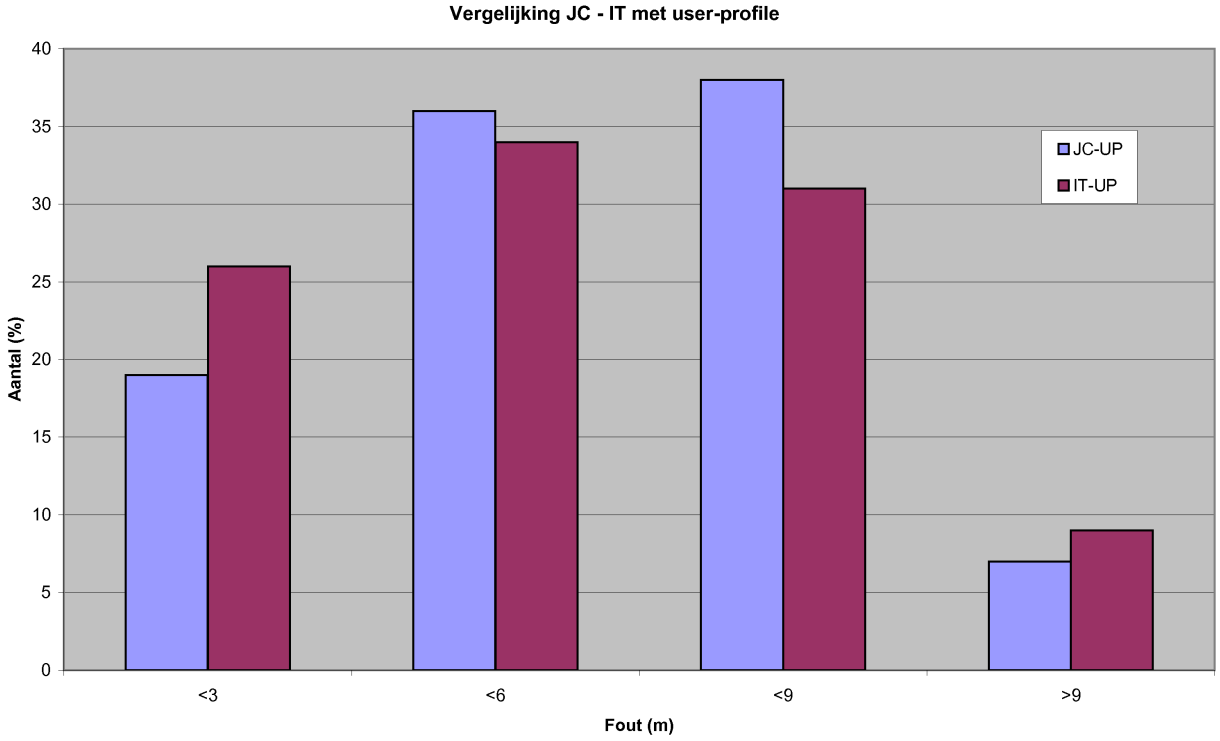
Figuur 6.2: Metingen radio map zonder user-profile

Als we de figuur samenvatten, dan zien we dat beide algoritmen in iets meer dan 50% van de

gevallen de juiste locatie tot op zes meter benaderen. In 13% (IT) en 16% (JC) van de gevallen was de fout groter dan negen meter. Die grote fouten waren vooral alleenstaande gevallen door een hele slechte en afwijkende meting.

Met user-profile

Ook bij deze metingen (zie figuur 6.3) valt onmiddellijk op dat beide algoritmen vrij gelijklopend zijn in resultaten. Wat ook opvalt is dat de slechte metingen, in tegenstelling tot het gedeelte zonder user-profile, meestal in serie kwamen: was de eerste locatiebepaling slecht, dan waren dikwijls de direct daaropvolgende locatiebepalingen ook slecht. De reden hiervoor is natuurlijk dat het user-profile ervoor zorgt dat er enkel een locatie gezocht wordt die op een realistische afstand van de vorige positie gelegen is. Het nadeel hiervan is dus ook dat bij een heel slechte eerste localisering het een hele tijd duurt eer de fout weer binnen de perken is. Het feit dat er niet door muren kan gesprongen worden draagt hier ook toe bij, want het kan zijn dat de goeie kamer dan nooit meer gevonden wordt.



Figuur 6.3: Metingen radio map met user-profile

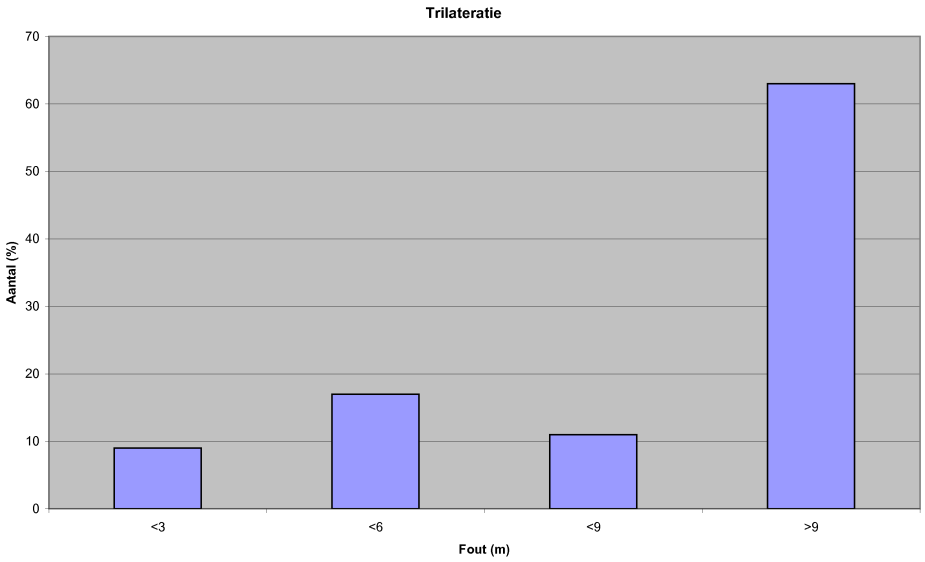
Gaan we nu de resultaten uit figuur 6.2 vergelijken met figuur 6.3 dan zien we dat de algoritmen met user-profile iets beter presteren dan de algoritmen zonder user-profile maar het verschil is niet echt spectaculair te noemen. Bekijken we nu echter enkel de afstand >9 m dan zien we wel dat het user-profile voor minder grote fouten zorgt. Ook hier weer is de oorzaak dat er niet zomaar van de ene kant van de map naar de andere kan gesprongen worden. Wordt er dan toch al eens een hele slechte meting gedaan, dan zorgt het user-profile ervoor dat de fout binnen de perken blijft.

$Algoritme$	$Minimale\ Fout(m)$	$Maximale\ Fout(m)$
IT	0.2	13.4
$IT - UP$	0.4	15.6
JC	0.5	16.7
$JC - UP$	0.4	12.9

Tabel 6.1: Maximale en minimale fout bij de radiomap-algoritmen

6.1.3 Trilateration

De gedane metingen voor het trilateratie-algoritme (zonder user-profile) zijn uitgezet in figuur 6.4.



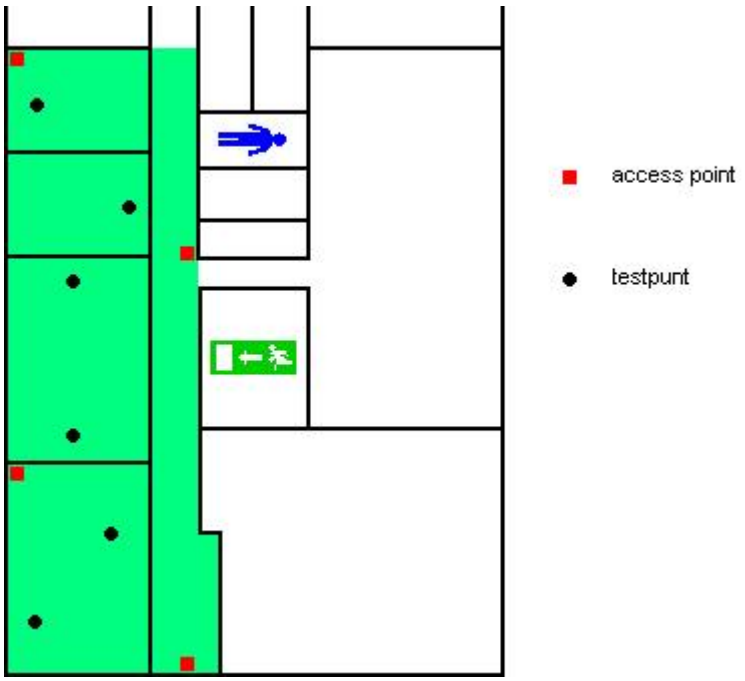
Figuur 6.4: Metingen trilateratie

Als we deze figuur analyseren kunnen we vrij snel besluiten dat dit algoritme niet aan de eisen en verwachtingen voldoet. Meer dan 60% van de locatiebepalingen had een fout van meer dan negen meter, en de locatiebepalingen bleken ook heel onvoorspelbaar. Zoals ook al in [34] te lezen valt, blijkt trilateratie niet de geschikte technologie om in een binnen-omgeving toe te passen. Binnenshuis zijn er te veel factoren zoals andere apparaten, muren, meubelen enzovoort die het heel moeilijk maken om op een correcte manier de positie in te schatten.

6.2 Metingen naar juistheid van de kamer

6.2.1 Opstelling

Deze test gebeurde in een gebied van ongeveer 150 m^2 groot. We maakten gebruik van dezelde vier access points als in de eerste metingen en verspreidden ze over het testgebied (zie figuur 6.5).



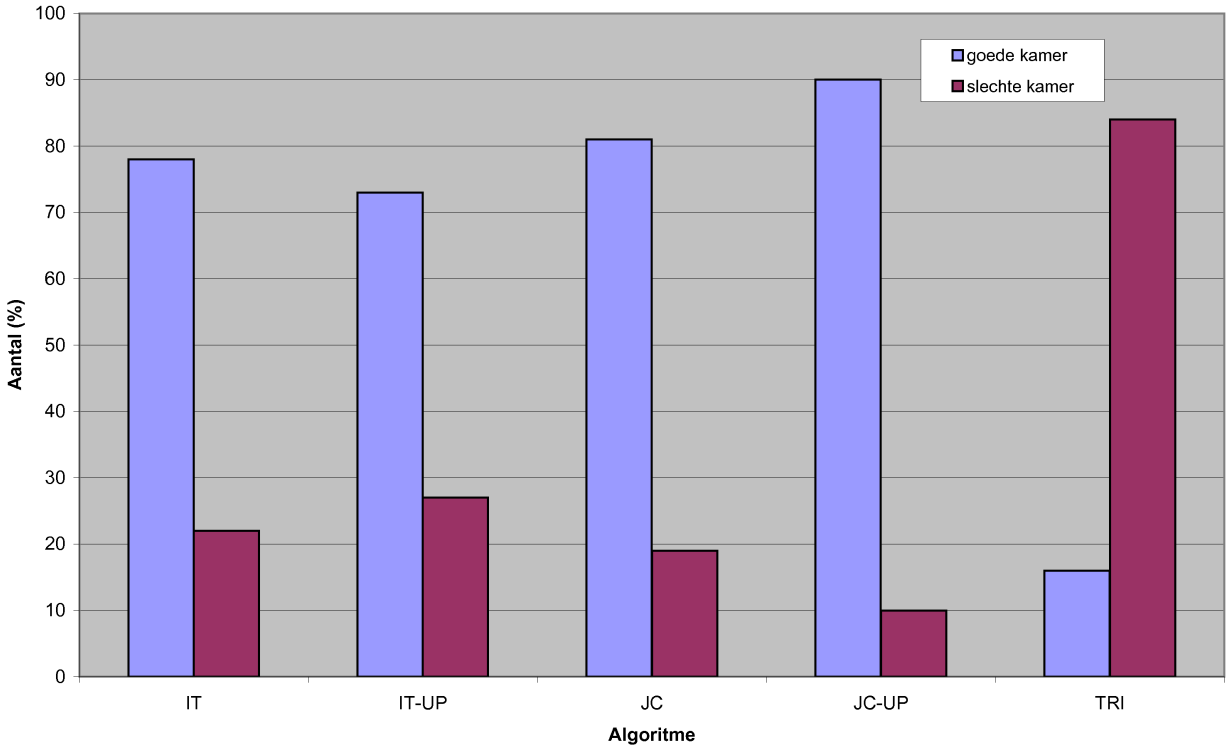
Figuur 6.5: Opstelling voor metingen naar juistheid van de kamer

De referentiepunten werden weer standaard op anderhalve meter van elkaar gelegd, wat zorgde voor een totaal van zesenvijftig referentiepunten. Op elk punt werd de omgeving weer gedurende één minuut ingescand.

Voor het testwerk kozen we zes willekeurige testpunten uit. We deden vijftig localisatiebepalingen per punt, wat neerkomt op een localisatietijd van ongeveer anderhalve minuut. Zowel de testen als het scannen van de referentiepunten gebeurden 's avonds wanneer er weinig personen in de buurt waren, zodat er niet teveel interferentie was.

6.2.2 Resultaten

De resultaten van de metingen staan weergegeven in figuur 6.6.



Figuur 6.6: Metingen juistheid kamers

We merken meteen op dat we vrij goede resultaten bekomen hebben. Alle radio-map algoritmen leverden een juistheid op van vijfenzeventig tot zelfs negentig procent. In de meeste van de slechte resultaten, werden we gelocaliseerd in de kamer naast de kamer waar we ons werkelijk bevonden, of stonden we vrij dicht tegen een deur waardoor het al eens gebeurt dat het localisatie-algoritme ons om de hoek localiseerde wat aanzien werd als een verkeerde kamer omdat zich een muur bevond tussen ons en het bekomen resultaat.

De resultaten die we bekwamen met de user-profile algoritmen kwamen meestal weer in serie. Hiermee wordt bedoeld dat, als de eerste localisering fout was, meestal minstens de helft van de resterende negenenveertig localiseringen ook fout waren. Dit komt natuurlijk opnieuw doordat het algoritme deze tijd nodig heeft om zich te verplaatsen naar de werkelijke positie waar we ons bevonden. Was de eerste localisering goed, dan was er veel kans dat geen enkele van de vijftig localiseringen een fout maakte. Bij de algoritmen zonder user-profile merkten we dit fenomeen natuurlijk niet op, deze konden zich vrij snel herstellen van een fout, maar konden daarentegen natuurlijk ook een aantal goeie localiseringen doen gevolgd door een paar mindere door een aantal slechte WiFi-metingen.

Het enige algoritme dat hier weer compleet uit de toon valt, is het trilateratie-algoritme: slechts in zestien procent van de gevallen slaagde dit algoritme erin om ons in de goeie kamer te localiseren, en meestal was het totaal niet te voorspellen waar we zouden terechtkomen. Dit was echter wel te verwachten, gezien de metingen uit het eerste deel van dit hoofdstuk.

Hoofdstuk 7

Future Work

7.1 Authenticatie van personeelsleden via eID

De personeelsleden loggen nu in op de Appointment Scheduler via een wachtwoord. Men zou een analoge beveiliging kunnen opzetten zoals bij het inloggen van bezoekers waardoor de personeelsleden via hun eID geauthenticeerd worden. In de eID Middleware documentatie zit een document waarbij een voorbeeld wordt uitgewerkt voor een Apache server. We kunnen dit niet rechtstreeks implementeren in de webservice die we gebruiken in OSGi.

7.2 Appointment Scheduler beveiligen met HTTPS

We zouden (net zoals we bij de PDA geprobeerd hebben) de verbinding met de Appointment Scheduler kunnen beveiligen met SSL. Dit kan nuttig zijn bij zowel het inloggen met een wachtwoord als het inloggen via de eID. Normaal gezien ondersteunt de webserver HTTP over SSL (HTTPS) en biedt zo een dergelijke service aan (sleutels, ...). De server moet dan enkel nog juist geconfigureerd worden. Helaas zit er in OSGi nog geen HTTPSService. Dit zou later nog kunnen geschreven worden.

7.3 eID op PDA

Indien IBM in een nieuwe versie van de J9 compiler de gegeneerde sleutels van haar 'keytool' (programma dat publieke/private sleutels genereert) compatibel zou maken met die van de

standaard JRE, zouden we de verbinding met de PDA kunnen beveiligen met SSL.

Een nog betere oplossing zou zijn indien er eID Middleware zou uitkomen geschikt voor Pocket-PC's. Voorlopig zijn hier nog geen concrete plannen voor.

7.4 OCSP

De controle van de geldigheid van de certificaten gebeurt nu met CRL. Indien, zoals reeds vermeld in het hoofdstuk Implementatie, J9 OCSP in een latere versie zou ondersteunen, dan zou authenticatie sneller en accurater kunnen gebeuren (de 'online' lijsten zijn altijd up-to-date).

7.5 Veilige PIN invoer

Om het geheel nog veiliger te maken, zouden we in principe geen invoer van de PIN mogen accepteren via het toetsenbord. Deze invoer kan door een andere pc applicatie onderschept worden. Er bestaan speciale toestellen die toelaten om op een veilige manier een PIN in te geven en dan nog kan er gefraudeerd worden. Daarnaast weet de kaarthouder niet welke gegevens en commando's naar de kaart gestuurd worden (WYSIWYS: What You See Is What You Sign) maar dit zou ons wat te ver leiden.

7.6 Verbeteren user-profile

Het door ons ontwikkelde user-profile is nog voor verbetering vatbaar: zo komt het voor dat de ingang naar een bepaalde kamer niet gevonden wordt en de locatiebepaling tegen de muur blijft opbotsen. Een oplossing hiervoor zou bijvoorbeeld kunnen zijn om, indien de locatie met de hoogste mogelijke probabiliteit onder een bepaalde drempel-probabiliteit zit het user-profile te resetten zodat het terug bij de echte locatie kan geraken. Andere verbeteringen zouden kunnen zijn om rekening te houden met nog meer factoren zoals bijvoorbeeld de nabijheid van andere mensen (meer kans om te stoppen en een praatje te maken).

7.7 Locatiebepaling met PDA

Het lezen van de referentiepunten en het localiseren door gebruik te maken van een laptop is vrij onhandig wegens de grootte en het gewicht van dit toestel. Oorspronkelijk was het de bedoeling dit met een PDA-toestel te doen, maar door de incompatibiliteit van het netwerkkaartje van deze PDA met PlaceLab was dit echter onmogelijk. In de toekomst is het natuurlijk de bedoeling een kaartje te vinden waarmee dit wel mogelijk is.

Hoofdstuk 8

Conclusie

Het ontwikkelen van een goed localisatiesysteem aan de hand van WiFi in een indooromgeving bleek geen makkelijke opgave. De aanwezigheid van storende straling van andere elektronische apparaten, de vele obstakels zoals muren, meubels en mensen, en andere externe factoren zorgen vaak voor een verstoord signaal.

In het geval van het trilateratiegebaseerd algoritme bleek dat het algoritme hier niet tegen opgewassen was en onmogelijk zijn omgeving kon inschatten waardoor de localisering onvoorspelbaar was en vaak de mist inging.

De radiomapalgoritmen daarentegen bewezen wel degelijk hun waarde. Door het gebruik van twee fasen werd het wisselvallige aan het WiFi-signaal voor een groot deel opgevangen. Het scannen van de referentiepunten voor een periode die lang genoeg is, zorgt voor een betere inschatbaarheid van het signaal. Samengevat zorgen deze soort algoritmen voor een localisering die gemiddeld correct is tot op vijf meter in een grote ruimte, en in gemiddeld tachtig procent van de gevallen de goeie kamer kiest.

Het user-profile betekende niet echt een spectacularie verbetering in de localiseringsresulaten, maar zorgde toch voor twee grote voordelen. Ten eerste werd het aantal grote fouten (fouten groter dan negen meter) gevoelig beperkt. Ten tweede zorgde het voor een beter performantie door een groot deel van de mogelijke posities onmiddellijk als 'onmogelijk' te bestempelen waardoor de berekeningen voor deze posities onnodig worden en niet meer dienen uitgevoerd te worden. Samenvattend kunnen we stellen dat het user-profile zijn voordelen bezit maar nog niet als 'af' kan bestempeld worden.

De huidige smart card technologie gaat snel vooruit. De mogelijkheden worden ook alsmaar

uitgebreider. Met de komst van multi-applicatie kaarten en platformen die dit ondersteunen, wordt het mogelijk om meerdere functionaliteiten op één kaart te bundelen. De eID heeft ons het voordeel gegeven dat elke inwoner van België in de toekomst een dergelijke kaart zal hebben (wat een aanzienlijke kostenbesparing is). Daarbij komt dat we zelf geen sleutels of certificaten hoeven te maken om op een smart card te 'pushen'. De PKI op de eID is goed aangepakt en maakt gebruik van de laatste standaarden (X.509v3, OCSP, ...). Zo kan elke burger zich authenticeren, digitale handtekeningen plaatsen, enz. Dit biedt heel wat voordelen voor bv. e-commerce. Verder zal de eID niet alleen voor authenticatie en identificatie gebruikt worden maar kan de kaart uitgebreid worden met medische gegevens (SIS), studentengegevens, ... Als de overheid zorgt voor verbeterde software dan ligt er een mooie toekomst voor de elektronische identiteitskaart in het vooruitzicht.

Java biedt goede ondersteuning voor PKIs en is dan ook een zeer geschikte taal om hiermee te werken.

Bijlage A

Smart card Technologie Scan

A.1 Gemplus

A.1.1 Algemeen

Gemplus is één van de belangrijkste leveranciers voor smart cards. De firma heeft een uitgebreid gamma. Hier wordt één van de 'topmodellen' besproken nl. de GemXpresso Pro R3.2. Deze versie bestaat in 2 varianten, een variant met 18 Kbytes EEPROM en de uitgebreide versie met 32 Kbytes EEPROM. De GemXpresso wordt geleverd met het GemSAFE applet. GemSAFE is een Java applet dat alle nodige functies biedt om een smart card in een Public Key Infrastructure(PKI) te integreren. De applet is geschikt voor de beveiliging van zowel identiteits- als bedrijfsapplicaties. Ze is ook nuttig om informatie over de kaarthouder en gevoelige data op te slaan. GemSAFE implementeert een state-of-the-art security en voldoet aan de laatste standaarden voor smart cards en PKI applicaties. De applet is ingebakken in het ROM zodat de EEPROM vrij is om gegevens van de applicatie op te slaan.

A.1.2 Specificaties

GemXpresso Pro 3.2 E32 PK	
Chip	Infineon SLE66CX322P chip
Geheugen	EEPROM: 32 Kbytes
Standaard	- Java Card Virtual Machine, RTE en API volgens JC2.1.1
	- CardManagement & API volgens OP2.0.1'
Encryptie	- 3DES (ECB,CBC)
	- RSA tot 2048 bit
	- SHA-1 (Digitale handtekening)
	- On - card asymmetrische sleutelpaar generatie
	- op publieke-sleutelgebaseerde DAP
	(=Data Authentication Pattern)
Beveiliging	Verschillende hard- en software maatregelen tegen:
	Side Channel aanvallen, Invasive aanvallen, ...
Ondersteunde Protocollen	- T=0;T=1; PPS
	- baud rates tot 115 kbps
Features	- verwijdering van applet
	- ondersteuning voor gefragmenteerd geheugen beheer
	- garbage collection
	- Java Card code en adresseerruimte tot 128 Kbytes
Applets	MPCOS applet :
	- data opslag met op 3 DES gebaseerde beveiliging
	- ISO7816-4
	GemSafe applet:
	- PKI applet voorzien van digitale handtekening en verificatie
	- kaart/terminal authenticatie geschikt voor E-Sign
	- ondersteunt PKCS#15
	- ondersteunt Identrus vereisten
	- ondersteunt ISO 7816-4-6-8-9-15

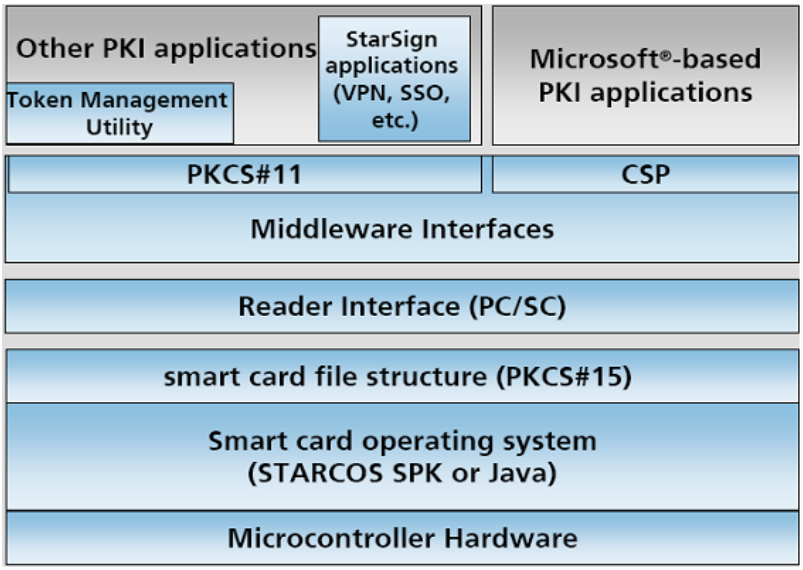
A.1.3 Prijzen

- Voor bestellingen kleiner dan 5 stuks = 47 euro per kaart met GemSafe libraries. Deze prijs is excl. BTW (21%).
- Voor bestellingen groter dan 5 stuks = 46 euro per kaart met GemSafe libraries. Deze prijs is excl. BTW (21%).
- Voor bestellingen groter dan 50 stuks = 34 euro per kaart met GemSafe libraries. Deze prijs is excl. BTW (21%).

A.2 Giesecke & Devrient

A.2.1 Algemeen

Giesecke & Devrient biedt een gamma smart cards aan onder de naam StarSign. Deze bestaat uit 2 soorten. De eerste soort, de StarCos serie, heeft een eigen besturingssysteem. De tweede soort, de Smart Cafe Expert serie, zijn Java Cards. Een overzicht van de StarSign architectuur wordt gegeven in Figuur A.1. De StarSign is beschikbaar in verschillende formaten, bv. als een smart card of USB token.



Figuur A.1: Architectuur van de StarSign familie

A.2.2 Specificaties

StarCos

	SPK 2.3	SPK 2.4	SPK 2.5 DI Type A,B,C
Chip	Philips	Philips	Infineon SLE66CLX320P
Geheugen	32KByte	32KByte	32KByte
Standaarden	ISO 7816-3/4/8	ISO 7816-3/4/8	ISO 7816-3/4/8
			- Digitale handtekening (CWA 14169)
Encryptie	- DES, 3DES	- DES, 3DES	- DES, 3DES
	- DSA, RSA	- RSA	- RSA
Protocol	T=0;T=1	T=0;T=1	
Features	- verscheidene toegangscontroles	Zoals 2.3 +	
	- RSA-encryptie (1024 bit)	- 4 logische kanalen	
	- on-card RSA sleutel generatie	- versie 'FIPS'	
	- digitale handtekening(RSA,DSA)	- versie met biometrie	
	- ITSEC E4 gecertificeerd		
Applets	- StarSign SafeSign	- StarSign SafeSign	- StarSign SafeSign
	- DIN Sig		
Opm			- contactloos protocol (ISO 14443)
			- ondersteunt Felica specificatie

Java Card

	Expert 2.0	Expert 64
Chip	Infineon SLE66CX322P	Renesas AE 46C1
	- Common Criteria EAL5+	- Common Criteria EAL4+
		- FIPS 140-2 Level 3
Geheugen	32 KByte EEPROM	64 Kbytes EEPROM
Standaard	- GlobalPlatform 2.0.1'	- GlobalPlatform 2.0.1'
	- Java Card 2.1.1	- Java Card 2.2
	- ISO 7816 -3/4	- ISO 7816 1-5
		- NIST FIPSPUB 140-2
Encryptie	- DES, 3DES	- DES, 3DES, AES
	- RSA (1024 bit)	- RSA(2048 bit), DSA(1024 bit)
	Met cryptoprocessor: hash:	- RSA,DSA on-card generatie
	- SHA-1,MD-5,RIPEMD	- hash: zoals 2.0
		- Digitale handtekening
		- DAP verificatie
Beveiliging		- beveiligd tegen SPA,DPA en DFA
		- firewall veilig voor DFA
		- versleutelde opslag van data
Protocol	- T=0;T=1	- T=0;T=1;PTS/PPS
	- baud rates tot 115kbps	- baud rates tot 115kbps
Applets	- EMV,PKI, OTP,...	- StarSign SafeSign
		- Identification, PKI,...
Features	- ActivCard Gold ondersteuning	- ActivCard Gold ondersteuning
		- Star Sign Token voor Java
		- biometrische versie beschikbaar

A.2.3 Prijzen

Er zijn geen prijzen vermeld. Men heeft wel het volgende laten weten:

- minimum bestelling van 100 stuks
- verplicht zelfde aantal SafeSign licenties (1 token = 1 licentie)

Er is wel een 'Evaluation Kit' beschikbaar op aanvraag voor test-doeleinden.

A.3 Axalto

Het betreft hier de Cyberflex Crystal.

A.3.1 Specificaties

	Cyberflex Crystal
Chip	Common Criteria EAL4+
Geheugen	32 Kbytes EEPROM
Standaard	- ISO 7816
	- Java Card 2.1.1
	- GlobalPlatform 2.0.1
	- CWA 14169 (Digitale handtekening)
Encryptie	-DES, 3DES
	- RSA
	- SHA-1
	- on-card generatie DES en RSA
Protocol	T=0
Opm	MiFare(contactloos) en HID beschikbaar

A.3.2 Prijzen

Niet beschikbaar.

A.4 SafeNet - DataKey

A.4.1 Algemeen

De firma Datakey werd recent overgenomen door SafeNet en de overgang is nog volop bezig. Dus het model is voorlopig nog DataKey 'gemerkt'. Het Model 330j is SafeNet's Java - gebaseerde smart card, ontworpen voor Java Card v2.1.1 en Global Platform v2.0.1 specificaties. Het Model 330j bevat SafeNet's JCCOS (Java-based Cryptographic Card Operating System) OS applet. Deze applet is een applicatie met efficiënt cryptografisch- en databeheer, ingebouwd in het ROM. Met deze ingebouwde applet bespaart het Model 330j het meeste van zijn 32K EEPROM voor andere applicaties en data opslag (in de toekomst kunnen bedrijven dan eigen applicaties toevoegen). Om Global Platform v2.0.1 te ondersteunen (voor het laden en verwijderen van de applets) laat de architectuur toe om digitale 'credentials' te beheren door gebruikers de mogelijkheid te bieden om de inhoud van hun eigen kaart te wijzigen (in de mate dat dit toegelaten is door de veiligheidspolicy van de organisatie). SafeNet's Model 330j komt ook tegemoet aan de GSC-IS native card-edge interface vereisten. Deze biedt veel voordelen vanuit het standpunt van beveiliging, gebruik van geheugen, deployment en kaartbeheer.

A.4.2 Specificaties

	330J
Geheugen	- 32 Kbytes OS in Rom
	- 32 Kbytes EEPROM
Standaarden	- ISO 7816 2-4
	- FIPS PUB 186 (Digitale Handtekening Standaard)
	- PKCS# 1/3/11
Encryptie	- cryptografische coprocessor
	- on-card DES encryptie
	- RSA/DSA/ECC sleutel generatie
	- RSA/DSA/ECDSA voor digitale handtekening
Beveiliging	Hard - en software bescherming tegen power- en timingaanvallen
Protocol	T=1
Features	DataKey CIP ondesteunt PC/SC

A.4.3 Prijzen

Niet beschikbaar.

A.5 Axalto-eID

A.5.1 Algemeen

EID is speciaal ontworpen voor de overheid door Axalto (Schlumberger) om als eID kaart te fungeren. De BelPIC Applicatie v 2.0 is de applet die zorgt voor alle ondersteuning en bestaat uit 2 applicaties:

- elektronische handtekening applicatie
- elektronische identificatie applicatie

A.5.2 Specificaties

	eID
Chip	-16 bit controller
	-crypto processor (RSA en DES)
Geheugen	32 Kbytes EEPROM (Cristal Applet)
Standaard	-ISO 7816 1-9
	-ISO CEI 9564-1
	-ISO/IEC 9798-3
	-PKCS# 15
	- Java Card 2.1.1
	- Global Platform
Encryptie	- digitale handtekening(Cryptoki, PKCS#11)
	- PKCS#1 2.1
	- publiek: RSA (2048 bit)
	- privaat: RSA (1024)
Protocol	T=0
Applet	BelPIC

A.5.3 Prijzen

De eId starter kit Light omvat 1 eId testkaart, 3 downloadbare sleutelparen (handtekening en authenticatie) van certificaten en toegang tot de validatiediensten voor certificaten. Dit alles is het 'goedkoopste product' van de eId shop nl. 85 euro per kit. De kaarten zelf zijn compatibel met elke PC/SC reader.

A.6 Conclusie - Vergelijking

A.6.1 Vergelijking

Model	R3.2	StarCos	SmartCafe	Cyb.Crist.	330J	eID
Geheugen	*	*	+	*	+	+
Protocol/Standaarden	+	-	+	-	*	*
Encryptie	+	*	+	*	+	+
App/Features	+	*	+	*	+	*
Software	+	-	+	*	+	+

- N: niet geweten: stond niet in specificatie, ...
- +: goed
- *: redelijk
- -: minder goed

Bij versies met een afgeslankte versies (minder geheugen, minder software, ...) werd altijd de 'duurste' versie genomen. De prijs werd niet in de tabel opgenomen omdat deze niet altijd bekend was (wegens: geen reactie meer, enkel beschikbaar in buitenland, ...).

A.6.2 Conclusie

Om te beginnen zou ik willen stellen dat het niet eenvoudig is om losweg een oordeel over bepaalde producten te vellen. Het is immers onmogelijk ze allemaal aan te kopen om te testen. Maar door een vergelijking van de specificaties en gegevens van de fabrikanten zelf kunnen we toch een redelijk zicht krijgen op de kaarten.

De R3.2 heeft een applet (GemSAFE applet) dat voorziet in alle functionaliteiten zoals beveiliging, bestandsbeheer en certificaten. Er zijn nog kaarten die soortgelijke applets aanbieden maar die van Gemplus lijkt er toch bovenuit te steken. Qua uitgebreidheid scoort deze kaart dus zeer goed. Ook het feit dat ze een verdeler in België hebben is een voordeel.

De firma Gisecke & Devrient biedt eveneens een heel gamma aan Smart Cards (de StarSign familie). De dienstverlening naar de klant toe is zeer goed. Het 'lichtere' broertje (de StarCos serie) wordt ondersteund door hun eigen OS (nl. StarCos Operating System), de duurdere versie (de Sm@rtCafe reeks) zijn Java Cards. Deze laatste zijn dus te prefereren omwille van hun compatibiliteit met Java. De Sm@rt Cafe Expert 64 valt hierbij het meest in het oog aangezien het de meeste mogelijkheden biedt qua geheugen en beveiliging.

Axalto profilteert de Cyberflex Cristal als een Java Card met gecertificieerd applet, OS en chip. Het zou foutloze security moeten bieden die voldoet aan Commmon Criteria normen. Over het algemeen lijkt deze kaart toch minder functies te hebben dan de andere kaarten (bijvoorbeeld enkel ondersteuning voor T=0 protocol).

SafeNet-DataKey heeft zowel een Java versie als een niet Java versie van zijn model 330 en maakt vooral reclame voor zijn User - en Token/Host Authentication. De Java versie heeft ook zijn eigen OS applet (nl. JCCOS) - een 'hoogst efficiente cryptografische en data management applicatie' - dat ingebouwd is in zijn ROM (dus het EEPROM is vrij voor andere applicaties of data). Het claimt ook gemakkelijk uitbreidbaar te zijn voor toekomstige requirements.

De eID kaart is speciaal ontwikkeld voor de Belgische overheid waardoor ook niet alle opties zoals op andere kaarten te verwachten zijn. Dit is ook niet noodzakelijk aangezien het enkel aan de requirements moet voldoen. Zo ondersteunt de kaart geen verwijdering van applicaties of bestanden, enkel T=0 protocol, ... Al bij al heeft deze kaart genoeg aan boord om te voldoen aan de eisen van een elektronische identiteitskaart, ook met het oog op mogelijke toekomstige toepassingen (denken we maar aan: e-commerce, eID als studentenkaart, implementatie van SIS, ...). De eID wordt uitvoerig besproken in het hoofdstuk 'Smart Cards'.

Van de algemene kaarten komen de Sm@rt Cafe Expert 64 en de GemXpresso R3.2 als beste uit 'de test'. Het grote verschil tussen de twee is de grootte van het EEPROM (resp. 64 Kbytes en 32 Kbytes). Helaas was er van deze eerste geen prijs beschikbaar, wat een beslissende factor had kunnen zijn.

Bijlage B

CD-Rom

Op de bijgevoegde CD-Rom bevindt zich alle gebruikte software:

- Java 1.4 (Server) en Java 1.5 (Client)
- Ant
- Eclipse ontwikkelomgeving
- Server
  - IGSO
  - MySQL + data
  - Bijhorende bundels
- Client
  - PDAClient + Login Module
  - PCClient
  - JPC/SC
  - Godot: open source middleware eID
  - eID Middleware
  - PlaceLab

Er bevindt zich ook een document op waar uitleg gegeven wordt over de installatie en het gebruik van de software. Tenslotte vonden we het ook aangewezen om een elektronische versie van onze thesis in te leveren.

Bibliografie

[1] How wifi works. http://computer.howstuffworks.com/wireless-network.htm.

[2] Nobuo Kawaguchi and Seigo Ito. Bayseian based location estimation system using wireless lan.

[3] Moustafa A. Youssef, Ashok Agrawala, and A. Udaya Shankar. Wlan location determination via clustering and probability distributions.

[4] Moustafa A. Youssef, Ashok Agrawala, A.Udaya Shankar, and Sam H. Noh. A probabilistic clustering-based indoor location determination system. 2002.

[5] Kotchakorn Maksuriwrong, Vara Varavithya, and Nachol Chaiyaratana. Wireless lan access point placement using a multi-objective genetic algorithm.

[6] Sumit Dhar. Introduction to smart cards. http://www.rootshell.be/∼dhar/smartcards.html.

[7] Iso specificaties. http://www.iso.org.

[8] Pc/sc workgroup. http://www.pcscworkgroup.com/.

[9] M.u.s.c.l.e. middleware. http://www.linuxnet.com/middle.html.

[10] Opensc project. http://www.opensc-project.org/index.html.

[11] Open card. http://www.opencard.org/.

[12] Globalplatform. http://www.globalplatform.org/.

[13] Java card. http://java.sun.com/products/javacard/.

[14] Markku Sievanen. Application protocol data unit. http://www.tml.hut.fi/Studies/T-110.497/2003/lecture4.pdf.

[15] Ietf werkgroep voor x509. http://www.ietf.org/html.charters/pkix-charter.html.

[16] R. Housley. Rfc 2459: X.509 certificate and crl profile. http://www.ietf.org/rfc/rfc2459.txt?number=2459, 1999.

[17] Pki forum. http://www.pkiforum.org/whitepapers.html.

[18] Rsa laboratories. http://www.rsasecurity.com/rsalabs/.

[19] Java cryptography architecture. http://java.sun.com/j2se/1.4.2/docs/guide/security/CryptoSpec.ht

[20] Java cryptography extension. http://java.sun.com/products/jce/.

[21] Greg Travis. Using jsse for secure socket communication. Technical report, IBM Corporation, 2002.

[22] Charlie Lai, Li Gong, Larry Koved, Anthony Nadalin, and Roland Schemers. User authentication and authorization in the java platform. Technical report, Sun, IBM, Onebox, 2000.

[23] Danny De Cock, Christopher Wolf, and Bart Preneel. The belgian electronic identity card (overview). *Lecture Notes in Informatics*, 77:298–301, 2006.

[24] Marc Stern. *Belgian Electronic Identity Card Content*. CSC, 2.2 edition.

[25] Patrick Andries. *eID Middleware Architecture Document*, 1.0 edition.

[26] K. Bogaert and M. Stern. *Belgian eID Toolkit Developer's guide*. CSC, 1.0l edition.

[27] Marc Stern. *Belgian eID Run-time Users guide*. CSC, 1.1.0 edition.

[28] Danny De Cock. eid godot software. http://homes.esat.kuleuven.be/~decockd/wiki/bin/view.cgi/Ma

[29] The osgi alliance. http://www.osgi.org/.

[30] Mysql. http://www.mysql.com/.

[31] Soap: Simple object acces protocol. http://www.w3.org/TR/soap/.

[32] Ssl: Secure sockets layer. http://wp.netscape.com/eng/ssl3/.

[33] Place lab. http://www.placelab.org/.

[34] Binghao Li, Andrew Dempster, Chris Rizos, and Joel Barnes. Hybrid method for localization using wlan.

Lijst van figuren

Lijst van tabellen