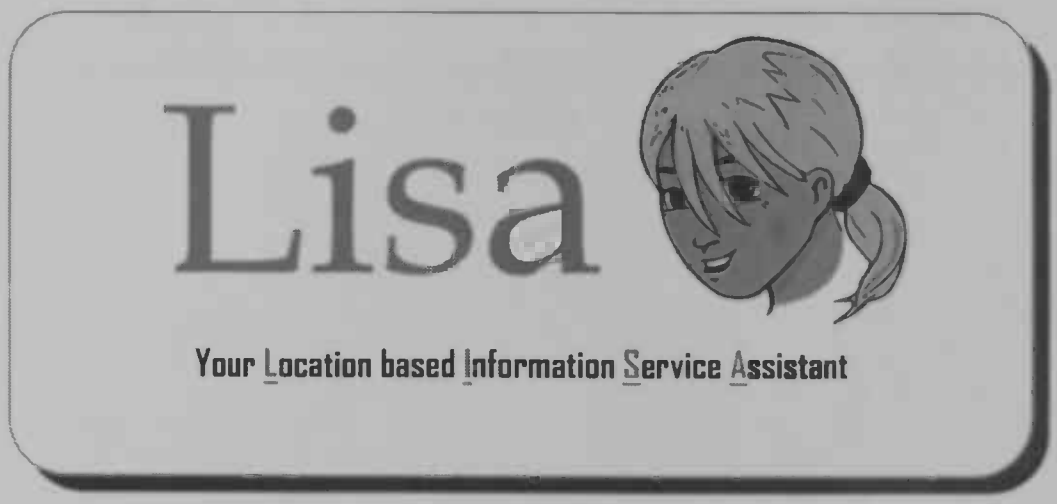
Rijks*universiteit* Groningen

RuG

Ontwerp en ontwikkeling van een *Location Based* Service

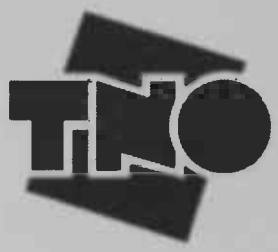
WORDT
NIET UITGELEEND



Albert B. Wijbenga

Informatica, Software and Systems Engineering (SSE)
Instituut voor Wiskunde en Informatica
Faculteit der Wiskunde en Natuurwetenschappen
Rijks*universiteit* Groningen
Begeleiders: dr. ir. J.A.G. Nijhuis en ir. S. Achterop

Afstudeerlocatie: TNO Informatie- en Communicatietechnologie
Afdeling User Centered Interaction
Groningen
Begeleider: drs. A.P.J. de Jong



April 2005

Ontwerp en ontwikkeling van een *Location Based* Service

Doctoraal thesis, vakgroep Informatica, Rijks*universiteit* Groningen.

Rijksuniversiteit Groningen
Bibliotheek Wiskunde & Informatica
Postbus 800
9700 AV Groningen
Tel. 050 - 363 40 01

Albert B. Wijbenga, s1156888.
Informatica, richting Software and Systems Engineering (SSE),
Rijks*universiteit* Groningen, april 2005.

Begeleiders R*u*G:
Dr. ir. J.A.G. Nijhuis
Ir. S. Achterop (2e begeleider)

Externe begeleider TNO:
Drs. A.P.J. de Jong

Abstract

Location Based Services is een techniek die erg in opmars is, de afgelopen vijf jaar werden er *LB* applicaties ontwikkeld. Dit verslag beschrijft het ontwerpen en ontwikkelen van een *Location Based Service*. Dit is naar aanleiding van de opdracht vanuit TNO Informatie- en Communicatietechnologie (TNO ICT). De probleemstelling van het afstudeerwerk luidt:

> *Op welke wijze kan een nieuwe* Location Based Service *applicatie op een Pocket PC (PPC) door TNO ICT zo optimaal mogelijk worden ontwikkeld, waarbij rekening gehouden wordt met de mogelijkheden en beperkingen van de bestaande technieken?*

Eerst wordt een concept van een *LB* dienst uitgewerkt en beschreven. Aan de hand hiervan worden de requirements en het functioneel ontwerp opgesteld.

De in dit verslag beschreven concept/dienst is Lisa. Lisa staat voor Locatie gebaseerde Informatie Systeem Assistent, met *Lisa* op een PDA kunnen gebruikers stadswandelingen maken en bewandelen. Dit is een soort vervanging van het papieren stadsgids. Onderweg krijgen de gebruikers digitaal informatie over locaties, en daarnaast ontvangen zij berichten van andere gebruikers op hun PDA.

Vanwege het *Location Based* aspect van *Lisa* wordt er gekeken naar de bestaande lokalisatie technieken (GPS, Cell-ID, etc.), de verschillende communicatie methoden (UMTS, GPRS, etc.) en protocollen (XMPP, SOAP, etc.). Naar aanleiding van een vergelijkingsonderzoek wordt een keuze gemaakt welke technieken ingezet zullen worden voor de applicatie. Het is voor TNO ICT interessant om verschillende (nieuwe) technieken bij elkaar te brengen.

Naast de verschillende technieken wordt het ontwerp van de architectuur beschreven, die verschillende technieken in huis heeft. Deze architectuur is zodanig dat er geen conflicten zullen optreden tussen al die verschillende technieken. Het *Instant Messaging* protocol XMPP wordt op een uitzonderlijke manier ingezet als communicatie protocol voor het over en weer zenden van data tussen de cliënt en de server.

Een eis vanuit TNO ICT is dat er gebruik wordt gemaakt van het LB platform MiloWare. In dit verslag wordt beschreven hoe hier gebruik van wordt gemaakt. Een andere eis is dat Microsoft MapPoint wordt gebruikt. Het blijkt dat er veel nadelen van MapPoint zijn, in dit verslag wordt beschreven hoe MapPoint toch optimaal ingezet kan worden.

Ten slotte wordt in dit verslag op aanbevelingen voor verder onderzoek gedaan, en suggesties voor eventuele features die in de toekomst in *Lisa* geïmplementeerd kunnen worden. Hierbij wordt ook de eindconclusie weergegeven, over hoe TNO ICT op een optimale manier een *Location Based Service* kan ontwerpen en ontwikkelen.

Voorwoord

Na vijf en een half jaar Informatica aan de RuG, vormt dit afstudeerverslag het eindresultaat van dat werk. Op het moment van schrijven zit ik nog bij TNO Informatie- en Communicatietechnologie (TNO ICT), middenin het vierdejaarsvak 'Bedrijfsstage'. Normaal gesproken dient dat vak vóór het afstudeergedeelte afgerond te zijn, maar in april 2004 viel mijn oog op een zeer uitdagende afstudeeropdracht van TNO ICT. De opdracht was te mooi om waar te zijn en zeer uitdagend, deze opdracht kon ik niet laten lopen. Dit verslag beschrijft de uitwerking van deze opdracht.

Dit verslag bevat 112 pagina's, een getal wat op zich niks zegt over de inhoud. Mijn afstudeerdocent Nijhuis zei al in het begin van de afstudeerperiode: "Ik wil geen 'dikmakerij' in je verslag zien". Ik ben van mening dat ik dat zo goed mogelijk heb vermeden.

Dank aan iedereen die in welke vorm dan ook heeft meegewerkt aan deze afstudeeropdracht. Dit zijn de mensen die rondom mij aanwezig waren gedurende de periode, in het bijzonder Sven Schultz, Arnoud de Jong en Karen Groenink. Naast deze mensen wil ik Jos Nijhuis ook bedanken voor zijn feedback tijdens het schrijven van dit verslag.

Met Sven Schultz heb ik heel erg prettig mogen samenwerken aan de ontwikkeling van *Lisa*. Zijn relaxte houding ten aanzien van mijn Doofheid vond ik zeer goed. Ik ben van mening dat Sven en ik een mooi product hebben ontworpen en ontwikkeld, wat ook blijkt uit de belangstelling van diverse partijen (het Groninger Museum, Marketing Groningen, het Algemeen Dagblad).

Ook wil ik Corrie Spaan bedanken voor haar enorme flexibiliteit en haar inzet tijdens het tolken. De studie was heel anders gelopen als dat niet het geval was.

Tot slot wil ik ook Marieke Wolthuis, mijn vriendin, bedanken voor haar input ten aanzien van mijn eindverslag: "Schrijf dit op, vermeld dit en dat in je verslag. Dit vindt jouw docent interessant.".

Ikzelf vond het zeer interessant en uitdagend om bij TNO ICT te mogen werken, ik heb kennis mogen maken met allerlei technologieën (UMTS, GPS, RFID, XMPP, Pocket PC 2003, Visual Studio .NET 2003, C#, etcetera). Vooral het programmeren in C#, in een .NET omgeving, vond ik erg leuk en interessant. Het is voor mij een hele ervaring om in een TNO-innovatieve omgeving te mogen werken, een front van de nieuwste technologische ontwikkelingen.

Het (benodigde) ontspanningsaspect met collega's tijdens de stage was aanwezig in de vorm van: tafelvoetballen, kitesurf-dag, naar de film gaan, de ICT-Expo Noord evenement in de Martiniplaza, de TNO-dag in Marine-vliegkamp Valkenburg, etcetera. Hiermee vormt TNO ICT een zeer aangename werkomgeving, waar ik met grote plezier mijn afstudeerstage heb gedaan.

Ik wens u veel plezier met het lezen van dit verslag, en ik hoop dat dit u meer inzicht geeft bij het ontwikkelen van een *Location Based* applicatie, en in het algemeen over *Location Based Services*.

Albert Wijbenga

Inhoudsopgave

Abstract 4

Voorwoord 5

Inhoudsopgave 6

Hoofdstuk 1. De afstudeeropdracht 8

1.1. Opdrachtomschrijving 8
1.2. Probleemstelling 9
1.3. Doelstellingen 9
1.4. De dienst Lisa 9
1.4.1. Functioneel ontwerp van *Lisa* 12
1.5. Afstudeerlocatie 14
1.6. In dit verslag 15

Hoofdstuk 2. Lokalisatie technieken voor *Lisa* en scanning 17

2.1. Lokalisatie techniek selectiecriteria 17
2.2. Global Positioning System (GPS) 20
2.2.1. Elementen van GPS 20
2.2.2. Galileo, de Europese GPS variant 21
2.3. 2G/3G netwerken positionering 22
2.3.1. De cell-id methode 22
2.3.2. Radio-based lokalisatie 23
2.3.3. Network Assisted GPS (AGPS) 25
2.3.4. Wireless Fidelity (WiFi)/Wide Local Area Network (WLAN) 26
2.3.5. Radio Frequency Identification (RFID) 27
2.4. Scanning techniek 29
2.4.1. Radio-Frequency Identification (RFID) 30
2.4.2. Barcode scanning 30
2.4.3. Smartcard scanning 30
2.4.4. Infrarood scanning 30
2.5. Location Based Platform: *MiloWare* 31
2.5.1. Architectuur 32
2.5.2. *Milo* webservices 32
2.6. Conclusie 33

Hoofdstuk 3. Mobiele communicatie technieken en protocollen 37

3.1. General Packet Radio Service (GPRS) 37
3.2. Universal Mobile Telecommunication System (UMTS) 38
3.3. De communicatieprotocollen 39
3.3.1. Instant Messaging (IM) 40
3.3.2. XML 41
3.3.3. XMPP protocol (Jabber) 41
3.3.4. SOAP (Webservices) 45
3.4. Conclusie 47

Hoofdstuk 4. Productontwikkeling *Lisa* **49**

4.1. Architectuur en implementatie 49
4.1.1. Interne communicatie van het systeem 51
4.1.2. *MiloWare* 57
4.1.3. Jabber 61
4.2. Microsoft MapPoint 62
4.2.1. Wat is MapPoint en is het wat? 63
4.2.2. MapPoint webservices code 64
4.2.3. Licentie MapPoint 65
4.3. Data/Object model 66
4.3.1. De database van *Lisa* 66
4.3.2. De objecten van *Lisa* 67
4.4. GPS in .NET 68
4.5. Structuur van de applicatie 69
4.6. De Grafische User Interface (GUI) 70

Hoofdstuk 5. Lisa en haar toekomst **75**

5.1. Aanbevelingen voor verder onderzoek 75
5.2. Features voor Lisa in de toekomst 75
5.3. Conclusie 76

Gebruikte afkortingen **78**

Geraadpleegde literatuur- en bronnenlijst **80**

Appendices **83**

Appendix I: Opdrachtomschrijving TNO ICT en projectorganisatie 83
Appendix II: Ontwikkelmethode 86
Appendix III: GPS doorgelicht 88
Appendix IV: WiFi lokalisatie algoritme 92
Appendix V: Use-cases 94
Appendix VI: MapPoint code 109
Appendix VII: GPS code 111

Hoofdstuk 1. De afstudeeropdracht

Inleiding

Deze afstudeeropdracht is uitgegeven door TNO Informatie- en Communicatietechnologie (TNO ICT). Vanuit TNO ICT bestond de vraag om een *Location Based Service* (LBS) applicatie te ontwikkelen, een techniek die erg in opmars is. In samenwerking met andere partijen ontwikkelt TNO ICT diensten en technieken. Het doel van de ontwikkeling van deze LBS applicatie is dat TNO ICT aan de hand van deze applicatie naar klanten toe kan demonstreren wat de mogelijkheden zijn op het gebied van *Location Based Services.*

De algemene opdrachtomschrijving vanuit TNO ICT luidt als volgt:

> *"Ontwerp en ontwikkel een* LBS *dienst in .NET die gebruik maakt van het* LBS-*platform. De dienst moet gaan draaien op een Pocket PC. Aan de ene kant communiceert de dienst met het* LBS *platform. Het platform voorziet de cliënt van locatie informatie, bijvoorbeeld waar iemand zich bevindt en meer. Daarnaast communiceert de applicatie met de MapPoint server. In beide gevallen gebeurt de communicatie via webservices. Via het ontwerpproces, dat zal bestaan uit het maken van een functioneel ontwerp, grafisch ontwerp en objectmodel, zal een uiteindelijke keuze gemaakt worden wat er definitief gebouwd zal gaan worden."*

Deze afstudeeropdracht is duaal uitgevoerd door Sven Schultz, afkomstig van Human Technology (Hanzehogeschool) en Albert Wijbenga. Hierbij lag de verantwoordelijkheid voor het ergonomische aspect (met name de Grafische User Interface) bij Sven Schultz. In dit verslag zal eveneens kort op het GUI-ontwerp worden ingegaan. De softwarematige ontwikkeling van de dienst werd ontworpen en uitgevoerd door Albert Wijbenga.

De afstudeerders waren elk verantwoordelijk voor hun eigen aspect bij het uitvoeren van de opdracht. Om die reden is er gekozen om voor beide apart een opdrachtomschrijving, een probleemstelling en doelstellingen te formuleren. In dit verslag worden enkel de werkzaamheden van Albert Wijbenga omschreven, tenzij nadrukkelijk is aangegeven dat het andermans werk betreft (in dit geval het werk van Sven Schultz).

In dit hoofdstuk worden de opdrachtbeschrijving, de probleemstelling en de doelstellingen besproken (paragraaf 1.1 tot en met 1.3). Daarna wordt het dienstconcept, *Lisa*, en LBS in het algemeen nader omschreven in paragraaf 1.4. In deze paragraaf wordt ook de omschrijving van de requirements opgenomen. In paragraaf 1.5 wordt kort de afstudeerlocatie besproken, daarna wordt in paragraaf 1.6 een vogelvlucht door het gehele verslag weergegeven.

1.1. Opdrachtomschrijving

De afstudeeropdracht omschrijving vanuit TNO Informatie- en Communicatietechnologie voor Albert Wijbenga luidt als volgt:

> *"Ontwerp de architectuur en ontwikkel in .NET de software voor de* LBS *dienst die op een Pocket PC dient te draaien. De structuur hiervan moet voldoen aan de eisen van de gebruiker en naar functionaliteit, en hanteert het* client-server *model. Voor het* Location Based *aspect dient literatuuronderzoek naar de verschillende lokalisatie en communicatie technieken gedaan te worden, hieruit volgt de beslissing welke technieken gebruikt zullen worden."*

1.2. Probleemstelling

Uit de opdrachtomschrijving wordt de onderstaande probleemstelling geformuleerd, geldende voor Albert Wijbenga.

> *Op welke wijze kan een nieuwe* Location Based Service *applicatie op een Pocket PC (PPC) door TNO ICT zo optimaal mogelijk worden ontwikkeld, waarbij rekening gehouden wordt met de mogelijkheden en beperkingen van de bestaande technieken?*

1.3. Doelstellingen

Aan de hand van de probleemstelling kunnen een aantal doelstellingen worden geformuleerd. Deze dienen gerealiseerd te worden voor het einde van de afstudeerperiode, te weten op 6 maart 2005. De realisatie van de stellingen is vastgelegd in dit afstudeerverslag, indien van toepassing in een bijlage.

- Concipieer een *LB* dienst, en werk deze dienst uit.
- Breng de methoden en technieken voor lokalisatie in kaart.
- Breng de verschillende protocollen op het gebied van client-server communicatie in kaart.
- Breng conclusies en aanbevelingen uit naar aanleiding van het uitgevoerde literatuuronderzoek.
- Ontwikkel de *LB* dienst in .NET, waarbij de geselecteerde lokalisatie methode(n) wordt gehanteerd, en de geselecteerde (mobiele) communicatie techniek(en).
- Maak gebruik van de Microsoft MapPoint Webservices.
- Maak gebruik van het *Location Based* platform *MiloWare.*

1.4. De dienst Lisa

Location Based Services zijn alle producten en diensten die worden aangeboden aan klanten op basis van de geografische positie van de klant. De ontwikkeling van LBS is ongeveer 5 jaar geleden begonnen, ondanks de relatief vele jaren, staat de ontwikkeling nog in zijn kinderschoenen. Toch zijn de verwachtingen ten aanzien van LBS diensten hoog gespannen, bijvoorbeeld bij de mobiele (netwerk) operatoren:

> *"LBS will be as important for Mobile Operators in 2007 as SMS is currently"*

Bron: Ovum report November 2003

LBS diensten kunnen gebruik maken van een aantal verschillende lokalisatietechnieken bijvoorbeeld Cell-id, GPS, WiFi, Bluetooth of RFID. Al deze technieken zijn in staat om de positie van de gebruiker te bepalen. De genoemde technieken worden ook in dit verslag omschreven.

Voorbeelden van (bestaande) LBS diensten zijn:

- Location Based Informatie:
  Informatie die wordt aangeboden aan de klant welke gebaseerd is op zijn/haar huidige locatie. Een voorbeeld hiervan is het vinden van een restaurant in de huidige regio en speciale behoeften op dat moment. Een voorbeeld hiervan is de Starbucks Coffee dienst, waarmee men alle Starbucks Coffeeshops in de buurt kan vinden.
- 112 Hulpdiensten:
  Hulpdiensten kunnen door middel van LBS de exacte locatie van de beller achterhalen. In Amerika dienen alle operators van mobiele communicatie een bepaalde nauwkeurigheid op het gebied van lokalisatie te garanderen voor mensen die het 911 alarmnummer bellen.
- Tracking diensten:
  Deze uitgebreide categorie biedt bijvoorbeeld bedrijven de mogelijkheid om de precieze locatie van de auto's van hun wagenpark te bepalen. Een ander voorbeeld is dat postorderbedrijven perfect haar klanten op de hoogte kan houden van de status van het pakket onderweg.
- Autonavigatie:
  Er zijn momenteel veel navigators voor in de auto op de markt, een voorbeeld is de bekende TomTom Navigator. Deze dienst begeleidt de berijder van A naar B.

Tijdens brainstormsessies in de eerste fase van het afstudeerproject kwamen er enkele concepten van LBS diensten voor op de PDA naar voren:

- Taxicentrale
  Deze dienst houdt in de gaten waar de taxi's zich bevinden. De gebruiker kan waar dan ook een taxi-oproep doen, de dienst bekijkt dan welke taxi het meest in de buurt is. De chauffeur van die taxi wordt dan geattendeerd op de oproep, en haalt de klant af. De taxi's kunnen bij deze dienst aangeven of ze wel of niet bezet zijn.
- Dating service
  Potentiële daters kunnen zich voor deze dienst inschrijven. De dienst bekijkt dan of er een match, iemand die ook een date wil volgens bepaalde criteria, in de buurt is. Is dat het geval, dan worden beide hierop geattendeerd. Hierbij kan ook onderscheid worden gemaakt in bijvoorbeeld: uitgaan, concerten, liefde en sporten.
- Reisinformatie
  De gangbare methode om onderweg OV reisinformatie te verkrijgen is via het bellen van het 9292 nummer. In plaats hiervan kunnen de gebruikers met behulp van hun locatie via de nieuwe dienst sneller reisinformatie verkrijgen. Dus in plaats van zelf aan te geven waar hij of zij zich bevindt, wordt hem of haar direct de up-to-date informatie gepresenteerd.
- Stadswandelingen
  Bezoekers van een stad kunnen stadswandelingen lopen onder begeleiding. Tijdens het lopen van een dergelijk route wordt hen informatie aangeboden. Dit is een alternatief voor de papieren stadswandelingen. Daarnaast kunnen ze bijvoorbeeld ook zelf routes maken en deze aan anderen beschikbaar stellen.
- Uitgaansinformatie
  Elke dag kan er uitgaansinformatie worden aangeboden in het gebied of de gebieden die de gebruiker geselecteerd heeft. De gebruiker kan ook zien of er bekenden in de buurt zijn die misschien ook interesse hebben.

- ANWB campinggids / Hotels
  Europese campinggids voor op je Pocket PC. Door middel van een landkaart kan een camping in de buurt van jouw verblijf worden gevonden en geboekt. Naast campings kan dit ook gelden voor hotels of andere vormen van accommodatie.
- City Game
  Een interactief spel, waarbij de locatie van de speler een rol speelt. De speler kan in een stad meedoen aan een opgezet spel waarbij hij/zij zich door een stad moet verplaatsen op zoek naar informatie. Gedurende het spel wordt hem/haar interessante informatie aangeboden. Dit komt overeen met 'stadswandelingen', maar dan met een spelelement.
- Dwaaldetectie voor zorginstellingen
  Vanwege de krappe zorg in Nederland zijn instellingen voor demente bejaarden niet altijd in staat alle bejaarden te 'bewaken', dat ze niet zomaar de deur uitgaan. Dit kan met behulp van deze dienst worden vergemakkelijkt. Deze dienst houdt de locaties van de bejaarden in de gaten. Er hoeft dan alleen te worden ingegrepen als er daadwerkelijk een demente bejaarde onbedoeld de deur uitgaat.

Na enkele brainstormsessies is de keuze gevallen op '*stadswandelingen*'. De argumenten voor de keuze zijn het volgende:

- De opdracht wordt haalbaar geacht in de gestelde tijdsperiode.
- De opdracht is technisch gezien ook haalbaar.
- De opdracht is onafhankelijk van derden te realiseren, d.w.z. dat de opdracht geheel binnen het project uitgevoerd kan worden.
- Er is een grote en brede (mogelijke) gebruikersgroep.
- Er is interesse van een bestaande klant, namelijk Cultuur in de buurt, dit is een project waarbij Zappwerk en Surfnet betrokken zijn.
- Deze keuze heeft de voorkeur vanuit TNO ICT en de betrokken afstudeerders.

Met deze keuze is het dienstconcept *Lisa* ontstaan. De afkorting *Lisa* staat voor: *L*ocatie gebaseerde *I*nformatie *S*ervice *A*ssistent. *Lisa* biedt de volgende mogelijkheden:

> *"Bezoekers van een stad kunnen stadswandelingen lopen onder begeleiding. Tijdens het lopen van een dergelijke route wordt hen informatie over de locaties, die bij de route horen, aangeboden. Tijdens het wandelen kan de gebruiker ook objecten 'scannen', en extra informatie over deze objecten ontvangen. Dit is een alternatief voor de papieren stadswandelingen. Daarnaast kunnen ze bijvoorbeeld ook zelf routes maken en deze aan anderen beschikbaar stellen."*

Op de volgende pagina wordt de dienst in prozastijl beschreven, waaruit blijkt welke technieken benodigd zijn en welke eisen daarbij behoren.

De Amsterdamse Piet wil zijn vriend Jan bezoeken, Jan woont in Groningen. Op een dag komt Piet aan in het Groninger hoofdstation. Piet is in het bezit van een PDA met daarop de dienst. Piet is door Jan ingelicht dat Jan al een route voor Piet heeft gemaakt, waarbij hij de hotspots van Groningen tegenkomt onderweg naar het huis van Jan. Piet start de dienst op, en pikt Jan's route op. Piet ziet direct dat het startpunt van de route het Station is, direct daarna krijgt hij een beschrijving van de renovatie van de stationshal. Geïnteresseerd leest Piet het verhaal, hierna loopt hij naar buiten. Op de kaart, op de PDA, ziet hij dat hij naar het water moet lopen. Daar ziet hij de markante 'gouden toren' van het Groninger Museum. Op dat moment wordt hij geattendeerd dat hij in de buurt van het Groninger Museum is. Ook krijgt hij hierover informatie. Een paar seconden later krijgt Piet de melding dat er een bericht is

achtergelaten op die locatie. Piet leest het bericht, en ziet dat het van Jan afkomstig is. Jan vertelt dat de koffie in het museumrestaurant heerlijk is, en dat de gouden toren in de zon echt op goud lijkt. Op dat moment breekt de zon door de wolken, de toren begon te glanzen als goud! Piet loopt door, naar de Folkingestraat, op dat moment krijgt hij te horen dat hij in de buurt is van een joodse synagoge. Geïnteresseerd leest Piet de verzonden informatie, ondanks de informatie weet Piet toch niet wat de tekens boven de deur van de synagoge betekenen. Piet laat een bericht achter:"Wat betekenen de tekens boven de deur?". Piet komt later vast wel weer terug, dan is er vast een reactie. Hij loopt door de Folkingestraat, en ziet een beeld staan. Op dat beeld staat een nummer, Piet houdt zijn PDA bij dat nummer. Een paar seconden later krijgt hij informatie over dat beeld. Piet loopt door en ziet de Korenbeurs, met daarin een supermarkt. Prompt krijgt hij een bericht binnen:"Piet, wil je even een brood meenemen? Ben platzak op dit moment". Piet doet even boodschappen, hierna kijkt hij op de kaart, en ziet dat het huis van Jan al aardig in de buurt is, Piet loopt in de oostelijke richting, volgens de route op de kaart. Piet komt langs het News Café, hier krijgt hij wederom een bericht:"Kom even binnen, ik zit hier, Jan".

In de volgende subparagraaf worden de requirements en de benodigde technieken afgeleid uit het bovenstaande verhaal.

1.4.1. Functioneel ontwerp van *Lisa*

Uit het hierboven beschreven scenario kunnen de requirements worden afgeleid. Deze zijn door beide afstudeerders in overleg met de begeleiders vanuit TNO ICT vastgesteld. De beschreven requirements vormen de rode draad bij het gehele ontwikkelingsproces. In de volgende hoofdstukken worden ze stuk voor stuk behandeld, waarbij tevens wordt besproken hoe ze worden vervuld.

User requirements

Om een route te kunnen creëren of te kunnen opvragen moet de gebruiker in staat zijn tot het volgende.

De gebruiker heeft de mogelijkheid om:

i. Een route op te vragen, hierbij wordt onderscheid gemaakt tussen categorieën: populair, officieel en gebruiker.
Dit vergroot de gebruikersvriendelijkheid van de dienst, de gebruiker krijgt hierdoor het gevoel dat hij of zij meer zeggenschap heeft over de dienst. Vanuit technisch oogpunt heeft het ook een voordeel: categoriseren van de routes heeft minder dataverkeer tot gevolg.

ii. Af te lezen waar hij zich bevindt en begeleid te worden naar de volgende locatie op de route.
Dit is een essentieel Location Based *aspect, het draait allemaal om de locatie waar de gebruiker zich bevindt.*

iii. Informatie op locatie over de locaties te ontvangen en te lezen.
Dit spreekt voor zich.

iv. Op locatie berichten achter te laten en berichten van andere gebruikers te lezen.
Deze feature vergroot de interactiviteit tussen de gebruiker en de dienst.

v. Een nieuwe route aan te maken en te delen met andere gebruikers.
Dit is de hoofdfeature van de dienst, het bewandelen, het maken en het delen van routes.

vi. Nieuwe locaties en diens beschrijvingen aan te maken.
Een route bestaat immers uit een rij oriëntatiepunten c.q. locaties.

vii. Onderweg objecten te 'scannen', en informatie te verkrijgen over de desbetreffende objecten.
Dit is een extra mogelijkheid om onderscheid te kunnen maken tussen objecten die dichtbij elkaar zijn, hieronder kan bijvoorbeeld worden verstaan: een speciaal tegeltje, een beeld, enzovoort.

viii. Wandelingen van maximaal 2,5 kilometer lang te creëren.
Uit onderzoek bij de VVV is gebleken dat de stadswandelingen gemiddeld 2,5 kilometer lang zijn. Dit getal wordt dan ook overgenomen.

Functionele requirements

Om aan de user requirements te kunnen voldoen moet de dienst bepaalde technieken en bepaalde mogelijkheden bevatten:

1. Het moet een lokalisatie techniek bevatten.
2. Het moet de beschikking hebben over een scan methode.
3. De PDA moet draadloos kunnen communiceren met het LBS platform.
4. De informatie moet binnen een acceptabele tijd over en weer worden verzonden tussen de cliënt en de server.
5. Beveiliging in de vorm van gebruikers authenticatie is gewenst.
6. Het moet de beschikking hebben over geografische kaartgegevens.
7. De locaties en de routes moeten 'ergens' worden opgeslagen.

Performance requirements

Aangezien de dienst op een PDA dient te draaien, zijn er enkele performance eisen aan de PDA.

De PDA moet kunnen worden uitgerust met:

1. Een lokalisatietechniek.
2. Draadloze communicatietechniek.
3. Een scan techniek.
4. Een operating system, waarop de applicatie kan draaien.

Een eis van TNO ICT is dat de applicatie wordt ontwikkeld in .NET, met als programmeertaal C#. Het ligt daarom voor de hand dat de operating system op de PDA Pocket PC 2002/2003 is.

Potentiële doelgroep/gebruikersgroep

De doelgroep houdt mensen in die:

- actief een route willen maken/navigeren.
- zelf een PDA bezitten, weten hoe ze met Pocket PC 2002/2003 moeten omgaan.
- ervaring met het gebruik van PDA's in combinatie met een lokalisatie techniek en een draadloze communicatietechniek.
- het interessant, leuk vinden om extra informatie te verkrijgen van moderne innovatieve middelen.
- leergierig, nieuwsgierig zijn.
- in staat zijn Nederlands te kunnen schrijven en te lezen.

Use-cases

In Appendix V is de uitwerking van de use-cases opgenomen. Deze zijn naar aanleiding van de bovenstaande requirements in samenwerking met Sven Schultz uitgewerkt.

1.5. Afstudeerlocatie

Inleiding
De afstudeerlocatie is TNO Informatie- en Communicatietechnologie, gevestigd in Groningen. Het is één van de zeer weinige R&D instituten in Noord Nederland, welke veel innovatief werk verzet.

De afstudeeropdracht wordt binnen het UCI (User Centered Innovation) afdeling uitgevoerd. Dat afdeling is een innovatieteam waarin R&D'ers met kennis van de markt, gebruikersbehoeften, gebruikersinterfaces, technologie en dienstontwikkeling op het gebied van mobiele communicatie samenwerken in multidisciplinaire teams.

Hieronder is een quote van de website van TNO, waar in vogelvlucht het bedrijfsprofiel en de geschiedenis van TNO ICT wordt doorgenomen.

Bedrijfsprofiel
"TNO Informatie- en Communicatietechnologie is een uniek centrum van innovatie op het gebied van telecommunicatie en informatietechnologie in Nederland. TNO helpt bedrijven in uiteenlopende sectoren succesvol te innoveren. Dit kan een nieuw product of dienst opleveren, een compleet nieuwe werkwijze of een nieuwe strategische visie op de toekomst. TNO Informatie- en Communicatietechnologie staat niet alleen voor technologische kennis, maar beschikt ook over deskundigheid op sociaal-cultureel en bedrijfseconomisch gebied. Dankzij deze unieke combinatie kunnen innovatieve producten en diensten, netwerken, platformen en ICT-oplossingen worden ontwikkeld en gebouwd. Daarnaast wordt op de haalbaarheid binnen de bedrijfsvoering van de klant of in de markt getoetst.

TNO Informatie- en Communicatietechnologie maakt gebruik van de diensten van driehonderd onderzoekers en een aantal aan diverse universiteiten verbonden professoren. We zijn onderdeel van TNO, een toonaangevende onafhankelijke contractresearch-organisatie waarvan de expertise en het onderzoek in belangrijke mate bijdragen aan de concurrentiekracht van bedrijven en organisaties, aan de economie en de kwaliteit van onze maatschappij als geheel.

Meer dan 50 jaar geleden realiseerde PTT, het toenmalige Nederlandse staatsbedrijf voor telecommunicatie en post, zich al dat structureel onderzoek en ontwikkeling cruciaal is voor de bedrijfsvoering. In die dagen ontwikkelden apparatuurleveranciers meestal systemen in eigen huis. Standaardisatie stond nog in de kinderschoenen en basisonderzoek was nodig om het Nederlandse netwerk snel en tegen lage kosten te ontwikkelen. Door het inrichten van een centraal laboratorium (het Dr. Neher Laboratorium, later PTT Research en KPN Research geheten) in 1953 bewees het bedrijf dat het R&D zeer serieus nam.

In 2002 werd het besluit genomen dat specifieke R&D wel nodig was, maar niet langer binnen de KPN organisatie behoefde plaats te vinden. Op 1 januari 2003 is TNO Telecom opgericht door de overgang van KPN Research naar TNO, waaraan ook onderdelen vanuit bestaande TNO-instituten worden toegevoegd die zich specialiseren op telecommunicatie en ICT gebied. De nieuwe combinatie is dan ook gebaseerd op meer dan vijftig jaar ervaring in

onderzoek op het gebied van telecommunicatie en informatietechnologie, alsmede op de sterke klantrelaties binnen de publieke en telecomsector.

Per 1 januari 2005 bestaat TNO uit vijf kerngebieden in plaats van uit vijftien instituten. Hierdoor ontstaat een grotere kritische massa en kan TNO zich beter richten op de grote maatschappelijke thema's. De voormalige instituten TNO Telecom en een gedeelte van TNO Strategie, Technologie en Beleid zijn opgenomen in TNO Informatie- en Communicatietechnologie. Een paar voorbeelden van ons historisch track record. De eerste elektronische rekenmachine (PTERA = PTT Electronische Reken Automaat) is gebouwd in 1953 en werd ingezet bij verkeerssimulaties en stagnatieberekeningen aan het telefonienetwerk. Eén jaar later werd een krachtigere opvolger ontwikkeld, de ZEBRA (Zeer Eenvoudige Binaire Reken Automaat). Zestig ZEBRA's vonden hun weg naar eigenaren over de hele wereld. Er werden onder andere successen behaald op het gebied van automatische postsortering, handschriftherkenning en telex via radio. Voorts speelde het bedrijf een belangrijke rol in internationale standaardisatiecommissies, zoals GSM, UMTS, ADSL en ATM, als ook met betrekking tot standaarden voor gepercipieerde spraakkwaliteit."

Bron: http://www2.tno.nl/informatie- en communicatietechnologie/wie we zijn/

1.6. In dit verslag

In de volgende hoofdstukken worden de functionele requirements 1 tot en met 7 aan de hand van diverse selectiecriteria behandeld. Hieruit worden, met argumenten, beslissingen genomen welke technieken gebruikt zullen worden.

Om aan de requirements te voldoen zijn er diverse onderzoeken gedaan. Tijdens de brainstormfase is het concept, welke staat beschreven in paragraaf 1.4, in samenwerking met Sven Schultz uitgedacht. Parallel hieraan wordt een literatuuronderzoek naar de lokalisatie technieken uitgevoerd, dit is uitgebreid beschreven in hoofdstuk 2, *Lokalisatie technieken voor* Lisa *en scanning*. In dit hoofdstuk worden de requirements 1 en 2 behandeld. Dit onderzoek is geheel door Albert Wijbenga uitgevoerd.

In hoofdstuk 3, *Mobiele communicatie technieken en protocollen*, worden de functionele requirements 3, 4 en 5 behandeld. Binnen TNO ICT zijn er veel praktijkervaringen op het gebied van protocollen binnen mobiele communicatie, dat wordt ook benut in dit verslag. Naast de praktijkervaringen worden de resultaten van het literatuuronderzoek naar de protocollen gepresenteerd. Ook dit behoort tot het afstudeerwerk van Albert Wijbenga.

Verder worden de functionele requirements 6 en 7 in hoofdstuk 4, *Productontwikkeling Lisa*, behandeld. In dat hoofdstuk zal ook uitgebreid worden ingegaan op de architectuur van *Lisa*, vanuit oogpunten van Software Engineering. Tevens wordt in dat hoofdstuk het functioneel ontwerp beschreven. In dit hoofdstuk wordt ook het Grafische User Interface besproken, dit is voornamelijk het werk van Sven Schultz.

De bijlagen bevatten:

- De exacte opdrachtomschrijving van TNO ICT en projectorganisatie.
- De wiskundige achtergronden van de GPS en WiFi lokalisatietechnieken.
- Een schematische weergave van eXtreme Programming.
- De uitgewerkte use-cases.
- Fragmenten van de source code.

Tot slot wordt in hoofdstuk 5 de beantwoording van de probleemstelling behandeld. Hieruit komen aanbevelingen voort, over het hoe van een gedegen ontwikkeling van een *Location Based* applicatie.

Hoofdstuk 2. Lokalisatie technieken voor *Lisa* en scanning

Inleiding

De eerste functionele requirement is: *het moet een lokalisatie techniek bevatten.* Om te bepalen welke lokalisatie techniek geschikt is volgt in dit hoofdstuk een verslag van het literatuuronderzoek naar de verschillende lokalisatietechnieken. Aangezien er bepaald is dat de dienst op een *Personal Digital Assistent* (PDA) dient te draaien, zijn er veel mogelijkheden om de geografische positie te bepalen. Het device vormt dus geen beperking, want de PDA kan, via Bluetooth, gekoppeld worden met andere devices die dan ook weer andere lokalisatie mogelijkheden bieden.

Hieronder is een overzicht gepresenteerd van de lokalisatie methoden die momenteel bestaan, ze worden in de volgende paragrafen behandeld aan de hand van in de paragraaf 2.1 genoemde selectiecriteria.

1. Global Positioning System (GPS), in appendix III wordt de wiskundige achtergrond van deze techniek uitvoerig behandeld.
2. Galileo, de Europese GPS variant.
3. Positionering in 2G/3G netwerken.
4. Wireless Fidelity (WiFi) / Wireless Local Area Network (WLAN), in appendix IV wordt de wiskundige achtergrond van deze techniek behandeld.
5. Radio-Frequency Identification (RFID).

De lokalisatietechnieken worden behandeld in de paragrafen 2.2 en 2.3.

Naast de lokalisatie techniek bestaat ook de functionele requirement 2, *het moet de beschikking hebben over een scan methode.* Dat wordt ook in dit hoofdstuk behandeld. Drie scantechnieken zullen worden behandeld.

1. Radio-Frequency Identification (RFID).
2. Barcode scanning.
3. Infrarood scanning.

De genoemde scantechnieken zullen worden behandeld in paragraaf 2.4.

Verder zal in paragraaf 2.5 het Location Based platform, *MiloWare*, worden omschreven. Hierna volgt de conclusie welke technieken gebruikt zullen worden voor de dienst, *Lisa.*

2.1. Lokalisatie techniek selectiecriteria

Er zijn momenteel veel verschillende lokalisatie technieken ontwikkeld. Om een overwogen keuze te maken worden selectiecriteria bepaald. Hierbij dient ook opgemerkt te worden dat niet alle lokalisatietechnieken onder de loep worden genomen. Een voorbeeld hiervan is de Bluetooth techniek. Alleen GPS, Galileo, 2G/3G netwerken, WiFi en RFID worden onder de loep genomen.

De selectiecriteria zijn:

- Beschikbaarheid
- Nauwkeurigheid (in meters)
- Tijdsduur (seconden)

- Techniek (station-based of triangulatie)
- Betrouwbaarheid (laag-gemiddeld-hoog)
- Software (COTS: ja of nee)
- Hardware (devices)
- Ontsluitingsmethode

De genoemde selectiecriteria zullen hieronder stuk voor stuk worden behandeld. Ze zijn opgesteld naar aanleiding van de beschikbare technieken die naar voren kwamen tijdens het literatuuronderzoek.

Beschikbaarheid
Dit houdt in hoeverre de techniek benut kan worden, dit kan zijn: wereldwijd, nationaal, regionaal. Hiernaast wordt ook beoordeeld of de techniek buitenshuis of binnenshuis gebruikt kan worden. Voor sommige technieken zijn externe netwerken/scanners/identificatie-kenmerken benodigd om de positie te kunnen bepalen. Ook dit wordt meegewogen.

Nauwkeurigheid
De nauwkeurigheid wordt uitgedrukt in aantallen meters, dat getal geeft dé foutmargé weer. Dus als een positie is berekend, dan bevindt de werkelijke locatie binnen de straal van die marge.

Tijdsduur
Het is bij een *LB* applicatie van belang dat de positie real-time wordt berekend. De tijdsduur wordt uitgedrukt in aantallen seconden.

Techniek
Er zijn grofweg twee technieken; - station-based en - triangulatie. Voor de eerste kan de positie niet cliënt-sided worden bepaald, maar is afhankelijk van de positie van het station. Met andere woorden, de positie van het station is eveneens de positie van de cliënt.

Betrouwbaarheid
Dit hangt een beetje samen met de nauwkeurigheid en beschikbaarheid, maar andere factoren kunnen een rol spelen, zo kunnen er instanties zijn die invloed hebben op in hoeverre de techniek beschikbaar is.

Software
Zijn er *commercial off-the-shelf* (COTS) componenten beschikbaar, om de positie te verkrijgen? En open source?

Hardware
Welke externe apparatuur is benodigd om de positie te bepalen? Is het al op de markt?

Ontsluitingsmethode
Dit is de methode om de positie te verkrijgen, voor sommige technieken dienen de netwerk operatoren benaderd te worden. Andere methoden zijn dan weer onafhankelijk van de netwerk operatoren, en de positie kan aan de kant van de cliënt verkregen worden (cliënt-sided) .

In de ideale situatie zou de beschikbaarheid wereldwijd en zowel buiten en binnen moeten zijn, waar ter wereld kan dan de dienst *Lisa* worden benut. Voor de nauwkeurigheid wordt het volgende argument gehanteerd.

De dichtheid van bestrating in steden wordt uitgedrukt in de zogenaamde omgevingsadressendichtheid, dit is het aantal adressen per vierkante kilometer [W9]. Volgens de statistiek ligt het gemiddelde aantal adressen per vierkante kilometer in een gemiddelde stad op 2.000 adressen/km^2 [W9]. Per adres is het dus 500 m^2, ongeveer 23 meter bij 23 meter. Aangenomen wordt dat er per adres niet meer dan 2 locaties in een route zitten, dus om de 10 meter in het geval dat per adres 2 locaties worden opgenomen in de route. De nauwkeurigheid (foutmarge) mag dan maximaal 11,5 (12) meter bedragen, voor een gedegen *LB* applicatie.

Vanuit onderzoek naar gebruikersvriendelijkheid bij diensten blijkt dat de wachttijd niet langer mag bedragen dan 15 seconden, anders verdwijnt de aandacht van de eindgebruiker. De ideale tijdsduur ligt dan bij minder dan 15 seconden, en de positieberekening moet real-time gebeuren.

Ideaal is dat de positie zo onafhankelijk mogelijk wordt bepaald, geheel aan de kant van de cliënt. De voorkeur ligt dan bij de triangulatie techniek, wat in de meeste gevallen cliënt-sided is.

De positiebepaling moet altijd betrouwbaar zijn, dat wil zeggen dat er bijvoorbeeld geen instanties zijn die invloed kunnen hebben op de werking van de lokalisatie techniek.

De voorkeur gaat uit naar open source componenten, ze zijn immers gratis en de interne code kan altijd naar believen aangepast worden.

Voor sommige lokalisatie technieken zijn extra hardware benodigd, deze moet dan ook verkrijgbaar zijn voor het in gebruik nemen van *Lisa*. Het gebruiksgemak moet ook hoog zitten.

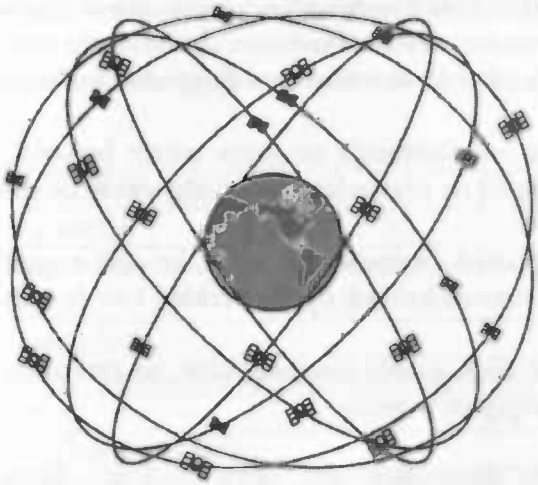
Voor de ontsluitingsmethode geldt dat de positie onafhankelijk verkregen kan worden, dit hangt een beetje samen met Techniek.

De ideale situatie zou dan zijn:

Beschikbaarheid:	Wereldwijd, buiten, binnen.
Nauwkeurigheid:	<12 meter.
Tijdsduur:	Real-time, < 15 seconden.
Techniek:	Triangulatie.
Betrouwbaarheid:	Hoog.
Software:	Open source (C)OTS beschikbaar.
Hardware:	Makkelijk te verkrijgen en makkelijk in gebruik.
Ontsluitingsmethode:	Onafhankelijk van netwerk.

2.2. Global Positioning System (GPS)

Het Global Positioning System is door de Verenigde Staten in 1978 gelanceerd, de eerste GPS satelliet is in dat jaar in de lucht gegaan. De meest recente satelliet die is gelanceerd was op 6 november 2004. GPS is een satelliet navigatie systeem om de exacte locatie van een object, dat in staat is het GPS signaal te ontvangen, te bepalen. Daarnaast biedt GPS ook een zeer nauwkeurige tijdsreferentie, waar dan ook ter wereld.



Figuur 2.1: GPS satellieten constellatie.

Hieronder worden de elementen waar het GPS systeem uit bestaat, toegelicht. Daarna, aan de hand van de selectiecriteria, diens eigenschappen in een overzicht.

2.2.1. Elementen van GPS

GPS bestaat uit drie elementen [A4]; 1. het ruimtelijke segment, 2. het gebruikers segment en 3. het control segment. Het ruimtelijke segment bestaat uit de satellieten zelf, in de omloop rondom Aarde. Het gebruikerssegment bestaat uit de devices die het GPS signaal kunnen ontvangen. Het control segment bestaat uit grondstations die ervoor zorgen dat de satellieten werkend zijn.

Ruimtelijk segment

Het ruimtelijke segment bestaat uit een constellatie van 24 satellieten en een paar reserve. Elke satelliet is gepositioneerd, 20.200 kilometer hoog, in diens eigen omloop rondom Aarde. Elke satelliet draait twee keer per dag om de Aarde, elk satelliet doet er dus 12 uur over. De satellieten bevatten elk een atoomklok, de tijd van deze klok wordt constant broadcast verzonden. Naast de tijd wordt ook de aanvullende informatie verzonden, hieronder wordt onder meer verstaan de huidige positie, de bewegingsrichting, de snelheid. Deze informatie wordt ontvangen door de grondstations. Aan de hand van deze klokken worden de intervallen (de tijd dat het signaal de weg aflegt) berekend. Het signaal gaat met de snelheid van het licht naar de Aarde, ondanks deze gigantische snelheid kost het toch een paar nanoseconden om de weg af te leggen. Dit is echter voldoende om de afstand tussen de stations op Aarde en de satellieten te berekenen. De receiver verkrijgt dus de benodigde latitude, longitude en altitude via het doorberekenen van de afstanden tot vier afzonderlijke GPS satellieten. Voor de exacte (wiskundige) beschrijving hiervan, zie de bijlage III en/of [L6].

Gebruikers segment
Het gebruikers segment bestaat uit de ontvanger die in de hand ligt, of die in een voertuig/device is gemonteerd, bijvoorbeeld een PDA of een vliegtuig. Deze ontvangers zijn in staat de GPS signaal te ontvangen, te decoderen en te verwerken. Er zijn honderden verschillende modellen, de typische handheld heeft het formaat van een GSM mobiele telefoon. Echter de laatste ontwikkelingen hebben tot nog kleinere modellen geleid.

Control segment
De control segment is gestationeerd op de Aarde, het bestaat uit meerdere grondstations:

- Een 'master' station in Schriever Air Force Base in Colorado
- Vijf geautomatiseerde monitor stations: Hawaï en Kwajalein in de Pacifische Oceaan; Diego Garcia in de Indische Oceaan; Ascension Island in de Atlantische Oceaan; en Colorado Springs in Colorado.
- Vier grote antenne grondstations welke rechtstreeks contact hebben met de satellieten om commando's en updates van data te verzenden, en om informatie over de status van de satellieten te ontvangen.

Beschikbaarheid:	Wereldwijd, buiten.
Nauwkeurigheid:	3-10 meter.
Tijdsduur:	Constellatie zoektijd maximaal 30 seconden. Positiebepaling: real-time.
Techniek:	Triangulatie.
Betrouwbaarheid:	Hoog. Echter defensie van de VS heeft invloed op betrouwbaarheid van GPS.
Software:	Diverse COTS beschikbaar, geen open source.
Hardware:	GPS ontvangers zijn makkelijk te verkrijgen en makkelijk in gebruik op verschillende manieren: insteekkaarten, Bluetooth.
Ontsluitingsmethode:	Onafhankelijk, mits Defensie van de VS geen maatregelen genomen heeft.

2.2.2. Galileo, de Europese GPS variant

Hoewel *Galileo* nog niet in de lucht is, is het toch de moeite waard hier kort iets over te vermelden.

De Verenigde Staten (VS) hebben in 2000 besloten dat Defensie de bevoegdheid krijgt om de defensieve en civiele gedeelten van GPS te kunnen uitschakelen. De soevereiniteit van het GPS systeem ligt dus in de VS. Europa maakte hier zich zorgen over, dit heeft, naast enkele commerciële redenen, geleid tot de totstandkoming van het *Galileo*-project. De bedoeling is dat *Galileo* in 2008 operationeel is, en is (voor de publieke service niveau) geheel beschikbaar voor de wereld. *Galileo* en GPS zijn gezamenlijk interoperationeel en zijn ook compatibel met elkaar. Verder heeft *Galileo* de zogenaamde '*service-levels*', dit houdt in dat er commerciële service (het gebruik van het systeem wordt in rekening gebracht), publieke service (publiekelijk beschikbaar) niveaus zijn.

Hoewel beide systemen gestoeld zijn op dezelfde wiskundige principes, zijn er toch enkele noemenswaardige verschillen:

- In plaats van 24 satellieten bij GPS, heeft *Galileo* 30 satellieten. Deze zijn ongeveer 24.000 km. hoog gepositioneerd.

- Performance van een GPS/*Galileo* combinatie neemt toe met maximaal 500% ten opzichte van GPS [A3].
- De statusinformatie van de *Galileo* satellieten wordt ook verzonden, dit is cruciaal voor applicaties waar veiligheid (*safety*) van belang is, bijvoorbeeld bij rijdende treinen, bij landende vliegtuigen, etcetera [A5].
- De ontvangers zijn ook in staat positie-informatie te ontvangen van de grondstations, dit vergroot de accuratesse [A6].
- Voor de civiele sector biedt *Galileo* een nauwkeurigheid tot 1 meter, voor de commerciële sector kan *Galileo* een nauwkeurigheid tot 0,10 meter realiseren[A6].

Voor verdere informatie over de *Galileo*-project wordt verwezen naar [A6].

Beschikbaarheid:	Pas in 2008. Alleen buiten.
Nauwkeurigheid:	< 2 meter.
Tijdsduur:	Constellatie zoektijd korter dan 15 seconden. Positiebepaling: real-time.
Techniek:	Triangulatie.
Betrouwbaarheid:	Hoog. Vanuit Europees civiele oogpunten opgezet.
Software:	Geen.
Hardware:	Niet beschikbaar.
Ontsluitingsmethode:	Onafhankelijk van netwerk, cliënt-sided.

2.3. 2G/3G netwerken positionering

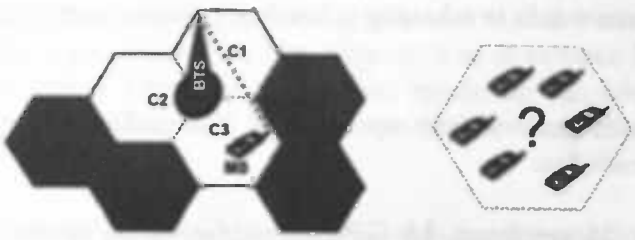
Momenteel zijn er drie technieken voor positionering binnen 2G/3G netwerken:

1. De *cell-id* methode
2. *Radio-based* lokalisatie
3. *Network Assisted GPS* (AGPS)

Ze zullen stuk voor stuk worden toegelicht in de volgende paragrafen. Alle beschreven methoden zijn ook van toepassing bij de UMTS en de GPRS (3G) netwerken, zie [A1] en [A7].

2.3.1. De cell-id methode

De *cell-id* methode is de meest simpele manier om de locatie van een mobiele device te bepalen, het vereist geen speciale apparatuur. De GSM toestellen zijn in staat wereldwijd de *cell-id* op te vragen in 3 seconden. De *cell-id* is de identificatie code van de cel in het GSM netwerk, dus de code van de 'zendmasten'.



Figuur 2.3: De *cell-id* van de mast (BTS), MS bevindt zich dus in cel C3, echter de exacte positie van de device in de cel blijft onbekend.

De nauwkeurigheid van deze methode is dus afhankelijk van de grootte van de cel. In stedelijke gebieden kan de accuratesse oplopen tot minimaal 150 m. Echter dit neemt dramatisch af als de cellen groter worden. Om die redenen wordt in de praktijk deze lokalisatiemethode nauwelijks benut [A9].

Overigens kunnen alleen mobiele toestellen met Symbian OS cliënt-sided de *cell-id* van de zendmasten verkrijgen, en doorgeven aan 'externe' applicaties op dat device. De *cell-id* kan ook via de netwerk operatoren verkregen worden, dit heeft echter de nadeel dat het veel overhead vergt, en daardoor meer tijd kost.

Beschikbaarheid:	Alleen voor operatoren en Symbian toestellen. Werkt zowel buiten als binnen.
Nauwkeurigheid:	Slecht. Minimaal de celgrootte.
Tijdsduur:	Real-time.
Techniek:	Station-based, dus de exacte positie in cel onbekend.
Betrouwbaarheid:	Laag. Exacte positie blijft onbekend.
Software:	Symbian OS.
Hardware:	Toestellen met Symbian OS.
Ontsluitingsmethode:	Beide. Netwerk operatoren kunnen benaderd worden om de positie te ontvangen, en met Symbian toestellen kan onafhankelijk de *cell-id* worden opgevraagd.

2.3.2. Radio-based lokalisatie

Voor de '*radio-based*' lokalisatie technieken zijn devices die radio signalen verzenden benodigd, de signalen worden ontvangen door de mobiele ontvanger. In het geval van 2G/3G netwerken gaat het om het signaalverkeer tussen de masten en de mobiele device.

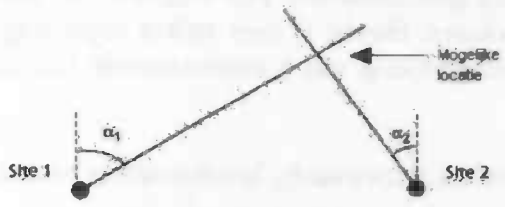
Er zijn drie gangbare *radio-based* lokalisatie methoden [A7]:

1. *Angle of Arrival* (AoA).
2. *Time of Arrival* (ToA).
3. *Time Difference of Arrival* (TDoA).

Deze drie methoden zijn allen afhankelijk van de radiogolven. Ondanks dat deze principes gestoeld zijn op radiogolven, kunnen ze ook toegepast worden in 2G/3G netwerken. Daarom zullen ze nader worden toegelicht.

Angle of Arrival (AoA)

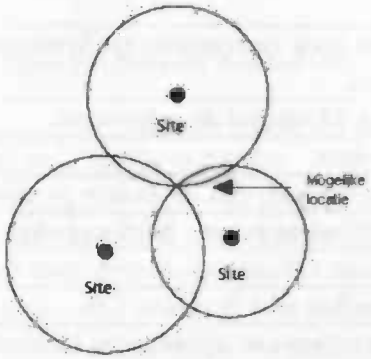
De AoA techniek bepaalt de positie van de mobiele device aan de hand van triangulatie. Zie figuur 2.4.



Figuur 2.4: Angle of Arrival

Het snijpunt van de twee lijnen, geformeerd uit de afstand tussen de twee zendmasten en de twee hoeken, α_1 en α_2, is de mogelijke locatie van de mobiele device. Hierbij moeten minimaal twee zendmasten in de buurt zijn, het gebruik van meerdere masten is mogelijk. Omdat (mobiele) antennes benodigd zijn voor deze methode, is het lastig om AoA te realiseren op een mobiele telefoon.

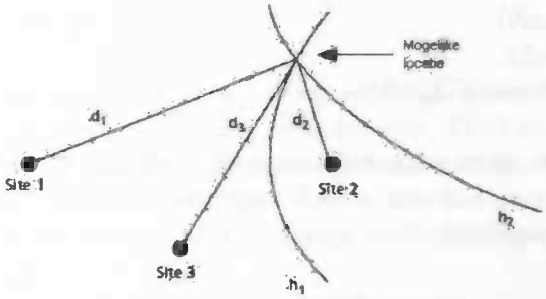
Time of Arrival (ToA)
Het principe van ToA lijkt op die van GPS, zie figuur 2.5.



Figuur 2.5: Time of Arrival

De mogelijke positie ligt dus bij het snijpunt van de drie cirkels. De propagatie tijd van de radiogolf is proportioneel met diens afstand, dit vermenigvuldigen met de lichtsnelheid heeft de afstand als resultaat, zie (2.1). Twee zendmasten levert een grijs gebied, wat niet exact is. Dit is niet het geval bij 3 zendmasten.

Time Difference of Arrival (TDoA)
In plaats van triangulatie wordt triliteratie gebruikt, het principe is vrijwel identiek, de zendmasten worden eveneens als referentiepunten gehanteerd. Zie figuur 2.6.



Figuur 2.6: Time Difference of Arrival

Bij ToA worden absolute tijdsmetingen gebruikt, bij TDoA enkel het tijdsverschil. Het tijdsverschil wordt geconverteerd in een constant afstandsverschil tussen twee zendmasten, hieruit wordt een hyperbool geconstrueerd. Het snijpunt van deze twee hyperbolen is dan de mogelijke positie van het object. Hierbij is zeer strikte regulering van alle tijdsklokken van de zendmasten vereist, een tijdsafwijking van 1 microseconde (µs) heeft al een afwijking van 300 meter tot gevolg.

Deze drie methoden worden als gelijkwaardig beschouwd in het schema op de volgende pagina.

Beschikbaarheid:	Operators. Werkt zowel buiten als binnen.
Nauwkeurigheid:	60-200 meter.
Tijdsduur:	Real-time.
Techniek:	Triangulatie.
Betrouwbaarheid:	Gemiddeld. Is van vele factoren afhankelijk. Nauwkeurigheid van tijdsmeting grote rol (TDoA).
Software:	Geen.
Hardware:	Geen.
Ontsluitingsmethode:	De operatoren kunnen aan de hand van de signaalsterkte van ontvangst de positie berekenen.

2.3.3. Network Assisted GPS (AGPS)

AGPS is een combinatie van GPS en de 2G/3G netwerken. AGPS bestaat uit de volgende componenten:

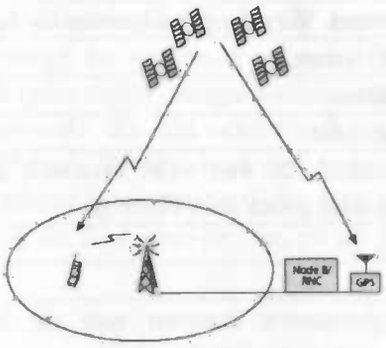
1. De draadloze handset met de partiele GPS ontvanger.
2. AGPS server met referentie GPS ontvanger.
3. Een draadloos netwerk infrastructuur dat de handset met de server verbindt.

Deze methode maakt dus gebruik van het 2G/3G netwerk om de referentie GPS signaal te kunnen verzenden/ontvangen. De GPS server assisteert als het ware de partiele GPS ontvanger, zo worden de volgende voordelen bereikt:

- Starttijd is sterk gereduceerd, aangezien de satellieten niet geacquisitieerd hoeft te worden.
- Positie nauwkeurig sterk toegenomen; het helpt de zwakke signalen te detecteren als men bijvoorbeeld binnenshuis bent.
- Energiebesparing van de GPS ontvanger.

Het basisidee van AGPS is dat het GPS referentienetwerk alle satellieten kunnen traceren, er zijn bij dat netwerk geen belemmeringen (gebouwen, bomen, e.d.). Dat netwerk is ook verbonden met het 2G/3G netwerk. De GPS netwerk houdt voortdurend de status van de constellatie van de GPS satellieten in de gaten, en levert data (positie van de handsets, zichtbaarheid van de satellieten, klok correcties, etc.). Zodra de request van de handset is verzonden, stuurt de AGPS netwerk de afgeleide positie informatie terug. De acquisitie tijd is sterk verminderd omdat de GPS ontvanger de Doppler vs. Code fase niet hoeft te berekenen, zie [L6], omdat het reeds bekend is in welke gedeelte van de lucht/ruimte de GPS ontvanger van de handset naar satellieten moet zoeken. Zie figuur 2.7 op de volgende pagina.

Momenteel zijn er zeer weinig AGPS devices op de (Nederlandse) markt, ook zijn de 2G/3G netwerken in Nederland nog niet aangepast voor AGPS.



Figuur 2.7: Network Assisted GPS

Beschikbaarheid:	Alleen in de VS. Alleen buiten.
Nauwkeurigheid:	2-7 meter
Tijdsduur:	Constellatie zoektijd veel korter dan GPS, wordt voor rekening genomen door grondstations.
Techniek:	Triangulatie.
Betrouwbaarheid:	Gemiddeld. Is afhankelijk van grondstations.
Software:	Wordt geleverd bij AGPS ontvangers.
Hardware:	AGPS ontvangers zijn in de VS verkrijgbaar.
Ontsluitingsmethode:	Met behulp van de grondstations wordt de GPS positie doorgegeven aan de ontvanger, hierdoor is er geen sprake van constellatietijd voor de ontvanger

2.3.4. Wireless Fidelity (WiFi)/Wide Local Area Network (WLAN)

Een WiFi netwerk bestaat uit WiFi hotspots, deze zenden het WiFi signaal (IEEE 802.11b), en vormen zo het netwerk waar mensen draadloos via verscheidene protocollen kunnen communiceren tussen verschillende partijen. Een WiFi lokalisatie techniek heeft voordelen ten opzichte van andere technieken:

- De WiFi netwerken bestaan al, en zijn zo goed als gangbaar en overal.
- Het is een softwarematige oplossing, dus laptops, PDA's, andere mobiele devices die WiFi kunnen, kunnen van deze techniek gebruik maken. Hardware aanpassingen zijn dus niet benodigd.
- Het heeft een groter bereik, met mindere middelen, vergeleken met andere indoor lokalisatie technieken.
- Het is zeer stabiel, vanwege het robuuste WiFi signaal

De positie van een object in een WiFi omgeving wordt bepaald aan de hand van de sterkte van het WLAN signaal. Aan de hand van de positionering algoritme [A11] wordt de locatie van het object bepaald. Bijvoorbeeld *Ekahau* maakt gebruik van deze techniek [W8]. Hierbij wordt gebruik gemaakt van de statistiek (kansberekening). Naast positionering door middel van signaalsterkte is ook triangulatie mogelijk, zie daarover paragraaf 2.3.2.

Weer is bij de WiFi lokalisatie techniek sprake van dualiteit; -station-based en - cliënt-based. De positie kan worden berekend aan de hand van het genoemde algoritme bij de kant van de cliënt. Daarnaast kan ook door middel van de identificatiecode van de hotspot, de hotspot verplaatst

niet. Als een device zich in de range van een hotspot bevindt, dan is de zone ook bekend, en daarmee ook de positie van de device. Een voorbeeld dat van dit principe gebruik maakt is de 'radius-server' [W10]. Voor de technische werking van de WiFi netwerken wordt verwezen naar [L11].

Voor het algoritme om de locatie te bepalen aan de hand van de signaalsterkte, zie appendix IV.

Beschikbaarheid:	Alleen binnenshuis waar WiFi netwerken aanwezig zijn.
Nauwkeurigheid:	Tientallen meter (hotspot range).
Tijdsduur:	Real-time.
Techniek:	Triangulatie en station-based.
Betrouwbaarheid:	Gemiddeld. Is afhankelijk van hotspots.
Software:	Bijvoorbeeld: Windows WMI, NDIS 5.0, Ekahau.
Hardware:	Gangbaar, WLAN kaarten zijn voldoende.
Ontsluitingsmethode:	De positie van de hotspot is bekend verondersteld, en is daarmee ook de locatie van de ontvanger in het hotspot gebied.

2.3.5. Radio Frequency Identification (RFID)

RFID is een techniek wat tegenwoordig (anno 2005) sterk in opmars is, de voorspelling is dat het vooral gebruikt zal worden in de logistieke sector om de productstromen in kaart te brengen. De voorganger van RFID is de barcode techniek, deze heeft echter nogal wat nadelen:

- De streepjescode moet altijd zichtbaar aan de buitenkant van het object zitten.
- De streepjescode kan niet hergeprogrammeerd worden.
- Beperkte (geen) storage level.

Een goed alternatief hiervoor is de zogenaamde '*smartcard*'. Echter, om de gegevens op de smartcard te raadplegen moet er (fysiek) contact zijn met de '*reader*'. De uitkomst is RFID, "*the contactless technology*".

Eigenlijk is RFID geen lokalisatie techniek, aangezien het puur om identificatie van objecten gaat. Echter, met *Lisa* kan RFID wel een lokalisatie rol hebben. Daarover wordt meer omschreven in (2.4.2).

Elementen van RFID

Een RFID systeem bestaat altijd uit twee componenten:

- De '*transponder*', ook wel de '*tag*' genoemd, dit bevindt zich op de te identificeren object.
- De '*reader*', het hangt af of deze de informatie op de transponder kan lezen/wijzigen.

De reader bestaat uit een radiofrequentie module (de transmitter en de receiver), een control unit en een koppelstuk die de signalen van de tag ontvangt. Ook wordt er vaak een interface (RS232, RS485, etc.) toegevoegd, om de data te kunnen doorsturen naar een ander systeem (PC, robot controle systeem, etc.).

De tag bevat de data over het object waarbij de tag hoort. Dit bestaat eveneens uit een koppelstuk die de signalen ontvangt en verstuurt en een elektronische microchip. De tag is passief, dat wil zeggen, het heeft geen eigen energievoorziening. De energie wordt verkregen uit de radiogolf die de reader veroorzaakt. Er zijn echter ook actieve tags, deze zijn wat groter in omvang, en hebben een batterij/accu. Ze zijn vaak ook veel duurder dan de passieve tags. De tag moet dus zich in het golf-gebied bevinden, wil de data van de tag ingelezen worden. De reader verstuurt de tijdssignaal en de flux naar de tag, en zo stuurt de tag de tijdssignaal en de data op de microchip terug.

RFID als lokalisatie techniek

Een van de aspecten van *LBS* is dat de applicatie weet dat de gebruiker in de buurt is van een bepaalde object/locatie, zodat de applicatie de juiste acties kan ondernemen. Met andere woorden, de applicatie weet wat het doen moet, zodra het object geïdentificeerd is. Dit kan op twee manieren:

1. De applicatie heeft de beschikking over de reader:
   De applicatie (gebruiker) identificeert de RFID object met de reader, de applicatie haalt uit de database om welk object het gaat en de bijbehorende informatie.
2. De applicatie heeft de beschikking over een eigen RFID-tag:
   Bij alle locaties/objecten is een reader gemonteerd, deze staat in verbinding met de server. Als een RFID tag in de buurt is, dit wordt dan gedetecteerd door de reader. De reader stuurt de data door naar de server, en de server stuurt de informatie van de object, die bij de reader hoort, naar de applicatie (gebruiker).

Aangezien het vrij prijzig is om overal readers neer te zetten, is er voor optie 1 gekozen.

Voor verdere (technische) details over RFID wordt verwezen naar [L5].

Beschikbaarheid:	Technologie is beschikbaar, echter in de praktijk nog niet overal toegepast.
Nauwkeurigheid:	Exact.
Tijdsduur:	Scantijd zeer kort; minder dan 1 seconde.
Techniek:	Radiogolven (station-based).
Betrouwbaarheid:	Hoog.
Software:	SDK.
Hardware:	Stand-alone readers verkrijgbaar, diverse uitvoeringen niet 100% verkrijgbaar.
Ontsluitingsmethode:	De positie van de reader/tag is bekend verondersteld, en is daarmee ook de locatie van de ontvanger.

2.4. Scanning techniek

Selectiecriteria
Voor de keuze van een scanning techniek worden de selectiecriteria bepaald.

De selectiecriteria zijn:
- Op de markt
- Gebruiksgemak
- Range (centimeters)
- Software (COTS: ja of nee)

De genoemde selectiecriteria zullen hieronder stuk voor stuk worden behandeld. Ze zijn opgesteld naar aanleiding van de beschikbare technieken die naar voren kwamen tijdens het literatuuronderzoek.

Op de markt
Sommige scanning technieken zijn vrij nieuw, en de scan-devices zijn daarom misschien (in mindere mate) beschikbaar op de markt.

Gebruiksgemak
Voor sommige scanning technieken zijn bepaalde handelingen verricht. Beoordeeld wordt in hoeverre deze handelingen omslachtig zijn.

Range
Om de objecten/tags/code te scannen moet de scanner zich binnen een bepaalde range bevinden. Maar ook weer niet te dichtbij, vanaf 50 cm is aardig *contactless*.

Software
Zijn er COTS componenten beschikbaar? Zijn er software development kits (SDK's) beschikbaar?

In het ideale geval zou de techniek op de markt moeten zijn, de verkrijging van apparatuur moet eenvoudig zijn. Om objecten te scannen moet er geen speciale handelingen verricht worden, in het minimale geval dient de scanner in de range gebracht te worden. Om het gebruiksgemak te vergroten moet de range minimaal 50 centimeter en maximaal 100 centimeter zijn.

De scanning techniek moet zonder obstakels te integreren zijn in de dienst, een SDK is daarom verreist.

De ideale situatie zou dan zijn:

Op de markt:	Ja, en makkelijk te verkrijgen.
Gebruiksgemak:	Alleen het brengen van de scanner binnen de range.
Range:	> 50 cm en < 100 cm.
Software:	SDK beschikbaar.

In de volgende paragrafen zullen de volgende technieken worden behandeld:

1. Radio-Frequency Identification (RFID).
2. Barcode scanning.

3. Smartcard scanning.
4. Infrarood scanning.

2.4.1. Radio-Frequency Identification (RFID)

Voor de werking hiervan wordt verwezen naar paragraaf 2.3.5. RFID is ook te gebruiken als scanning techniek. Er zijn momenteel diverse devices op de markt. TNO ICT heeft een RFID reader in bezit, deze kan via de seriële poort met de PC worden gekoppeld. Deze scanner kan RFID tags lezen. Naast deze scanner zijn er ook compact flash readers op de markt, echter deze zijn moeizaam te verkrijgen. Bij de CF readers worden ook SDK's geleverd.

De tag mag maximaal 6 centimeter in de buurt zijn van de reader, dit betreft de passieve tags. De actieve tags hebben een ruimere range, en zijn ook groter in omvang en ook qua prijs.

Op de markt:	Ja, maar moeilijk verkrijgbaar.
Gebruiksgemak:	Alleen het brengen van de scanner binnen de range.
Range:	< 6 cm.
Software:	SDK beschikbaar.

2.4.2. Barcode scanning

Deze techniek bestaat al vrij lang, en wordt voornamelijk in de logistiek gebruikt. Echter er zijn een paar nadelen hieraan, zie paragraaf 2.3.5.

Op de markt:	Ja, en makkelijk te verkrijgen.
Gebruiksgemak:	Met de infrarode reader over de barcode slepen.
Range:	< 0 cm.
Software:	SDK beschikbaar.

2.4.3. Smartcard scanning

Ook deze techniek bestaat al vrij lang, en wordt meestal gebruikt bij bankpassen. De smartcards moeten geconfigueerd worden, dit vermindert het gebruiksgemak. En bovendien zijn smartcards niet gangbaar als scanning techniek.

Op de markt:	Ja, en makkelijk te verkrijgen.
Gebruiksgemak:	Het in contact brengen van de smartcard en de reader.
Range:	< 0 cm.
Software:	SDK beschikbaar.

2.4.4. Infrarood scanning

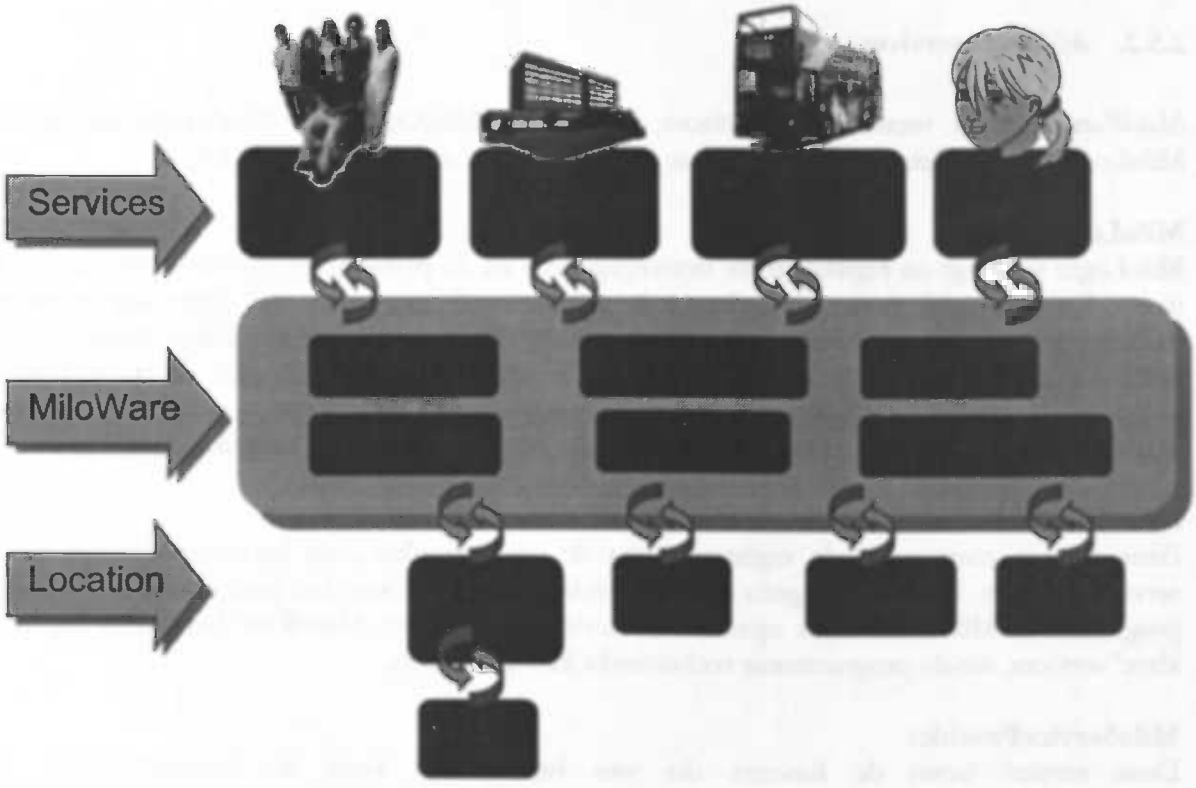
Deze techniek is in mindere mate gangbaar dan de barcode, want het vereist een speciale handeling: de ontvanger moet analoog gelegd worden met de zender.

Op de markt:	Ja, en makkelijk te verkrijgen.
Gebruiksgemak:	Analoog leggen met de zender.
Range:	< 7 cm.
Software:	SDK beschikbaar.

2.5. Location Based Platform: *MiloWare*

Een eis vanuit TNO ICT is dat de te ontwikkelen applicatie gebruik maakt van het *Location Based* platform *MiloWare*. *MiloWare* vormt daarom een belangrijk onderdeel van het gehele systeem. In deze paragraaf wordt verder ingegaan op wat *MiloWare* inhoudt.

De term *Milo* is afgeleid uit 'my location', *MiloWare* is door TNO ICT ontwikkeld voor klanten die *LB* diensten willen gaan aanbieden/ontwikkelen. Deze diensten kunnen dan draaien op het *MiloWare* platform. *MiloWare* bestaat uit bouwblokken, waarmee gemakkelijk een variëteit aan locatiediensten gebouwd en aangeboden kan worden. De centrale spil in *MiloWare* is het *Milo* platform, dit platform ontvangt en beheert ruwe locatiegegevens. Deze locatiegegevens kunnen afkomstig zijn van een netwerk, zoals WiFi of GPRS, maar ook van een mobiele device, zoals een GPS apparaat. Hieronder is een model welke het idee van *MiloWare* weergeeft (Figuur 2.8).



Figuur 2.8: Model van *MiloWare*

MiloWare bestaat dus uit drie lagen; - Location, - MiloWare en –Services. De Location laag ontsluit de positie van de devices die gebruik maken van *MiloWare*. Dit kan op verschillende manieren; via GPS, via het WiFi netwerk, via 2G/3G netwerken, etc. De MiloWare laag bevat 'slimme' rules, die in de gaten houdt waar de devices zich bevinden, en waar ze in buurt van wat

zijn. In deze laag worden ook de 'privacy-rechten' bijgehouden, sommige gebruikers willen niet dat hun locatie-informatie aan derden wordt doorgegeven.

Wat de dienst *Lisa* betreft, omdat GPS als lokalisatie techniek wordt gebruikt, wordt dit via de dienst doorgegeven aan de MiloWare laag, en niet rechtstreeks.

De ruwe locatie-informatie heeft relatief weinig waarde. Het *Milo* platform interpreteert en combineert daarom deze gegevens tot waardevolle en nuttige locatie-informatie.

2.5.1. Architectuur

Het meeste relevante onderdeel van *MiloWare* is het webservice gedeelte, via het SOAP protocol, zie (4.5.2), dit is als het ware de interface naar de externe *LB* applicaties. De werkelijke (ruwe) locatie-informatie wordt opgeslagen in de database van *MiloWare*.

Privacy is voor de eindgebruiker uiteraard essentieel, de eindgebruiker wil graag zelf bepalen in hoeverre de locatiegegevens beschikbaar worden gesteld aan dienstaanbieders. Dit wordt eveneens door *MiloWare* verzorgd.

Bij *MiloWare* wordt onderscheid gemaakt tussen de zogenaamde *LocationProvider* en de *ServiceProvider*. De ServiceProvider is dus de ontwikkelaar die *LB* diensten ontwikkelt en aanbiedt. De LocationProvider is kan bijvoorbeeld de netwerkoperator zijn (Surfnet).

2.5.2. *Milo* webservices

MiloWare heeft in totaal vier interfaces; MiloLogin, MiloOperator, MiloServiceProvider en MiloLocationProvider. Voor de gebruikte functies in het *Lisa*-project, zie (4.5.2).

MiloLogin
MiloLogin verzorgt en registreert de devicegegevens en de privacygegevens (in *Milo*-termen; de trust). De (mobiele) devices worden ook geregistreerd, aangezien een gebruiker meerdere devices in bezit kan hebben, bijvoorbeeld een PDA en een laptop. Beide hanteren dan verschillende lokalisatietechnieken. Er zijn drie trust niveaus: - miloPrivate, de trusts kunnen alleen een ID zien; - miloProtected, de trusts kunnen alle informatie zien; - miloPublic, iedereen kan alles zien. Deze interface kan worden gebruikt om de registratie te vergemakkelijken.

MiloOperator
Deze service zorgt voor de registratie van de eindgebruikers, de locationproviders en de serviceproviders. Echter dit is geen relevante webservice, aangezien het geen library is waarbij de programmeur MiloWare moet opstarten, resetten of afsluiten. MiloWare bevat dus 'kant-en-klare' services, die de programmeur rechtstreeks kan aanroepen.

MiloServiceProvider
Deze service bevat de functies die van belang zijn voor de ServiceProvider. De MiloServiceProvider service verzorgt alles wat te maken heeft met de (geregistreerde) absolute locatie-gegevens. Hieronder wordt o.a. verstaan het vinden van objecten in de buurt, het ophalen van de laatste positie, checken of een object zich op een gegeven locatie bevindt. In deze service zijn er twee soorten functies: subscriptions en niet-subscriptions. Bij de eerste wordt er niet direct respons gegeven, pas als er wordt voldaan aan de gegeven event (bijvoorbeeld de XnearYsubscription functie, als X vlakbij Y is, dan respons). En bij de andere soort, wordt er direct respons gegeven. Deze interface is als het ware de spil voor de service provider ontwikkelaar, hierin zitten functies die het *LB* aspect mogelijk maken.

MiloLocationProvider
In deze service zitten de functies die de positiegegevens van de devices verzorgt en doorgeeft aan de MiloWare laag. Ook de locaties kunnen via deze functies worden opgeslagen/updated. In *Milo*-termen worden de locaties '*zones*' genoemd. Dit is vooral handig voor de cell-id en de WiFi lokalisatie technieken, aangezien deze technieken niet met absolute posities maar met gebieden (zones) werken. Overigens de MiloLocationProvider webservice is enkel van belang voor de location providers, bijvoorbeeld de netwerk-opeators. Maar hier horen dus ook mensen met een client, die via GPS locatiegegevens verkrijgt, bij.

2.6. Conclusie

In hoofdstuk 2 zijn de lokalisatietechnieken uitgebreid behandeld, naar aanleiding van de voordelen en de nadelen van de technieken wordt een overzicht van al de technieken en de conclusie, de keuze voor een techniek, in deze paragraaf weergegeven.

Overzicht
Op de volgende pagina is een overzicht gepresenteerd van alle lokalisatietechnieken en diens eigenschappen. Omdat de scanningtechniek een kleine rol heeft, is er geen overzicht hiervan in dit verslag opgenomen.

	GPS	Galileo	Cell-id	Radio-based	AGPS	Wifi	RFiD
Beschikbaarheid	Wereldwijd,buiten.	Pas in 2008. Alleen buiten.	Alleen voor operatoren en Symbian toestellen. Werkt zowel buiten asls binnen.	Operators. Werkt zowel buiten als binnen.	Alleen in de VS. Alleen buiten.	Alleen binnenshuis waar WiFi netwerken aanwezig zijn.	Technologie is beschikbaar, echter in de praktijk nog niet overal toegepast.
Nauwkeurigheid	3-10 meter.	<2 meter.	Slecht. Minimaal de cellgrootte.	60-200 meter.	2-7 meter.	Tientallen meter (hotspot range).	Exact.
Tijdsduur:	Constellatie zoektijd maximaal 30 seconden. Positiebepaling: real-time.	Constellatie zoektijd korter dan 15 seconden. Positiebepaling: real-time.	Real-time.	Real-time.	Constellatie zoektijd veel korter dan GPS, wordt voor rekening genomen door grondstations.	Real-time.	Scantijd zeer kort; < 1 seconde.
Techniek	Triangulatie	Triangulatie.	Station-based, dus exacte positie in cel onbekend.	Triangulatie.	Triangulatie.	Triangulatie en station-based.	Radiogolven (station-based)
Betrouwbaarheid	Hoog. Echter defensie van de VS heeft controle over de betrouwbaarheid van GPS.	Hoog. Vanuit Europees civiele oogpunten opgezet.	Laag. Exacte positie blijft onbekend.	Gemiddeld. Is van vele factoren afhankelijk. Nauwkeurigheid van tijdsmeting grote rol (TDoA).	Gemiddeld. Is afhankelijk van grondstations.	Gemiddeld. Is afhankelijk van grondstation.	Hoog.
Software	Diverse COTS beschikbaar, geen open source.	Geen.	Symbian OS.	Geen.	AGPS ontvangers zijn in de VS verkrijgbaar.	Windows WMI, NDIS 5.0, Ékahau.	SDK.
Hardware:	GPS ontvangers zijn makkelijk te verkrijgen en makkelijk in gebruik op verschillende manieren.	Niet beschikbaar.	Toestellen met Symbian OS.	Geen.	Wordt geleverd bij AGPS ontvangers.	Gangbaar, WLAN kaarten zijn voldoende.	Stand-alone readers verkrijgbaar, diverse uitvoeringen niet 100% verkrijgbaar.
Ontsluitings-methode:	Onafhankelijk, mits Defensie van VS geen maatregelen genomen heeft.	Onafhankelijk van netwerk, cliënt-sided.	Beide. Netwerk operatoren kunnen benaderd worden om de positie te ontvangen, en met Symbian toestellen kan onafhankelijk de cell-id worden opgevraagd.	De operatoren kunnen aan de hand van de signaalsterkte van ontvangst de positie berekenen.	Met behulp van de grondstations wordt de GPS positie doorgegeven aan de ontvanger, hierdoor is er geen sprake van constellatietijd voor de ontvanger.	De positie van de hotspot is bekend verondersteld, en is daarmee ook de locatie van de ontvanger in het hotspot gebied.	De positie van de reader/tag is bekend verondersteld, en is daarmee ook de locatie van de ontvanger.

Lokalisatietechniek

Als gekeken wordt naar de ideale situatie van de lokalisatie techniek:

	Ideaal	**GPS**
Beschikbaarheid:	Wereldwijd, buiten, binnen.	Wereldwijd, buiten.
Nauwkeurigheid:	<12 meter.	3-10 meter.
Gemak:	Real-time, <15 seconden.	Constellatie zoektijd maximaal 30 seconden. Positiebepaling: real-time.
Techniek:	Triangulatie.	Triangulatie.
Betrouwbaarheid:	Hoog	Hoog. Echter defensie van de VS heeft invloed op betrouwbaarheid van GPS.
Software:	Open source COTS beschikbaar.	Diverse COTS beschikbaar, geen open source.
Hardware:	Makkelijk te verkrijgen en makkelijk in gebruik.	GPS ontvangers zijn makkelijk te verkrijgen en makkelijk in gebruik op verschillende manieren: insteekkaarten, Bluetooth.
Ontsluitingsmethode:	Onafhankelijk.	Onafhankelijk.

Dan komt GPS het meest overeen met de ideale situatie. Vooral de beschikbaarheid van software en de nauwkeurigheid spelen de belangrijkste rollen. Een nadeel van GPS is dat het niet binnenshuis werkt, daarvoor zou WiFi gebruikt kunnen worden. Omdat de ontwikkelomgeving binnenshuis is, dient een externe antenne aangeschaft te worden. Deze wordt dan buitenshuis geplaatst en verbonden met de GPS ontvanger.

De GPS techniek is gekozen voor de requirement 1: *het moet een lokalisatie techniek bevatten.*

Wegens de beperkte projectduur is besloten enkel GPS als lokalisatie techniek voor *Lisa* te gaan gebruiken. In de toekomst kan gekeken worden naar bijvoorbeeld WiFi lokalisatie.

Scanning techniek

Als gekeken wordt naar de ideale situatie van de scanning techniek:

	Ideaal:	**RFID:**
Op de markt:	Ja, en makkelijk te verkrijgen.	Ja, maar moeilijk verkrijgbaar.
Gebruiksgemak:	Alleen het brengen van de scanner binnen de range.	Alleen het brengen van de scanner binnen de range.
Range:	> 50 cm en < 100 cm.	< 6 cm.
Software:	SDK beschikbaar.	SDK beschikbaar.

Dan komt RFID het meest overeen met de ideale situatie. Vooral het gebruiksgemak weegt het zwaarst. Om RFID te gaan gebruiken voor *Lisa* wordt gekozen voor dat de PDA een eigen RFID reader krijgt. De CF readers zijn helaas moeizaam verkrijgbaar op de markt, ze zijn wel besteld. De bestelling is na lang wachten geannuleerd, vanwege de prijs en er waarschijnlijk een andere leverancier is gevonden. Echter, uiteindelijk is er niets meer vernomen van deze leverancier.

Als (tussen) oplossing is gekozen voor de handmatige invoer van een identificatie code dat op de objecten bevestigd is. Dit wordt de *LisaID* genoemd, voor de werking hiervan wordt verwezen naar hoofdstuk 4.

Aangezien het een eis van TNO ICT is, dat *MiloWare* wordt gebruikt, spreekt het voor zich deze dan ook te gaan gebruiken.

Hoofdstuk 3. Mobiele communicatie technieken en protocollen

Inleiding

In dit hoofdstuk worden de functionele requirements 3, 4 en 5 behandeld.

3. De PDA moet draadloos kunnen communiceren met het LBS platform.
4. De informatie moet binnen een acceptabele tijd over en weer worden verzonden.
5. Enige vormen van beveiliging is gewenst.

De dienst *Lisa* moet op een PDA draaien. Momenteel zijn er PDA's op de markt die een sim-kaart kunnen behuizen, echter de keuze is zeer beperkt. Er is dus gekozen voor het koppelen van de PDA aan een mobiele telefoon via Bluetooth.

De communicatie met het LBS platform vindt plaats via het world wide web, en de dienst zal daarom de beschikking moeten hebben over een dataverbinding. In Nederland zijn er twee beschikbare dataverbindingen: UMTS en GPRS. Aan de hand van de vastgestelde selectiecriteria zal de beslissing vallen welke techniek gebruikt zal worden.

Vanwege de beschikbaarheid van UMTS en GPRS is direct al voldaan aan de functionele requirement 3, *de PDA moet draadloos kunnen communiceren met het LBS platform*. In paragraaf 3.1 en 3.2 zullen deze twee genoemde technieken kort worden behandeld.

De functionele requirement 4, *de informatie moet binnen een acceptabele tijd over en weer worden verzonden*, is mede afhankelijk van het communicatieprotocol. Dit zal worden behandeld in paragraaf 3.3. Tijdens de onderzoeksperiode is er onderzoek gedaan naar de volgende protocollen/technieken:

- SOAP webservices
- XMPP protocol (Jabber)
- MSMQ
- Sockets (remoting)

De genoemde communicatieprotocollen/technieken zullen dan onder de loep worden genomen. In deze paragrafen zal de functionele requirement 5, *enige vormen van beveiliging is gewenst*, ook worden opgenomen, omdat het beveiligingsaspect samenhangt met het gebruikte protocol.

Vanwege dat *Lisa* op een PDA moet draaien, en de applicatie in .NET wordt ontwikkeld, zijn de implementatiemogelijkheden beperkt tot de features wat het *compact .NET franework* te bieden heeft. Heel veel features zullen worden toegevoegd in compact .NET versie 2.0, echter dit wordt in de loop van het jaar 2005 uitgebracht.

Tijdens het onderzoek zal hier ook rekening mee gehouden worden.

3.1. General Packet Radio Service (GPRS)

GPRS is eigenlijk een 2.5G network, een soort tussenvorm van UMTS en GSM. Enkele eigenschappen van GPRS [A12], [W11]:

- Snelheid: de theoretische maximale snelheid ligt bij 171.2 kbps, waarbij alle 8 time slots op dezelfde moment worden gebruikt.

- Gemiddelde datasnelheid is ongeveer gelijk aan 56 kbps.
- Verbinding wordt snel tot stand gebracht, geen dial-up.
- GPRS gebruikers zijn 'always online'.
- GPRS is packet switched, de data wordt gesplitst in aparte packets en weer bij elkaar gevoegd bij de ontvanger.
- Netwerk efficiënt: de netwerk resources en de bandbreedte wordt pas benut als er daadwerkelijk data wordt verzonden.
- Features als Internet, FTP, web-browsing, telnet, etc., zijn ook beschikbaar in GPRS netwerken.

GPRS kan voor vele toepassingen gebruikt worden. Op het gebied van communicatie: ineternet access, Internet toepassingen (e-mail, web-browsing, video telefoon, netwerk games, e-commerce). Geluidstoepassingen: hoogkwalitatief spraak, muziek.

Echter er kleven ook nadelen aan GPRS:

- De daadwerkelijke snelheid is vaak veel lager. De netwerkoperatoren laten het niet toe dat de gebruikers de beschikking krijgen over alle timeslots.
- Vertraging van verzending: GPRS packets worden in alle richtingen verzonden om dezelfde einddoel te bereiken.
- De modulatie van GPRS is gebaseerd op de GMSK (Gaussian Minimum Shift Keying) technologie, waardoor hogere bitsnelheden niet gerealiseerd kunnen worden en de transformatie naar 3G netwerken (UMTS) vergt meer tijd en geld.

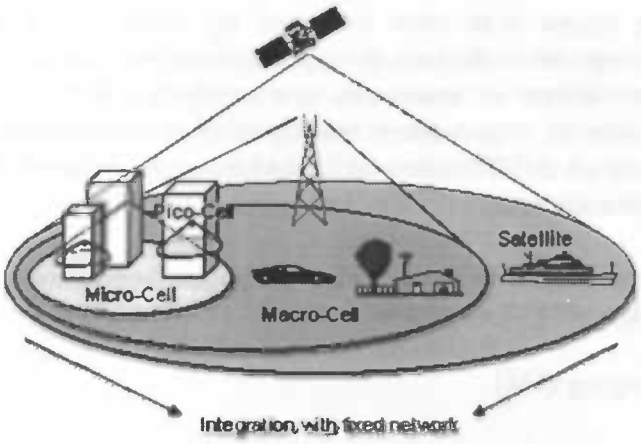
GPRS is al ongeveer 4 jaar de gangbare techniek voor mobiele datacommunicatie, TNO ICT is ook in het bezit van een GPRS simkaart. Voor het *Lisa* project kan GPRS gebruikt worden. Eind 2004 werd UMTS in Nederland geïntroduceerd. In de volgende paragraaf zal UMTS nader worden toegelicht.

3.2. Universal Mobile Telecommunication System (UMTS)

Het UMTS netwerk is in staat om data met hoge bitsnelheden te ontvangen en te verzenden. Het is een evolutie van het bestaande GSM netwerk, specifiek de radio-transmissie methode is nieuw. De bandbreedte van de radiokanalen is bij UMTS veel groter (5 MHz in plaats van 200 KHz in GSM). Hierdoor is UMTS in staat multimediale toepassingen te realiseren. Onder multimediale toepassingen wordt verstaan: de simultane transfer van spraak, data, tekst, plaatjes, audio en video met een maximale datasnelheid van 2 Mbps. Hoewel het GSM netwerk nog wel benut zal worden voor spraak en lage data, omdat GSM goedkoper en gangbaar is.

In figuur 3.1, op de volgende pagina, is een afbeelding dat de hiërarchie van UMTS weergeeft.

De maximale datasnelheid dat UMTS kan leveren ligt op 2 Mbps, echter dit is van toepassing op pico-niveau (zie figuur 3.1). UMTS is in staat realtime multimediale toepassingen te leveren, daarom wordt het asynchronous transfer mode (ATM) als transport protocol gehanteerd. Verder wordt Internet protocol (IP) gebruikt voor het routen van de packet data in het UMTS netwerk.



Figuur 3.1: UMTS hiërarchie structuur.

Vanwege de hoge snelheid zijn er vele multimediale toepassingen mogelijk [W12]:

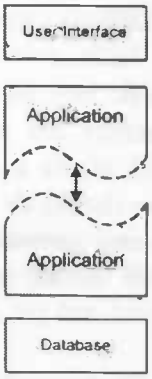
- Videotelefonie.
- Remote control.
- Audio on demand.
- Video clips.
- Virtuele scholen.
- Online bibliotheken.
- Etcetera.

Voor meer informatie over de evolutie van 1G naar 3G netwerken en de technische details van UMTS wordt [W13] zeer aanbevolen.

UMTS is vrij recent in Nederland geïntroduceerd, en is daarom in beginsel vrij prijzig. TNO heeft echter de beschikking over een UMTS simkaart, waardoor ook UMTS gebruikt kan worden voor *Lisa*.

3.3. De communicatieprotocollen

Omdat er met verschillende delen wordt gecommuniceerd, wordt het zogenaamde *client-server* model gehanteerd. De architectuur is (fysiek) *two-tiered*.



Figuur 3.2: two-tiered client-server model.

De communicatielaag tussen deze twee tiers kan op verschillende manieren plaatsvinden. Grofweg kun je communicatie in dat laag, de applicatie laag volgens het OSI model, verdelen in twee groepen, message driven en aanroepen van methodes/objecten. In .NET is voor het aanroepen van methodes de zogenaamde *remoting*-module ontwikkeld, echter dit wordt niet ondersteund in het compact .NET framework. Dan blijven de webservices over en de methodes van message-driven communicatie.

De message-driven methode wordt op de volgende pagina beschreven. De technologie van de webservices zal in paragraaf 3.3.2 worden behandeld.

3.3.1. Instant Messaging (IM)

Er zijn verschillende protocollen mogelijk voor message-driven communicatie, voor de dienst *Lisa* zal voornamelijk gekeken worden naar de *instant-messaging* protocollen. De argumenten hiervoor zijn onder andere:

- Het 'presence' aspect is bij vrijwel alle IM protocollen reeds ontwikkeld. Dit kan een voordeel zijn voor toekomstige features, bijvoorbeeld dat de gebruiker altijd is ingelogd bij de dienst, maar het presence geeft dan aan of de gebruiker gebruik wilt maken van de dienst. Zo zijn er vele features mogelijk.
- Extra functionaliteiten zoals het beheer van 'buddy-lists' zijn reeds ontwikkeld. Dit kan voordelig zijn, voor (toekomstige) interactie tussen meerdere gebruikers.
- Broadcasting van berichten is mogelijk.
- IM biedt de dienst *Lisa* veel uitbreidingsmogelijkheden in de vorm van bijvoorbeeld *chatting* met andere gebruikers, interactieve communicatie, etc.
- IM is zo goed als real-time.
- IM heeft geen invloed op het continueren in het gebruik van applicaties. De applicatie blijft door draaien als er een bericht is verzonden, er hoeft niet gewacht te worden op respons.

Er is een convergentie gaande tussen IM op PC's en IM op mobiele devices. Het toevoegen van de lokalisatie en mobiliteit aan IM geeft het presence aspect een nieuwe waarde. Bijvoorbeeld de gebruikers kunnen van andere gebruikers vernemen of ze beschikbaar zijn, en wat hun locatie is op dat moment.

De telecommunicatie industrie heeft hiertoe hoge verwachtingen, het verwacht dat IM een belangrijke communicatie technologie zal worden. Het voorziet de gebruikers van een persoonlijke communicatie portaal, waar de gebruikers hun voorkeuren kunnen instellen (beschikbaarheid, group-policy, etc.). Presence en IM geïntegreerd in de draadloze wereld zal een impact hebben op onze dagelijkse communicatie [A14].

Microsoft heeft in het .NET framework het message-driven protocol MSMQ (Microsoft Message Queuing) geïmplementeerd, echter dit wordt (wederom) niet ondersteund door compact .NET, en valt daarom ook gelijk af. Er zijn wél IM protocollen die gebruikt kunnen worden in compact .NET. Dat zijn bijvoorbeeld de MSN Messenger, XMPP protocollen. Wat betreft MSN Messenger dient de opmerking gemaakt te worden dat dit protocol niet open is, het is vrij lastig dit te implementeren in andere applicaties (o.a. door middel van reverse-engineering).

Er zijn ongetwijfeld andere technieken en protocollen beschikbaar, echter omdat IM zoveel voordelen heeft, ligt de focus op IM en de webservices.

Het XMPP protocol, ook wel het Jabber protocol genoemd, is een open protocol, en ook open source. Er zijn op Internet veel support en guides aanwezig voor dit protocol. XMPP is ook XML (Extensible Markup Language) gebaseerd, wat een enorm voordeel met zich mee brengt: grote flexibiliteit. Er zijn momenteel ook veel Jabber servers in de lucht, ze verzorgen dan al de gebruikersregistratie en de authenticatie. Dit is ook een belangrijk voordeel.

Daarom wordt enkel XMPP onder de loep genomen. Echter, er bestaat ook het SIMPLE protocol, daarover wordt kort verteld in het XMPP verhaal (subparagraaf 3.3.1). Dus blijven er twee technieken over die geschikt kunnen zijn voor de dienst:

- XMPP protocol (Jabber)
- Webservices

Beide technieken maken gebruik van XML, hierna wordt XML kort toegelicht. Daarna worden de technieken respectievelijk in de subparagrafen 3.3.3 en 3.3.4 behandeld.

3.3.2. XML

XML wordt vaak in één adem genoemd met webservices, maar ook met XMPP. XML is een zeer flexibele tekst formaat, het is afgeleid van SGML. Een ander voorbeeld van een tekst formaat is HTML. Echter XML vervangt echter **niet** HTML. XML is ontworpen om data om te schrijven, en wat de data is. HTML is ontworpen om de data weer te geven, en hoe het er uit ziet. XML structureert enkel data. Een voorbeeld van een notitie:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
     <to>Albert</to>
     <from>TNO</from>
     <heading>Herinnering</heading>
     <body>Vergeet niet dat je morgen een vergadering hebt!</body>
</note>
```

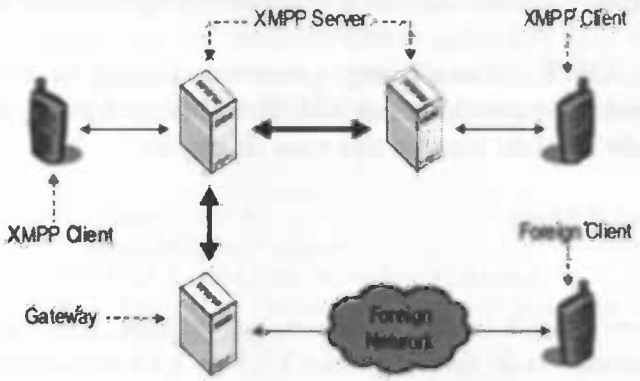
Zo zijn er vele voorbeelden te bedenken. XML is uiterst flexibel, niks is vantevoren gedefinieerd. Alles is mogelijk. Alle tags kunnen gedefinieerd worden, de tags dienen wel afgesloten te worden met de end-tags. Bijvoorbeeld `<note>` moet worden afgesloten met `</note>`. De eerste regel, de XML declaratie, definieert de XML versie en de encoding van de karakters (ISO-8859-1). De tags moeten wel correct genest worden, dus de end-tags zijn in omgekeerde volgorde dan de begin-tags.

Bijvoorbeeld `<note><heading>Herinnering<note><heading>` is fout. Sommige tags kunnen een attribuut bevatten, deze moeten altijd tussen quotes staan. Bijvoorbeeld `<note date="01/01/2005">`.

3.3.3. XMPP protocol (Jabber)

XMPP is een open en XML-gebaseerde protocol voor (bijna) real-time extensible messaging en presence. Het is de kern protocol van de Jabber Instant Messaging en Presence technologie, welke is opgesteld op duizenden servers op het Internet en wordt door miljoenen gebruikers wereldwijd gebruikt.

XMPP is officieel als IM protocol door het IETF in 2002 aangenomen, hierna zijn enkele technische specificaties opgesteld door de XMPP werkgroep, zie daarvoor [A15].
Het XMPP architectuur bestaat uit XMPP servers, XMPP cliënts en gateways naar andere netwerken. Zie figuur 3.3.



Figuur 3.3: XMPP architectuur.

De XMPP server levert de messages via het netwerk van XMPP servers en/of gateways. Naast de connection management en message routing bieden de XMPP servers de cliënts ook de mogelijkheid om cliëntspecifieke data te opslaan, bijvoorbeeld contact-lijsten (in XMPP termen: rosters). De XMPP cliënts worden met de XMPP servers verbonden via een TCP connectie. Een gateway vertaalt XMPP in de protocol welke wordt gebruikt door andere (niet-XMPP) netwerken, en vice versa. Deze *foreign* netwerken kunnen zijn: Internet Relay Chat (IRC), Short Message Service (SMS), SMTP (e-mail) en andere IM netwerken (AIM, ICQ, MSN Messenger, etc.).

Adressering

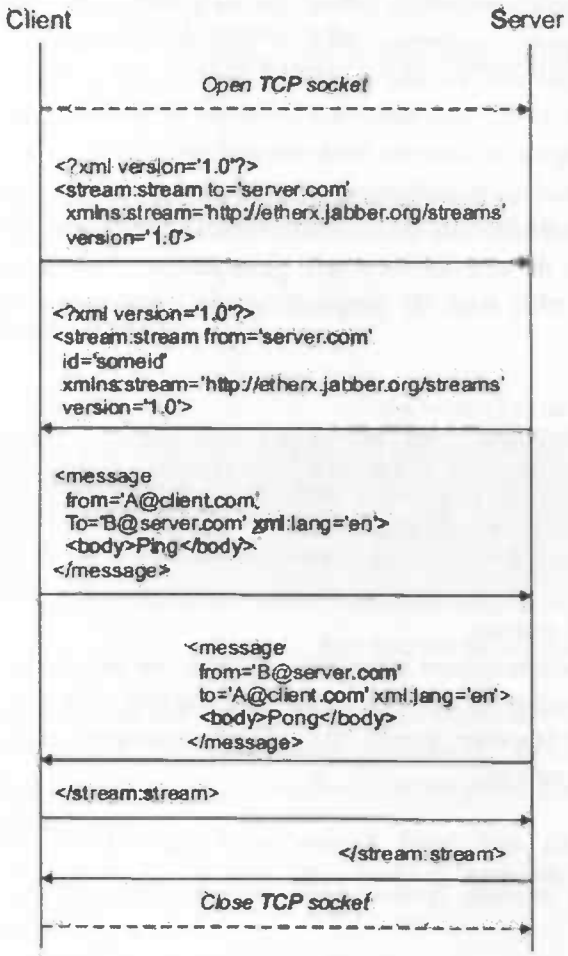
XMPP hanteert de concept van een entiteit, welke geïdentificeerd kan worden door een JID (Jabber ID). De JID is een combinatie van een e-mailadres en een URL; het bestaat uit een *node*, *domain* en *resource*. Alleen het domein gedeelte is verplicht, deze drie componenten zijn ter identificatie van de IM gebruiker, de XMPP server en de resource die de gebruiker gebruikt. Bijvoorbeeld een gebruiker genaamd "`Lisa`", gebruikt de "`LisaClient`" (resource) op "`jabber.xs4all.nl`" (domein), Lisa zou dan de JID hebben:

`Lisa@jabber.xs4all.nl/LisaClient` [3.0]

Gedeelde chatrooms worden ook geïdentificeerd aan de hand van de JID, bijvoorbeeld `chatroom1@jabber.xs4all.nl` identificeert een chatroom genaamd "`chatroom1`", welke beschikbaar is op het domein `jabber.xs4all.nl`.

XML streams

XMPP is gefundeerd op twee belangrijke concepten: XML streams en XML stanza's [A14]. Het idee van XML streaming is dat in plaats van een enkele connectie, een voortdurende verbinding wordt gebruikt om XML data elementen (XML stanza's in XMPP termen) over en weer te verzenden. Een basis communicatie over en weer tussen A en B vindt volgens onderstaand schema (figuur 3.4) plaats.



Figuur 3.4: Een simpele interactie tussen XMPP cliënt en server.

Er zijn drie gedefinieerde XML stanza's in XMPP: `message`, `presence` en `iq`. Alle drie hebben gemeenschappelijke attributen: `to`, `from`, `id`, `type` en `xml:lang`. Het `to` attribuut refereert naar welke JID de stanza verzonden moet worden, het `from` attribuut refereert naar de verzender, de `id` is de identificatie code voor de stream, in `type` wordt onderscheid gemaakt tussen een IM bericht en een presence bericht. Het `xml:lang` attribuut refereert naar de taal van de content in de XML stanza's.

Hieronder volgen de beschrijvingen van de drie gedefinieerde stanza's, en code-voorbeelden.

1. `Message` stanza:
   Dit element wordt gepusht van de zender naar de ontvanger. Bijvoorbeeld een XMPP cliënt verstuurt een IM bericht naar een andere gebruiker, of naar meerdere gebruikers. De ontvangers worden geïdentificeerd naar de hand van de `to` attribuut. Op de volgende pagina is een voorbeeld in code weergegeven van een `message` stanza.

```
<message
      to='B@server.com'
      from='A@client.com'
      type='chat'
      xml:lang='en'>
    <subject>Hello user B!</subject>
    <subject xml:lang='nl'>Hallo gebruiker B!</subject>
    <body>Can you send me your picture?</body>
```

```
    <body xml:lang='nl'>Kun je mij een foto van jezelf
                        verzenden?</body>
    <thread>thread-xx-yy</thread>
</message>
```

2. `Presence` stanza:
   Dit element is het aankondigingsgedeelte dat de andere gebruikers/cliënts informeert naar de aanwezigheid van de desbetreffende gebruiker. De `to` attribuut wordt niet gebruikt, immers de `presence` wordt gebroadcast. Mocht de `to` toch worden gebruikt, dan wordt het enkel naar de gespecificeerde ontvanger verzonden. Een voorbeeld in code:

```
<presence xml:lang='en'>
    <show>not available</show>
    <status>Having a coffee break</status>
    <status xml:lang='nl'>Koffie pause</status>
    <priority>1</priority>
</presence>
```

3. `Iq` (Info/Query) stanza:
   Dit is een request-response interactie, waarbij om bepaalde informatie wordt gevraagd. De `type` attribuut geeft aan om welke operatie het gaat: `get`, `set`, `result` of `error`. In een succesvolle operatie wordt `result` geretourneerd naar de `get` en de `set`, in het andere geval de `error`.

```
<iq to='A@client.com' type='get' id='r1'>
    <query xmlns='jabber:iq:roster'/>
</iq>
```

Deze request vraagt om de roster (buddy-list) van A. De server stuurt de respons terug:

```
<iq to='A@client.com' type=result' id='r1'>
    <query xmlns='jabber:iq:roster'>
        <item jid='Lisa@jabber.xs4all.nl/LisaClient'
              name='Lisa'
              subscription='both'>
            <group>Relatives</group>
        </item>
        <item jid='chatroom1@jabber.xs4all.nl'
              name='ChatRoom1'
              subscription='from'>
            <group>ChatRooms</group>
        </item>
    </query>
</iq>
```

SIP (Session Initiation Protocol)

Naast XMPP bestaat ook het SIP protocol, voor IM toepassingen is daarmee SIMPLE (SIP for Instant Messaging and Presence Leverageing Extensions) tot stand gekomen. Echter er zijn enkele belangrijke verschillen [A15].

De architectuur van SIMPLE is behoorlijk complex, het is gebaseerd op de SIP architectuur. De middleware laag kan niet dynamisch aangepast worden, server-server verbindingen zijn namelijk niet mogelijk. Dit in tegenstelling bij XMPP. SIMPLE vereist ook meer bandbreedte, volgens [A15] vraagt een simpele "Hello" bericht bij SIMPLE twee keer zo veel geheugen als bij XMPP. In het geval van een presence bericht, kost het tien keer zo veel geheugen bij SIMPLE.

SIP wordt voornamelijk gebruikt bij internet-telefonie, session-based multimedia [W18].

3.3.4. SOAP (Webservices)

> *"Web Services are a standards-based software technology that lets programmers and integrators combine exitsting and new systems or applications in new ways over the Internet, within a company's boundaries, or across many companies. Web Services allow interoperability between software written in different programming languages, developed by different vendors, or running on different operating systems or platforms."*

Bron: IDC, Web Services: Analyst: Sophie Janne Mayo, Maart 2002.

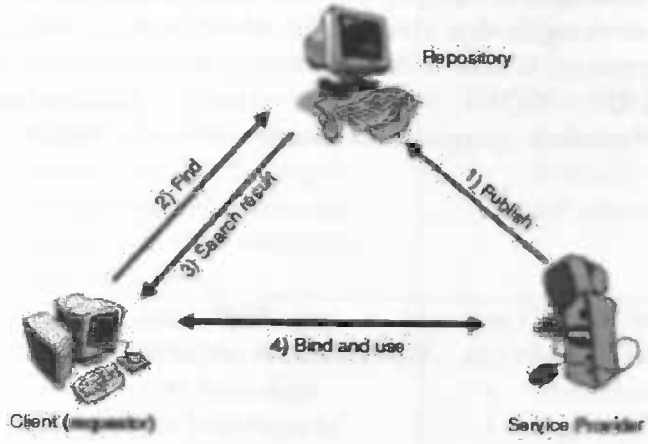
Webservices zijn dus services die via het web aangeroepen kunnen worden, de bestaande web-infrastructuur wordt hierbij gebruikt. Ze zijn 'netwerk-gebaseerde' services, waarbij de bestaande standaard technologies worden gebruikt, waardoor interoperabiliteit tussen verschillende systemen en applicaties mogelijk zijn.

Bij de webservices is de interoperabiliteit van groot belang, daartoe is het gehele webservice model opgedeeld in 3 delen.

Het model bestaat uit:

1. De client. Dit maakt gebruik van de webservice, het heeft stukjes informatie nodig.
2. De service provider. Deze verwerkt de informatie waar de cliënt om vraagt.
3. De repository. Dit is een soort telefoonboek van alle (beschikbare) webservices. Het bevat de omschrijvingen van de services, waarbij WSDL (Web Service Description Language) gebruikt wordt.

Hieronder staat dit model in figuur 3.5 weergegeven.



Figuur 3.5: Webservices model.

De cliënt en de service communiceren RPC-gewijs met elkaar, waarbij XML ook gebruikt wordt. De berichten zijn gecodeerd in SOAP.
Webservices hebben drie basis operaties:

1. Publish/Unpublish. De service provider maakt contact met de repository om diens services te registreren of het te verwijderen.
2. Find. Deze operatie wordt uitgevoerd door de requestor (cliënt) en daarbij wordt de cliënt verwezen naar de repository. Dit gebeurt via WSDL, de repository levert de informatie over hoe toegang verkregen kan worden tot de services. UDDI (Universal Description Directory and Integration) is de SOAP gebaseerde specificatie voor de respons van de repository.
3. Bind. Deze operatie vindt plaats tussen de cliënt en de service provider. Hierbij wordt nagegaan of de cliënt gebruik mag maken van de services.

De genoemde standaarden (XML, SOAP, WDSL en UDDI) zijn benodigd om gebruik te kunnen maken van de webservices. Ze zullen hierna worden beschreven.

XML

XML is een flexibele manier om brokken data te structureren, dat gebeurt op basis van diens formaat, structuur, lay-out. Dat structuur wordt gedeeld over het IP netwerk, waardoor van beide kanten (zender en ontvanger) bekend is in welke vorm de informatie wordt verzonden. Bijvoorbeeld, er moet informatie worden verzonden over de computer, dat kan zijn: processor snelheid, hoeveelheid geheugen, etc. Dit kan in XML worden gestandaardiseerd, voor meer informatie over XML zie paragraaf 3.3.2.

SOAP

Simple Object Access Protocol (SOAP) is een manier voor een programma om met andere programma's op andere platformen (bijvoorbeeld Linux vs. Windows) te kunnen communiceren. Hierbij wordt gebruik gemaakt van de technieken van HTTP en XML ter informatieuitwisseling. SOAP is dus een platform-onafhankelijke toegangsprotocol, een XML RPC.

WSDL & UDDI

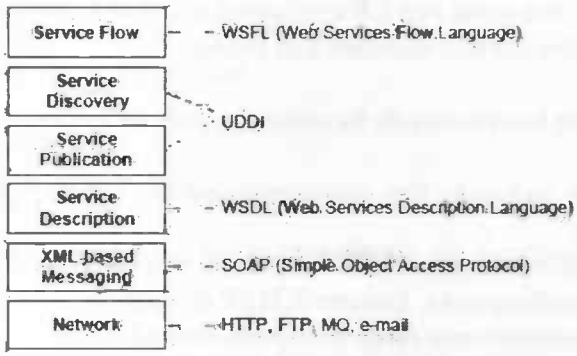
WSDL (Web Services Description Language) is een XML-gebaseerde taal, welke gebruikt wordt ter omschrijving van de aangeboden services. Hierdoor kunnen andere partijen gebruik van dezelfde services gaan maken. WSDL is de hoeksteen van het Universal Description, Discovery and Integration (UDDI). UDDI is een wereldwijde XML-gebaseerde register, waar geïmplementeerde webservices geregistreerd kunnen worden. WSDL is de taal waarin dat gebeurt.

WSFL

WSFL (Web Services Flow Language) is een XML taal, voor de beschrijving van de samenstelling van de webservices. WSFL houdt rekening met twee typen webservice samenstelling:

1. Flow models. Dit specificeert de gebruiksmogelijkheden van een collectie webservices, zodanig dat een nieuwe webservice geïmplementeerd kan worden. Hieruit komt vaak een business proces voort.
2. Global models. Dit specificeert de patroon van interactie van een collectie webservices, hieruit komt vaak een overzicht van interacties voort.

Deze standaarden van de webservices worden in figuur 3.5 weergegeven.



Figuur 3.6: Webservice layer structuur.

Momenteel zijn er twee standaarden waar webservices gebruikt kunnen worden: .NET en J2EE. Omdat het een eis van TNO ICT is om de applicatie in .NET te ontwikkelen, ligt de potentiele keuze bij de .NET versie. De .NET ontwikkelingsomgeving is optimaal ingericht voor gebruik en ontwerp van webservices.

3.4. Conclusie

Het maakt voor de dienst *Lisa* niets uit welke mobiele techniek gebruikt wordt, GPRS of UMTS. Toch wordt in het vervolg er van uit gegaan dat er UMTS gebruikt wordt. Immers, een hogere datasnelheid is vaak aangenamer voor het gebruik van een *LB* applicatie.

Wat betreft de protocollen, er zijn verschillende voor- en nadelen tussen de beide technieken, XMPP en Webservices. Ze worden hieronder op een rij gezet.

	XMPP	**Webservices**
Voordelen:	• IM aspect. • Open protocol. • Weinig dataverkeer. • Alleen verbinding tijdens verzenden. • Zeer flexibel vanwege eigen inrichting XML messages. • Middleware infrastructuur ontworpen voor 'many-to-one' verbinding.	• In .NET eenvoudig te implementeren. • Goede ondersteuning in .NET omgeving. • MapPoint en MiloWare ook in webservices. • Bedoeld voor verzenden van informatiestructuren.
Nadelen:	• Geen open compact framework implementatie. • Jabber server benodigd. • Bedoeld om berichtjes te verzenden en te ontvangen.	• Verbinding blijft tot respons binnen is. • Veel dataverkeer. • Lage snelheid (via GPRS/UMTS). • Registratie UDDI.

Bij de keuzebepaling spelen diverse factoren een rol:

- Nieuwe techniek. Het is uitdagend om zo veel mogelijk verschillende technieken te gebruiken. Het is voor TNO ICT ook van essentieel belang om kennis te maken met nieuwe technieken.
- Snelheid. Het verzenden en ontvangen mag geen of nauwelijks impact hebben op de continuïteit van de applicatie.
- Extra features. In het geval van *LB* applicaties is het ook van belang om te kijken naar de extra mogelijkheden die een techniek kan bieden.

Als gekeken wordt naar de bovenstaande factoren dan valt de keuze op XMPP;

- Het is een andere techniek. Een nieuw principe om een IM protocol te gaan gebruiken als RPC techniek.
- Het verzenden en ontvangen bij XMPP heeft een hoge snelheid, near real-time. Dit is niet het geval bij webservices. Immers XMPP is ontworpen voor instant messaging.
- Bij XMPP is de authenticatie reeds geïmplementeerd.
- Vanwege de IM eigenschappen zou in de toekomst zou een IM-chat module in de *LB* applicatie geïmplementeerd kunnen worden.

Ook voldoet XMPP aan de functionele requirements 4 en 5:

4. *De informatie moet binnen een acceptabele tijd over en weer worden verzonden:*
   XMPP is near real-time, de XML berichten bevatten zeer weinig data.
5. *Enige vormen van beveiliging is gewenst:*
   Aan deze requirement wordt voldoen in de vorm van de gebruikers authenticatie.

De nadelen van XMPP wegen niet op tegen de voordelen. Er zijn voldoende Jabber servers aanwezig op het world wide web. Dat het een IM protocol is, neemt niet weg dat er eigen ontworpen XML berichten over en weer verzonden kan worden. Alles is immers pure tekst.

Het nadeel dat er geen open implementatie is van Jabber voor het compact framework in .NET, kan worden opgeheven door te kijken naar COTS componenten die momenteel op de markt zijn.

In het volgende hoofdstuk zal de productontwikkeling van *Lisa* worden besproken, daarbij zal worden ingegaan op de resterende functionele requirements 6 en 7.

Hoofdstuk 4. Productontwikkeling *Lisa*

Inleiding

In dit hoofdstuk wordt de daadwerkelijke productontwikkeling van *Lisa* beschreven. Onder de productontwikkeling wordt voornamelijk verstaan de beschrijvingen van het volgende:

- Architectuur en implementatie
- Gebruik van MiloWare en MapPoint
- GPS in .NET
- Data- en objectmodel
- De UI structuur van de applicatie

Naast de productontwikkeling worden in dit hoofdstuk de functionele requirements 6 en 7 onder de loep genomen:

6. Het moet de beschikking hebben over geografische kaartgegevens.
7. De locaties en de routes moeten 'ergens' worden opgeslagen.

Een eis van TNO ICT is dat er gebruik wordt gemaakt van Microsoft MapPoint webservices. In feite is daarmee reeds voldaan aan de requirement 6, *het moet de beschikking hebben over geografische kaartgegevens*. Echter achteraf is gebleken dat MapPoint nogal wat nadelen heeft, en dat enkele trucs benodigd zijn om het optimaal te kunnen benutten. Daarover wordt meer verteld in subparagraaf 4.2.3.

Het zwaargewicht van dit hoofdstuk ligt bij de paragraaf 4.1, Architectuur. In dat paragraaf wordt toegelicht hoe alle technieken die in de vorige hoofdstukken zijn beschreven worden gebruikt. Na dat paragraaf volgt de beschrijving van het Data- en Object model. In deze paragraaf worden de objecten die gebruikt zijn voor *Lisa* beschreven

Ten slotte wordt in dit hoofdstuk de structuur van de user interface van *Lisa* uitgewerkt. Dit is voornamelijk het werk geweest van Sven Schultz, aangezien dit bij zijn opdracht hoorde.

4.1. Architectuur en implementatie

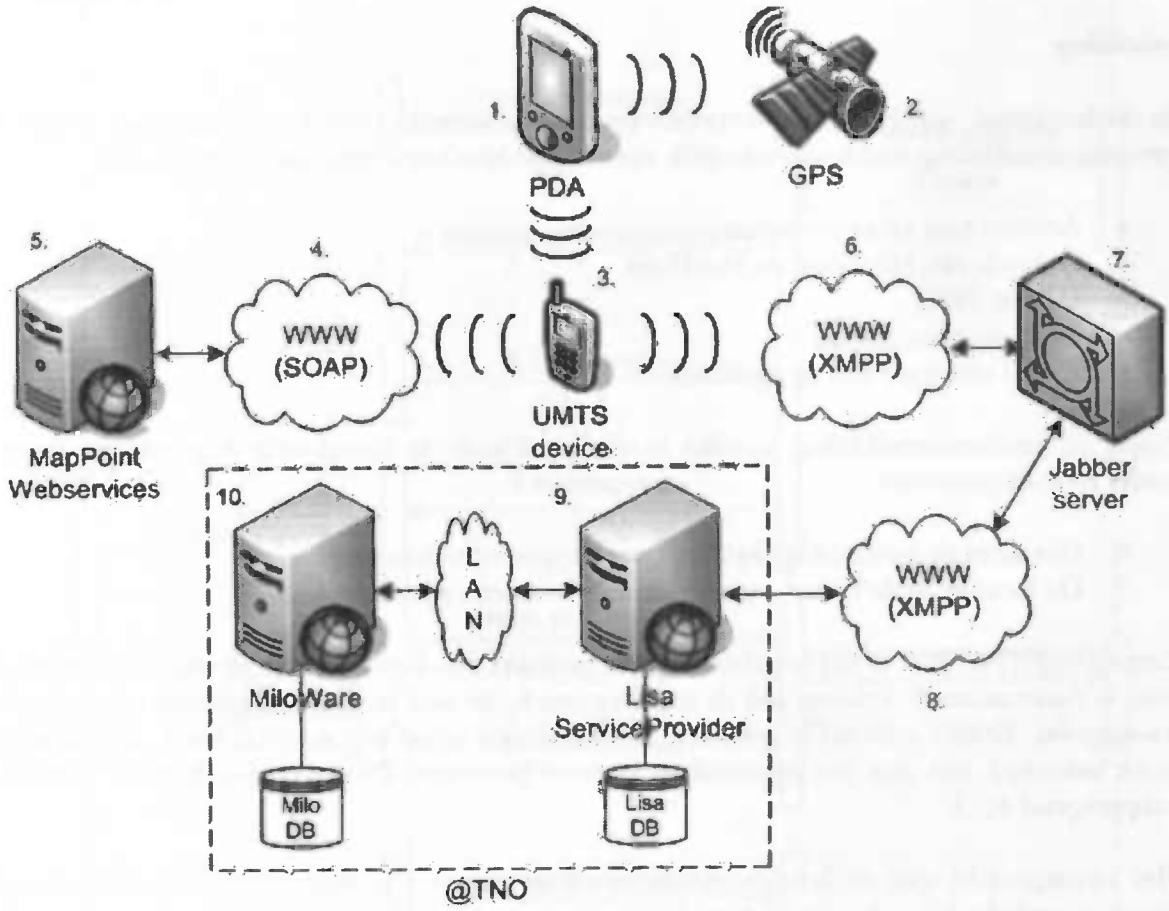
In deze paragraaf wordt de architectuur van *Lisa* uitgebreid behandeld. Onder een *software architectuur* wordt verstaan:

> *"The software architecture of a program or computing system is the structure or the structure of the system, which comprise software components, the externally visible properties of those components and the relationships among them."*

Bron: *Design & Use of Software Architectures*, J. Bosch, 2000.

Om een duidelijke en een simpel overzicht van de structuur van *Lisa* te verkrijgen, is een blik op figuur 4.1 op de volgende pagina voldoende. Het geeft de verschillende bouwstenen van het systeem weer. Daarbij wordt ook de techniek van de communicatie tussen de afzonderlijke bouwblokken weergegeven.

Deze bouwblokken zijn genummerd, 1 tot en met 10. De nummering wordt gehanteerd zodat er adequaat naar de bouwblokken verwezen kan worden in de toelichting.



Figuur 4.1: Domain model van *Lisa*.

Het middelpunt van het systeem lijkt de UMTS device (3) te zijn, maar in feite gebeurt het allemaal op de PDA (1). De UMTS device is via Bluetooth met de PDA verbonden, en is slechts een doorgeefluik van alle data. De device heeft verder geen enkele invloed op de inhoud van het dataverkeer. Dat geldt ook voor de Jabber server (7). Helaas was het niet mogelijk een interne Jabber server in TNO te draaien, dit wegens de aanwezige firewall. Het brengt veel rompslomp met zich mee om toch een Jabber server toegankelijk te maken voor buitenaf. Het gebruik van Jabber wordt nader toegelicht in subparagraaf 4.1.3.

De PDA is als het ware het gezicht van *Lisa* naar de gebruiker toe. Op de PDA zit de gebruikersinterface, waarmee de gebruikers met de dienst via het XMPP protocol (6, 8) kan communiceren. Die communicatie vindt in de vorm van requests en respons (sockets) plaats. De structuur van de interface van de applicatie wordt nader toegelicht in paragraaf 4.4, en het objectenmodel ervan wordt weergegeven in paragraaf 4.3.

Via SOAP webservices (4) wordt de MapPoint server (5) van Microsoft aangeroepen, om geografische kaarten te verkrijgen. Dit gebeurt aan de hand van de GPS positie (2). Over MapPoint wordt meer verteld in paragraaf 4.2. De positie wordt dus rechtstreeks verkregen op de PDA, door middel van de GPS ontvanger. De GPS ontvanger registreert en verwerkt het GPS signaal van de GPS satellieten constellatie.

Dan blijven de componenten *MiloWare* (10) en de *Lisa* serviceprovider (9) over. De *Lisa* cliënt heeft geen rechtstreekse verbinding met *MiloWare*, dit ook vanwege het veiligheidsaspect. Dat veiligheidsaspect is beter te beheersen als er slechts één entiteit is die *MiloWare* raadpleegt. Want *MiloWare* houdt ook bij in hoeverre de gegevens van de gebruikers zichtbaar zijn, dit is beter te beheersen als dat door één entiteit (de *Lisa* serviceprovider) beheerd wordt. Naast het veiligheidsaspect speelt het hebben van een eigen database ook een rol. De database van *MiloWare* heeft bijvoorbeeld geen tabellen voor routes. Dus een apart tussenstation is benodigd.

De *Lisa* serviceprovider is niets anders dan een database systeem die de posities en de requests van de gebruikers afhandelt en ook onderhoudt de serviceprovider de communicatie met het *MiloWare* platform. De requests zijn de aanvragen naar: routes, locatie informatie en de locatie gekoppelde berichten, dit wordt nader toegelicht in subparagraaf 4.1.1.

Daarmee worden de gegevens van de andere serviceproviders afgeschermd. Het gebruik van *MiloWare* in het systeem wordt nader toegelicht in subparagraaf 4.1.2.

4.1.1. Interne communicatie van het systeem

Er wordt in de cliënt van *Lisa* onderscheid gemaakt tussen twee lijnen; de gebruiker kan een route bewandelen (RouteLopen) of een nieuwe route aanmaken (RouteMaken). In de volgende secties wordt de interne communicatie tussen de cliënt en de serviceprovider apart, RouteLopen en RouteMaken, beschreven. Over het onderscheid tussen RouteLopen en RouteMaken wordt meer verteld in paragraaf 4.5.

De communicatie wordt gepresenteerd in de vorm van sequentiediagrammen, hierin wordt enkel de communicatie tussen de cliënt en de serviceprovider weergegeven. In de bijlage zijn de use-cases opgenomen, en de daarbij behorende uitgewerkte sequentiediagrammen. Daar zitten ook de diagrammen bij, die de communicatie over en weer tussen de gebruiker en de PDA voorstelt.

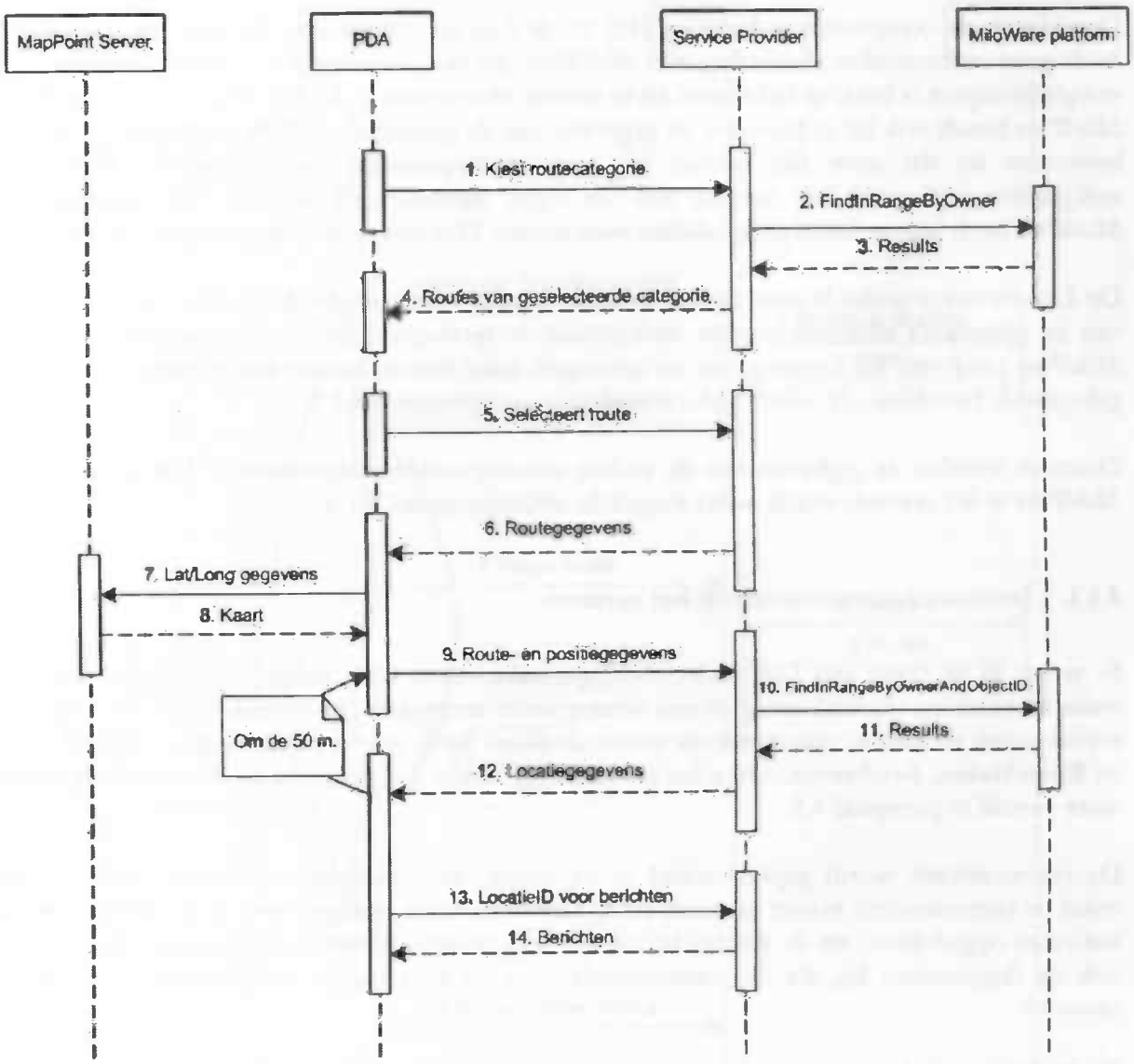
De in deze sectie beschreven communicatie verloopt steeds via de Jabber server (zie nummer 7 in de architectuur), dit wordt dan ook achterwege gelaten in de figuren 4.2 en 4.3.

In de figuren zijn de lanes tussen de levenslijnen ook genummerd, na de figuren volgt de toelichting.

RouteLopen

Op de volgende pagina is de grafische representatie van het dataverkeer welke plaatsvindt tussen de MapPoint server, de *Lisa* PDA, de *Lisa* Service Provider en het *MiloWare* platform. In dat overzicht is er reeds van uitgegaan dat de gebruiker al is ingelogd, en een bestaande route wilt bewandelen.

De communicatie tussen de PDA en de Service Provider vindt plaats via het XMPP protocol, de communicatie tussen de Service Provider en het *MiloWare* platform vindt plaats via webservices. De webservices worden ook gebruikt voor de communicatie tussen de MapPoint server en de PDA.



Figuur 4.2: Swimming lanes bij 'RouteLopen'

Eerst kiest de gebruiker de categorie van de route (1). Er zijn drie categorieën: 1. Officiële routes, 2. Populaire routes en 3. Gebruikersroutes. De selectie wordt met de GPS coördinaten, nader te noemen positie, naar de serviceprovider verzonden. De serviceprovider stuurt de coördinaten door naar het *MiloWare* platform (2). *MiloWare* gaat dan na welke locaties in de buurt zijn van de gegeven positie, en stuurt de resultaten hiervan terug naar de serviceprovider (3). De serviceprovider gaat in dat resultaat dan na welke locaties de startpunten vormen van de routes uit de geselecteerde routecategorie. Alle routes, waarvan de startpunten in de buurt zijn, worden opgehaald uit de database, en vervolgens geretourneerd naar de PDA van de gebruiker (4). De gebruiker krijgt dan alle routes gepresenteerd. De feitelijke data bestaat enkel uit de namen en de identificatie codes van de routes.

Uit de gepresenteerde routes kiest de gebruiker een route, hierdoor wordt de desbetreffende identificatiecode van de route naar de serviceprovider verzonden (5). Aan de hand van die identificatiecode wordt de gevraagde route uit de database opgehaald (6). De route bestaat dan uit de NAW gegevens van de locaties, diens identificatiecodes en diens posities die bij dat route horen. Naast de locatie informatie worden ook de latitudes en de longitudes van punten die op het pad van de route zitten, opgehaald. Deze gegevens worden dan verzonden naar de PDA.

De ontvangen positie-informatie (de posities van de locaties en de posities van de pad) vormen de parameters voor de request van een MapPoint kaart (7). Deze kaart wordt dan ingeladen en aan de gebruiker gepresenteerd in de applicatie (8).

Nu begint het echte wandelen. Zodra de gebruiker vijftig meter heeft afgelegd, wordt de positie en de identificatiecode van de eerstvolgende locatie naar de serviceprovider verzonden (9). Met die positie wordt in *MiloWare* nagegaan of de meegezonden locatiecode in de buurt is van die positie (10). Is dat het geval, dan worden vanuit de serviceprovider de locatiegegevens naar de PDA verzonden (12), dit is dus onder andere een *Location Based* aspect.

Zodra de locatiegegevens bij de PDA binnenkomen, wordt er gelijk een berichtenaanvraag naar de serviceprovider verzonden (13), met daarbij wederom een locatiecode. De serviceprovider gaat dan in de database na of er berichten bij dat locatie horen. Is dat het geval, dan worden deze berichten naar de PDA verzonden (14).

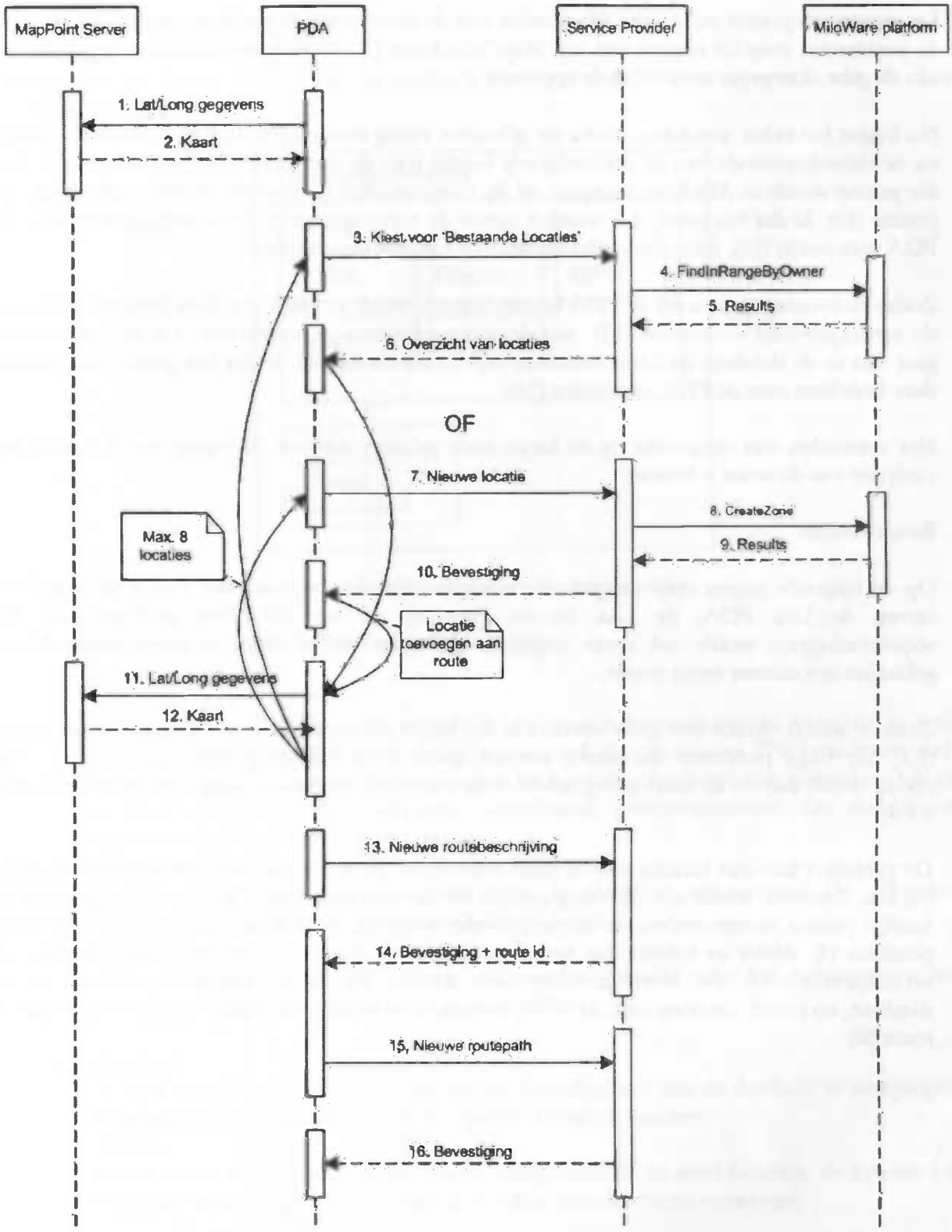
Het verzenden van de positie en de locatiecode gebeurt dus om de vijftig meter, totdat het eindpunt van de route is bereikt.

RouteMaken

Op de volgende pagina staat de grafische weergave van de communicatie tussen de MapPoint server, de Lisa PDA, de Lisa Service Provider en het MiloWare platform. Bij het sequentiediagram wordt ook ervan uitgegaan dat de gebruiker reeds is ingelogd, en dat de gebruiker een nieuwe route maakt.

Zoals te zien is vinden hier geen selecties in het begin plaats, de kaart wordt gelijk aangeroepen (1,2). De enige parameter die hierbij een rol speelt is de huidige positie van aanvraag, deze positie wordt dan op de kaart weergegeven in de vorm van een blauw vlaggetje (zie subparagraaf 4.1.4).

De gebruiker kan dan locaties aan de route toevoegen. Deze locaties kunnen bestaande locaties (3) zijn, daarvoor wordt een aanvraag gedaan bij de serviceprovider. Bij de aanvraag wordt de huidige positie meegezonden, de serviceprovider geeft die positie dan door aan het *MiloWare* platform (4). *MiloWare* bekijkt dan welke locaties in de buurt zijn, en geeft dat terug aan de serviceprovider (5). De serviceprovider haalt dan de informatie van al die locaties uit de database, en stuurt dat door naar de PDA, waarna de gebruiker de locatie kan toevoegen aan de route (6).



Figuur 4.3: Swimming lanes bij 'RouteMaken'

Naast het selecteren uit de bestaande locaties kan de gebruiker ook een nieuwe locatie toevoegen. De gebruiker heeft dan reeds de NAW en de beschrijving van dat locatie ingevoerd. Datgene wat is ingevoerd wordt dan naar de serviceprovider verzonden (7), de serviceprovider

slaat de informatie op in de database. Ook wordt de positie van de nieuwe locatie in de database van *MiloWare* opgeslagen (8,9). Ten slotte wordt een bevestiging verzonden naar de PDA (10).

Na elke opgeslagen locatie wordt de kaart van MapPoint ververst, de nieuwe locatie en het pad staan dan op de kaart (11,12).

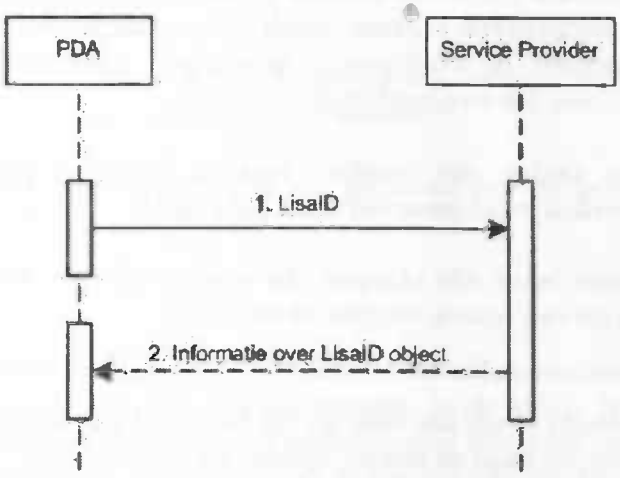
Tijdens het lopen wordt wederom om de vijftig meter de nieuwe positie opgeslagen. Deze informatie wordt op het eind verzonden, zodat het pad bekend is.

De gebruiker heeft de mogelijkheid om acht nieuwe of bestaande locaties toe te voegen aan de route, daarna wordt de route opgeslagen. Eerst wordt de routebeschrijving, die de gebruiker heeft ingevoerd, naar de serviceprovider verzonden (13). Na het verwerken van de beschrijving wordt een nieuwe identificatiecode aangemaakt voor dat route, die code wordt dan met een bevestiging terugverzonden naar de PDA (14). Ten slotte wordt die code op de PDA in de route verwerkt, waarna de gehele pad wordt verzonden (15). De serviceprovider slaat de pad op, en stuurt als dat goed gegaan is, een bevestiging terug (16).

Daarmee is het maken van een nieuwe route voltooid.

LisaID (vervanging van RFID)

Omdat er geen RFID reader voor de PDA voor handen is, is er een alternatieve oplossing ontwikkeld. De gebruiker voert een *LisaID* code in, deze code wordt dan verzonden naar de serviceprovider. De serviceprovider gaat dan in de database na welke informatie hierbij behoort, en stuurt deze informatie vervolgens terug naar de gebruiker.

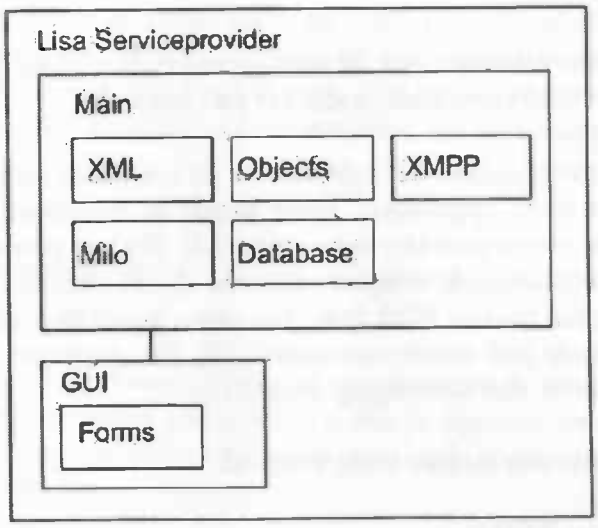


Figuur 4.4: Swimming lanes voor *LisaID*.

Er is tijdens het ontwikkelen niet veel aandacht hier aan besteed, omdat het om een zeer kleine feature gaat. Het is zeer aanbevolen in de toekomst de RFID mogelijkheden te integreren in de *Lisa* applicatie, voor zowel de scanning techniek en als lokalisatie techniek.

Interne structuur serviceprovider

Hieronder, in figuur 4.5, is een overzicht gegeven van de interne structuur van de *Lisa* serviceprovider.



Figuur 4.5: Componenten van de serviceprovider

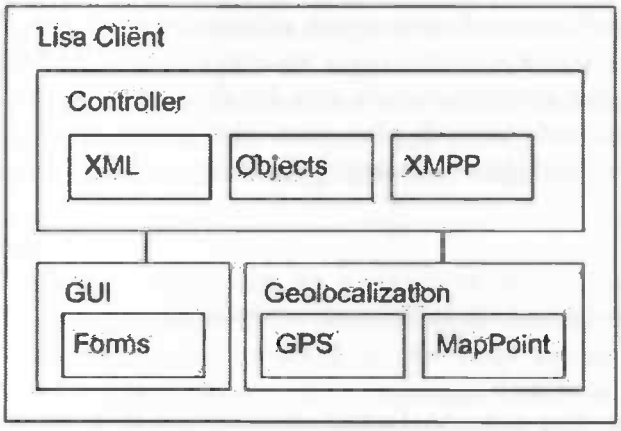
In het **main** gedeelte zitten de volgende componenten:

- XML:
  Hierin zitten alle functies die benodigd zijn voor het omzetten van objecten in xml-strings, en andersom. Dit wordt *serializen* genoemd. Er was binnen TNO reeds een xml-serializer geïmplementeerd. Hiervan wordt bij het *Lisa* project gebruik gemaakt. Ook zit in deze component de configuratie instellingen geïmplementeerd. De instellingen worden in een `.xml` file weggeschreven.
- Milo:
  In feite is dit slechts een interface naar de *MiloWare* webservices. De gebruikte webservices worden in subparagraaf 4.1.2 behandeld.
- Objects:
  Deze component bevat alle objecten die worden gebruikt voor het oproepen en het aanmaken van nieuwe routes, locaties en berichten.
- Database:
  In deze component zitten alle functies die benodigd zijn om de database te raadplegen. Hieronder wordt verstaan de insert-, update- en select functies.
- XMPP:
  Hierin zitten de Jabber functies, dus de inlog-functies, de send-functies, de listener om berichten te ontvangen. Ook zitten de serialize functies hierin verweven.

En in het **GUI** gedeelte zitten de forms, in feite is dit niets anders dan een textboxje waarin men kan zien wat er allemaal ontvangen en verzonden worden.

Interne structuur PDA cliënt

Hieronder, in figuur 4.6, is een overzicht gegeven van de interne structuur van de *Lisa* cliënt.



Figuur 4.6: Componenten van de cliënt.

De weergegeven componenten die in het **controller** gedeelte zitten hebben vrijwel identieke functies als die van de serviceprovider. Toelichting daarover wordt daarom achterwege gelaten. Wat wel wordt toegelicht is het **geolocalization** gedeelte. Daarin zitten de componenten die met geografische (kaart) gegevens werken.

- GPS:
  Deze component bevat alle functies waarmee de latitudes en de longitudes van de positie verkregen kan worden, hierin zitten NMEA parsers. Ook wordt hierin de pad, bij het routemaken, bijgehouden.
- MapPoint:
  Hierin zitten functies die de positie gegevens van locaties, de pad, etc. verwerken en functies die requests doen voor MapPoint kaarten.

Verder zitten in het **GUI** gedeelte de forms die interactie met de gebruiker mogelijk maken. Daarover wordt meer verteld in paragraaf 4.4

4.1.2. *MiloWare*

In deze subparagraaf worden de gebruikte functies van *MiloWare* beschreven, en daarbij ook de nieuwe functies die speciaal voor *Lisa* zijn aangemaakt. In de bijlage is een overzicht van alle *MiloWare* functies opgenomen. De nieuwe functies waren benodigd want de functie `XnearY Subscription` werkte niet naar behoren. Deze niet-werkende functie bekijkt of een gegeven entiteit in de buurt is van een andere entiteit is, *Location Based* operaties kunnen daaraan gekoppeld worden.

Termen van *MiloWare*

Om de wereld van *Milo* te doorgronden worden enkele *Milo* termen verklaard.

In *MiloWare* kunnen op de gebruikers zelf geen *Location Based* operaties worden uitgevoerd, want er kunnen meerdere devices zijn die bij de gebruiker horen. Een laptop, een PDA, een GSM toestel, etc. Daarom worden per device een zogenaamde *trust* aangemaakt. Die trust is de relatie

tussen de serviceprovider en de device, waardoor er een identificatie mogelijkheid ontstaat waarmee *LB* operaties uitgevoerd kan worden.

Omdat de meeste gebruikers het niet wenselijk vinden dat hun locatiegegevens op straat komt te liggen, bestaat er in *MiloWare* een structuur van zichtbaarheidsniveaus. Dit wordt de *visibilitytype* genoemd, er zijn drie niveaus: `miloPrivate`, `miloProtected` en `miloPublic`. Bij de eerste kunnen de trusts alleen een identificatiecode zien, bij de tweede kunnen de trusts alle informatie die beschikbaar is over de device/gebruiker zien. Het laatste houdt in dat iedereen, dus ook andere gebruikers, serviceproviders en locationproviders, alle informatie kunnen zien.

Een kort overzicht:

De wereld van *MiloWare* bestaat uit de volgende onderdelen:

1. Element:
   Een element kan zijn: persoon, locatie of een device in de werkelijke wereld. Ze worden ook wel entiteiten genoemd. De persoon is dus de eigenaar van een device (bijvoorbeeld een PDA). Een locatie is dus een bepaalde punt in de fysieke wereld, waar de persoon of device zich bevindt.
2. Relatie:
   Een relatie is de relatie tussen twee of meer elementen. De relatie wordt geïdentificeerd aan de hand van de identificatiecode van de trust. Er zijn varianten mogelijk, dat zijn dus de relaties tussen:
   - Persoon – device
   - Serviceprovider – device
   - Device – device
3. Rule:
   Dit is een set van condities welke gecombineerd wordt aan acties, d.w.z. welke acties ondernomen moet worden als er aan de gegeven condities worden voldaan. Bijvoorbeeld dat de serviceprovider een notificatie krijgt (actie) als een element met een bepaalde trust identificatiecode op een bepaalde punt is (conditie).
4. Service:
   Dit is een taak, welke uitgevoerd kan worden door *MiloWare* voor een dienst. Meestal betreft dit de taak waarbij informatie wordt opgevraagd, bijvoorbeeld de `PositionNow` en de `DistanceXY` functies.

De gebruikte webservices

Omdat de serviceprovider en de locationprovider in *Lisa* gelijk zijn, is er slechts een paar webservices benodigd. De benodigde webservices zijn hieronder beschreven, ze worden alleen aangeroepen vanuit de *Lisa* serviceprovider (zoals reeds is vermeld in sectie 4.1 over de architectuur). De beschreven webservices zijn de services die worden aangeroepen bij lopende interacties. De serviceprovider en de locationprovider dienen ook bij *MiloWare* geregistreerd te worden, echter dit is een eenmalige actie, en worden daarom buiten beschouwing gelaten. Bij de onderstaande webservices mag ervan worden uitgegaan dat de SP en de LP zijn geregistreerd onder de naam `miloLogin` en het daarbijbehorende wachtwoord, `miloPassword`.

Tijdens de implementatiefase kwam naar voren dat de subscription functie `XnearY` niet naar behoren werkt. Dit is echter een zeer belangrijke functie die nagaat of twee gegeven entiteiten bij elkaar in de buurt zijn, voor *Lisa* een fundamentele functie. Omdat dit niet werkt, is een alternatief uitgedacht. In plaats van passief te wachten of de twee gegeven entiteiten bij elkaar in

de buurt zijn, wordt er vanuit de serviceprovider actief nagegaan of die twee bij elkaar in de buurt zijn. Daarvoor bestaat in *MiloWare* een functie: `FindInRange`. Deze functie gaat na welke entiteiten in de buurt zijn van een gegeven entiteit. Echter deze functie is lang niet zo snel als de `XnearY` functie, dit komt omdat `FindInRange` steeds nagaat of de 'aanroeper' wel geauthoriseerd is om de gevonden entiteiten te mogen zien. In overleg met de ontwikkelaars van *MiloWare* is besloten dat Albert Wijbenga naar eigen inzicht alternatieve functies, afgeleid uit `FindInRange`, mag implementeren. Deze functies zijn onder toezicht van de ontwikkelaars geaccepteerd. De alternatieve functies zijn geworden: `FindInRangeByOwner` en `FindInRangeByOwnerAndObjectID`. Ze worden hieronder toegelicht.

- `FindInRangeByOwner`:
  Deze service is geïmplementeerd in de `ServiceProvider` module. Deze functie is afgeleid van de oorspronkelijke `FindInRange` functie, want in `FindInRange` wordt ook nagegaan of de aanvrager geauthoriseerd is, dit vergt echter nogal tijd. Daartoe is `FindInRangeByOwner` geïmplementeerd, de aanvrager bekijkt alleen de objecten die zijn aangemaakt door de aanvrager zelf. Immers; de maker van objecten mag ook diens objecten bekijken. Dit werkt vele stukken sneller.

  De `FindInRangeByOwner` bekijkt dus of een gegeven entiteit in de buurt is van een andere entiteit.

  Een (algemene) aanroep hiervan is:

  ```
  MiloServiceProvider msp = new MiloServiceProvider();

  MiloServiceProvider.ReturnResult r =
  msp.FindInRangeByOwner(miloLogin, miloPassword, TrustID,
  miloRange, miloLogin);
  ```

  De resultaten van de aanroep worden opgeslagen in `r` en zodoende kan daarin gekeken worden of er entiteiten zijn in de buurt. De identificatie van de owner zit in de `TrustID` parameter. De straal waarin gekeken moet worden is de `miloRange` parameter.

- `FindInRangeByOwnerAndObjectID`:
  Deze webservice is wederom afgeleid van de `FindInRangeByOwner` service, en komt ook van de `ServiceProvider` module. Het is bedoeld om na te gaan of de gebruiker in de buurt is van een bepaalde locatie. Dus in plaats van alle objecten in de buurt, wordt enkel gekeken naar dát object met de identificatiecode. De `FindInRangeByOwner` haalt dus <u>alle</u> entiteiten die in de buurt zijn.
  Daartoe wordt aan de `FindInRangeByOwnerAndObjectID` webservice een extra parameter toegevoegd, waardoor de service enkel nagaat of dát object in de buurt is.

  Een (algemene) aanroep hiervan is:

  ```
  MiloServiceProvider msp = new MiloServiceProvider();

  MiloServiceProvider.ReturnResult r =
  msp.FindInRangeByOwnerAndObjectID(miloLogin, miloPassword,
  TrustID, miloRange, miloLogin, miloID);
  ```

  Deze functie is vrijwel identiek als `FindInRangeByOwner`, alleen wordt het te vinden object ook gespecificeerd. Dit gebeurt in de vorm van de parameter `miloID`. Deze parameter is dus de identificatiecode van bijvoorbeeld een locatie.

- `NewAbsolutePosition`:
  Deze service is afkomstig van de `LocationProvider` module, deze service zorgt ervoor dat de positie van de device geregistreerd wordt, zodat het *MiloWare* platform *Location Based* operaties kan uitvoeren met de devices.

  Een (algemene) aanroep is:

```
MiloLocationProvider mlp = new MiloLocationProvider();

MiloLocationProvider.ReturnResult r =
mlp.NewAbsolutePosition(miloLogin, miloPassword, trustID,
longitude, latitude, 0);
```

  Met deze functie wordt de device vastgesteld op de gegeven latitude en longitude. De parameter `0` is het aantal seconden dat het device gekoppeld mag blijven aan de gegeven locatie. Echter, dit is niet van belang voor *Lisa*, en wordt daarom op nul gezet.

- `CreateZone`:
  De CreateZone service is ook afkomstig van de LocationProvider module

```
MiloLocationProvider mlp = new MiloLocationProvider();

MiloLocationProvider.ReturnResult r = mlp.CreateZone(miloLogin,
miloPassword, miloID, locNaam, locOmschrijving, longitude,
latitude, "Locaties", miloProtected);
```

  Met deze functie worden dus nieuwe locaties in de wereld van *Milo* aangemaakt. De parameter termen spreken voor zich waar ze voor dienen.

De volgende functies zijn allemaal benodigd ter registratie van de gebruikers en de devices.

- `InitEndUser`:
  Dit komt van de `Operator` module. Het zorgt ervoor dat een nieuwe gebruiker in de wereld van *Milo* wordt aangemaakt. Aan dat gebruiker kan dan devices (laptop, PDA, etc.) worden gekoppeld. Een voorbeeld volgt hierna, voor de gebruiker Lisa:

```
MiloOperator mo = new MiloOperator();

MiloOperator.ReturnResult r = mo.InitEndUser(miloAdminLogin,
miloAdminPassword, "", lisa, lisaPassword, trustID, false, "");
```

  Alleen de administrator mag nieuwe gebruikers toevoegen, vandaar de admin password en de admin loginnaam. De `""` parameter is de URL vanwaar de gebruiker geregistreerd wordt, dit is niet benodigd voor *Lisa*. Want er vinden geen rechtstreekse communicatie plaats met de cliënt. De parameter `lisa` is dus de gebruikersnaam, en `lisaPassword` spreekt voor zich. Ten slotte wordt een trust identificatiecode aangemaakt. De resterende twee parameters behoeven geen uitleg, ze zijn irrelevant.

- `AddDevice`:
  Deze functie is afkomstig van de `Login` module, en koppelt een device aan de gebruiker. Dat gebruiker wordt geïdentificeerd aan de hand van de benaming van de gebruiker. Hierdoor kunnen trusts worden aangemaakt tussen de device en de

serviceprovider. Met de onderstaande code wordt een PDA gekoppeld aan de gebruiker Lisa:

```
MiloLogin ml = new MiloLogin();

MiloLogin.ReturnResult r = ml.AddDevice(lisa, lisaPassword,
"PDAlisa", "PDA van lisa","", miloProtected);
```

Eigenlijk spreken de parameters voor zich.

- `AddTrust`:
  Met deze functie (ook afkomstig van de `Login` module) worden trusts aangemaakt tussen de device en de serviceprovider. Deze trust wordt dan bij alle *Location Based* operaties gebruikt om te identificeren om welke device het gaat.

```
MiloLogin ml = new MiloLogin();

MiloLogin.ReturnResult r = ml.AddTrust(lisa, lisaPassword,
"PDAlisa", miloLogin, active, trustID);
```

In het *Lisa* project zijn de identificatie codes van de trusts gelijk aan de gebruikersnaam, waardoor het eenvoudiger wordt om de database van *Lisa* te koppelen aan *MiloWare*.

4.1.3. Jabber

In deze sectie wordt het gebruik van de Winfessor SoapBox pakket nader toegelicht. Het betreft dus een COTS component, dus enkel de aangeroepen functies worden toegelicht.

Inloggen

Bij de inlog sessie wordt verondersteld dat het domein `jabber.xs4all.nl` de Jabber server is, die de communicatie tussen twee IM cliënts verzorgt. In subparagraaf 3.3.3 zijn reeds de Jabber termen verklaard.

Om bij de server te inloggen wordt de onderstaande code gebruikt:

```
m_Session = Login(username, password, resource, "jabber.xs4all.nl");
AvailableRequest announce = new AvailableRequest();
m_Session.SendAndForget(announce);

prd = new Session.PacketReceivedDelegate(OnIncomingMessage);
m_Session.AddHandler(typeof(Winfessor.SoapBox.Core.Message.MessagePacket
), prd);
```

De eerste regel is de daadwerkelijke login procedure, waarbij de gebruikersnaam, het wachtwoord, de resource en het domein van de server als parameters functioneren. Het tweede regel is het aanmaken van de *presence* van de gebruiker. In de derde regel wordt deze presence verzonden. De credentials worden reeds opgeslagen in de `m_session` variabele.

Ten slotte wordt in de laatste drie regels de listener aangemaakt, waardoor de applicatie XML berichten kan ontvangen.

Serialize, send en receive

Om de XML berichten te kunnen verzenden worden eerst de objecten omgevormd in XML. Dit gebeurt door middel van de XML *serializer*. In het onderstaande voorbeeld wordt het object `locatie` eerst geserialized, en daarna verzonden.

```
XmlSerializer xs = new XmlSerializer();
MessagePacket message = new MessagePacket();
message.Subject = "locatie";
message.Body = xs.Serialize(locatie);

message.To = server;
message.From = user;
m_Session.SendAndForget(message);
```

De variabele `server` is dus de JID, met daarin de domein informatie, de serviceprovider gegevens. Dat geldt ook voor de variabele user waarin de informatie over de gebruiker verwerkt zit. Deze structuur wordt aan beide kanten, serviceprovider en cliënt, gehanteerd.

Bij de serviceprovider en de cliënt zitten listeners geïmplementeerd, die 'luistert' naar inkomende berichten. Zodra de berichten binnenkomen, worden ze eerst aan de hand van de string in het `subject` attribuut naar de juiste handler gebracht. Daar wordt de XML bericht, indien nodig, gedeserialized. Bij de onderstaande stukje code wordt er reeds van uitgegaan dat de XML bericht al bij de juiste handler is gebracht:

```
XmlSerializer xs = new XmlSerializer();
locatie = (Locatie)xs.DeSerialize(m.Body, typeof(Locatie));
```

Dan is het object aangekomen bij de serviceprovider. Hierna kunnen database operaties worden uitgevoerd. Het hangt af van de subject string of het een insert of een select operatie betreft. Bij een insert operatie worden de attributen in het object in een `datarow` gezet, en tenslotte in de database van *Lisa* opgenomen. Betreft het een select operatie, dan worden eerst de gegevens uit de database in een `datarow` gezet, deze wordt dan omgevormd tot een object (de `ConvertDBToObject` functie).

In feite wordt er geen invloed uitgeoefend op de xml structuur van het Jabber protocol. Alleen het `<body>` gedeelte wordt ingevuld met xml strings, welke zijn geproduceerd door de serializer.

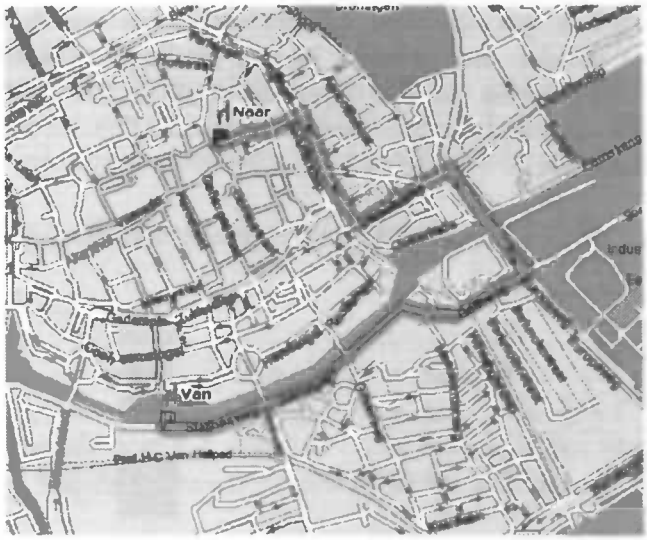
Een belangrijk nadeel van de gebruikte XML serializer is dat het niet dynamisch is, het kan slechts statische objecten serializen. Dus geen arraylists (vectoren in Java-termen). Hierdoor is het dataverkeer enigszins omvangrijk, zie paragraaf 4.3.2 (objectenmodel).

4.2. Microsoft MapPoint

Een van de onderdelen van de opdracht is dat we gebruik maken van MapPoint. In deze paragraaf wordt het gebruik van Microsoft MapPoint nader toegelicht. Daaronder wordt verstaan: algemene informatie over MapPoint, stukjes code om kaarten te genereren, hoe elementen op de kaart te plaatsen en additionele informatie over de licentie voor het gebruik van MapPoint. Ook worden in deze paragraaf de voor- en nadelen van MapPoint beschreven.

4.2.1. Wat is MapPoint en is het wat?

MapPoint Web Service is een XML gebaseerde webservice, welke derden kunnen gebruiken om kaarten te integreren in *LBS* applicaties. De laatste versie van deze webservice is versienummer 3.5. MapPoint is niet alleen in staat kaarten te genereren, maar ook om routes te berekenen. Echter deze routes zijn enkel geschikt voor auto's. Op de volgende pagina is daar een voorbeeld van te zien, de route is van het Groninger Museum naar de Martinitoren. Deze route is totaal ongeschikt als wandelroute. Er is bij Microsoft geïnformeerd naar de eventuele mogelijkheden om wandelroutes te genereren, echter dit is en blijft niet mogelijk. De feature om routes te berekenen valt daarom direct af.



Figuur 4.7: MapPoint alleen voor auto's.

Er dient dus een andere oplossing voor het laten uittekenen van routes op een MapPoint plattegrond bedacht te worden. Daarover wordt meer verteld in subparagraaf 4.2.2. Naast dit nadeel heeft MapPoint nog enkele noemenswaardige nadelen;

- Het aanvragen en het downloaden van MapPoint plattegronden gaat erg traag. Het opvragen en weergeven van 1 plattegrond duurt ongeveer 25 seconden wanneer gebruik gemaakt worden van GPRS (voor UMTS geldt een tijd van ongeveer 15 seconden).
- Er is veel dataoverdracht. Er is berekend dat met MapPoint gemiddeld 300 tot 500 kilobytes wordt verzonden, en zonder MapPoint wordt gemiddeld 50 tot 150 kilobytes verzonden.
- Geen duidelijke kaarten. De details op de plattegrond onderscheiden zich onvoldoende (dit is een bevinding van het onderzoek van Sven Schultz).

Er is een aantal redenen dat MapPoint toch is gekozen als leverancier voor de plattegronden tijdens het ontwikkelen van Lisa, dit zijn:

- De evaluatie versie voor MapPoint webservices is gratis, en het was gelukt deze te verlengen van 45 dagen naar het einde van de afstudeerperiode.
- TNO wil graag de mogelijkheden van MapPoint weten voor eventuele toepassingen in de toekomst.

- Alternatieve GIS aanbieders vragen veel geld voor het beschikbaar stellen van geografische kaarten. Bedragen vanaf € 50.000,- per jaar.
- In de toekomst zullen snellere dataverbindingen zorgen voor snellere weergave van de plattegronden.

De voordelen wegen zwaarder op tegenover de nadelen. Er is daarom gekozen voor het gebruik van MapPoint, en niet voor een alternatief (bijvoorbeeld ArcGIS).

4.2.2. MapPoint webservices code

Het is dus niet mogelijk om MapPoint routes te laten tekenen op de plattegronden. Hiervoor is een oplossing bedacht. Tijdens het maken van de routes worden de GPS coördinaten om de vijftig meter bijgehouden in een object en bij het verzenden van de route opgeslagen in de database. Die gegevens worden dan gebruikt bij het aanvragen van een MapPoint kaart, alle coördinaten worden meegegeven in het `pushpin` object. Want MapPoint heeft wél de mogelijkheid om zelf, aan de hand van latitudes en longitudes, iconen op de kaart te plaatsen. Er zijn in totaal 175 stock iconen.

In de `pushpin` object worden de iconen gespecificeerd, en diens latitude en longitude. Zie figuur 4.x.

Item	*Icoon*
Beginpunt van de route	
Locatie in de route	
Eindpunt van de route	
Dot voor de looproute	

Figuur 4.8: Iconen voor *Lisa* MapPoint.

De iconen hebben respectievelijk de nummers 31, 4, 29 en 8. Een `pushpin` object is een array van pushpins. Voor een compleet overzicht van alle iconen, zie [W15].

```
Pushpin[] pushpins = new Pushpin[1];
pushpins[0] = new Pushpin();
pushpins[0].IconDataSource = "MapPoint.Icons";
pushpins[0].IconName = "31";
pushpins[0].LatLong.Latitude = latitude;
pushpins[0].LatLong.Longitude = longitude;
pushpins[0].ReturnsHotArea = false;
```

De bovenstaande code geeft aan hoe de icoon 31 op de coordinaten (`latitude,longitude`) wordt geplaatst. De lengte van de array kan naar wens gespecificeerd worden.

Het `pushpin` object wordt dus meeverzonden met de request voor een MapPoint kaart. In appendix VI is een volledige procedure voor het aanvragen van een MapPoint kaart opgenomen.

De benodigde parameters zijn: een picturebox object, een zoom variabele, de latitude en de longitude, de latitudes en longitudes van de locaties op de route, de lengte van het looppad, het looppad (array van latitudes en longitudes) zelf.

De eerste latitude en longitude zijn benodigd om het midden van de route te bepalen, dit is dan het middenpunt van de plattegrond, zodat er enige zekerheid is dat de gehele route op de kaart wordt weergegeven. De lengte van het `pushpin` object is de som van de lengte van de locatie array en de lengte van het looppad. Middels een for-lus worden de iconen op de juiste plekken gespecificeerd.

Ten slotte worden al deze gegevens plus nog enkele additionele informatie in het `mapspec` object opgeslagen, en dit is dan de gehele specificatie voor de te aanvragen plattegrond.

Cliënt-sided MapPoint iconen tekenen
Naast het 'laten tekenen' door de MapPoint server, is het ook mogelijk om op de cliënt zelf objecten te tekenen op MapPoint. Het tekenen zelf is niet moeilijk, daarvoor kan het `System.Drawing` API gebruikt worden. Wat wel lastig is, is het tekenen van de juiste pixels.

Echter de MapPoint API heeft daarvoor twee functies:

- `ConvertToLatLong`
- `ConvertToPoint`

De eerste functie zorgt ervoor dat een gegeven pixelpunt wordt omgevormd tot een latitude/longitude. De tweede doet dat andersom. Dat tweede is dus van belang bij het tekenen van een icoon op de MapPoint kaart, op die manier kunnen de juiste pixels van kleur veranderen.

Er dient hierbij in acht worden genomen dat de output van de genoemde functies niet 100% nauwkeurig is, afwijkingen van tot 10 pixels zijn mogelijk.

4.2.3. Licentie MapPoint

Er zijn verschillende licentiemogelijkheden voor MapPoint. Ze zijn hieronder in een tabel gepresenteerd:

Licentievorm	Toelichting	Kosten
45-daags evaluatie	Voor het verkennen van mogelijkheden van MapPoint, voor research op universiteiten.	Gratis. Max. 45 dagen.
MSDN speciale evaluatie	Voor dezelfde doeleinden als de 45-daags evaluatie, plus de mogelijkheid voor 50.000 aanroepen op commerciële basis. Dit kan dus gebruikt worden voor test-pilots bij consumenten.	Gratis. Looptijd 1 jaar, 50.000 commerciële aanroepen.
Licentie	Dit is de echte licentie, voor commerciële doeleinden. Er zijn verschillende varianten mogelijk, bijvoorbeeld met of zonder routing.	€6.500,- voor 500.000 transacties, waarbij additionele transacties tussen 0,0065 en 0,00415 euro kosten.

Om *Lisa* te laten fungeren als portfolio voor het *MiloWare* project is er een MSDN evaluatie aangevraagd. TNO ICT kan de komende jaar *Lisa* blijven demonstreren.

Voor verdere informatie over de licentiemogelijkheden wordt verwezen naar [W15].

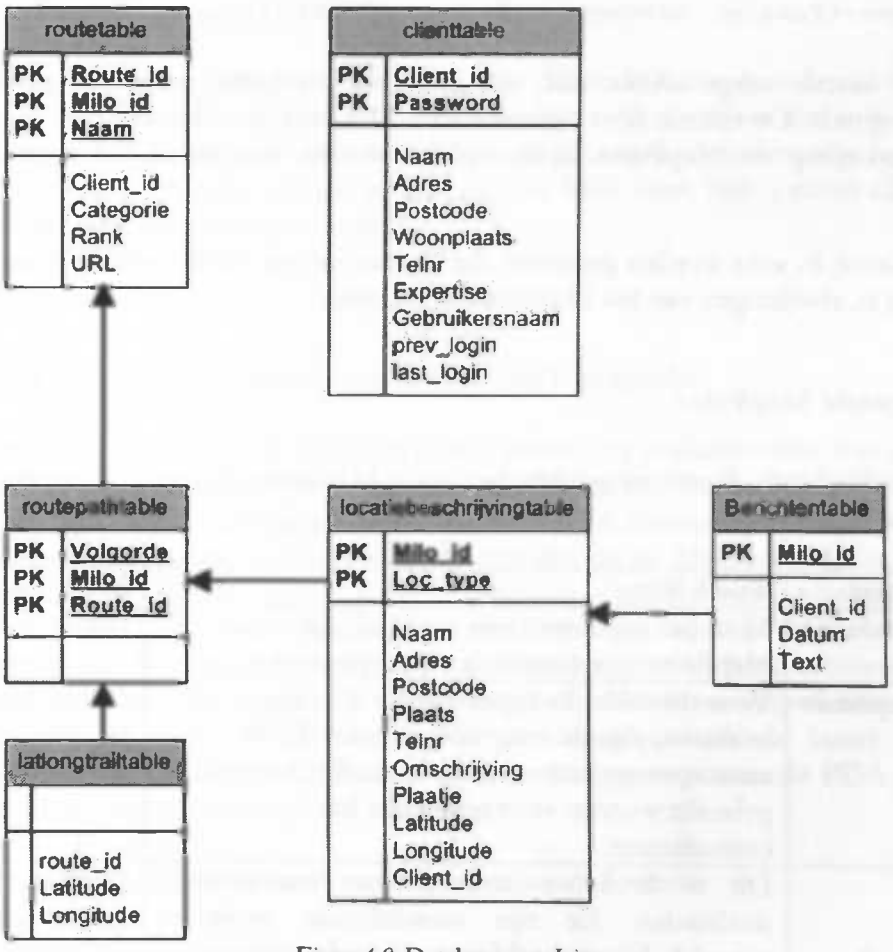
4.3. Data/Object model

In deze paragraaf wordt het datamodel en het objectmodel weergegeven en nader toegelicht. Eerst wordt een diagram van de database weergegeven en toegelicht. De database is geïmplementeerd in MySQL. Verder wordt een grafische representatie van de structuur van de objecten in *Lisa* behandeld.

4.3.1. De database van *Lisa*

De database van *Lisa* is in MySQL [W16], want deze DBMS is open source, en integratie in een .NET omgeving is vrij eenvoudig. Daarnaast wordt MySQL ook gebruikt voor *MiloWare*. Er kan daarmee tijd en moeite worden bespaard, omdat er veel kennis aanwezig is binnen TNO ICT.

Hieronder is een grafische representatie weergegeven van de database, welke is gekoppeld aan de serviceprovider van *Lisa*.



Figuur 4.9: Database van *Lisa*.

De database bevat alle gegevens over de locaties, de gebruikers, de routes. Ook onderhoudt de database de berichten die gekoppeld zijn aan de locatie. Er is geen relatie tussen de

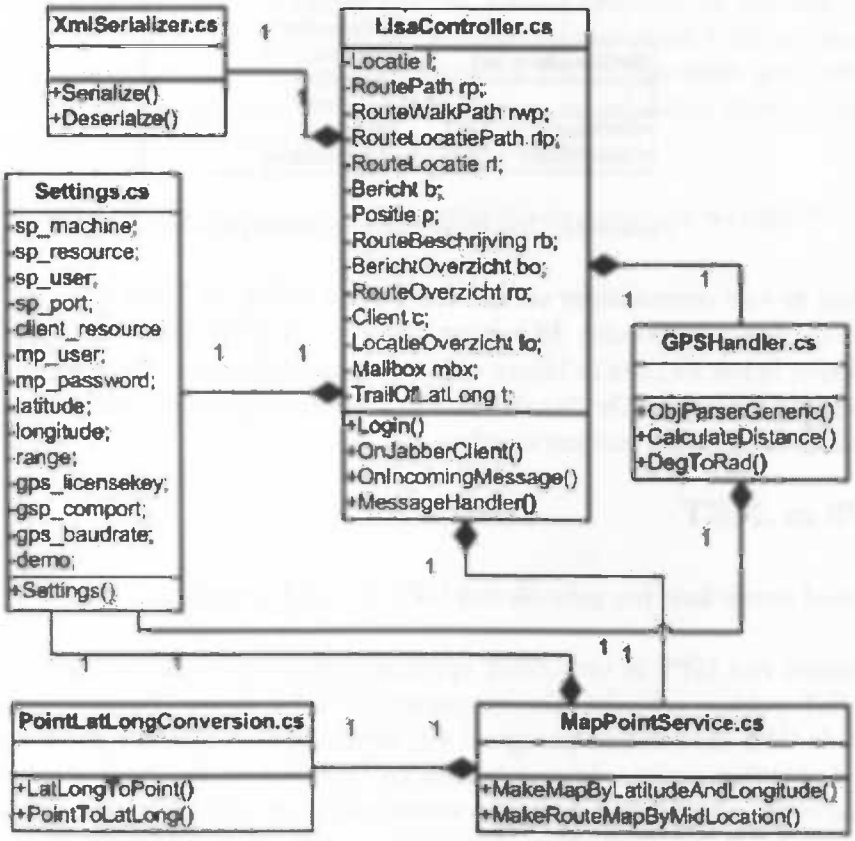
`clienttable` en de rest, want de cliëntgegevens worden enkel gebruikt bij het inloggen, en heeft verder geen rol bij het bewandelen en/of maken van routes.

In het schema zijn ook de *primary keys* (PK) opgenomen, deze voorziet de elementen in de tabellen van een unieke identificatie.

Er wordt in het `locatiebeschrijvingtable` onderscheid gemaakt tussen locaties en LisaID objecten. Vanwege deze constructie is het niet benodigd een aparte LisaID tabel aan te maken.

4.3.2. De objecten van *Lisa*

Hieronder zijn de diagrammen weergegeven van de objecten en de methoden die in *Lisa* worden gebruikt. Het betreft respectievelijk de diagrammen voor de cliënt op de PDA en de serviceprovider.



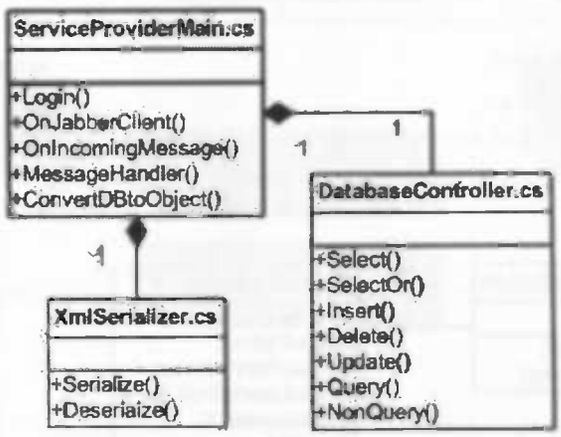
Figuur 4.10: UML diagram van de cliënt

Wat hier ontbreekt zijn de forms van de GUI, er zijn in totaal 24 schermen (inclusief de help-schermen). Daarover valt te vermelden dat aan elke form het object `LisaController` wordt gehangen, waardoor in elk form toegang is tot de objecten en de daarbijbehorende methoden. Immers aan de forms zijn acties gekoppeld, de gebruiker stuurt de gehele applicatie. De gebruiker maakt nieuwe locaties, nieuwe routes, nieuwe berichten, etcetera aan.

In het object `LisaController` zitten dus de messaging methoden, die het ontvangen en verzenden van XMPP berichten verzorgt. Aan dat object is ook het `XmlSerializer` object gehangen, deze verzorgt het serializen en het de-serializen van diverse objecten (bijvoorbeeld locatie, route, etcetera). Het `Settings` object is zo goed als overal bereikbaar, aangezien hierin

de belangrijkste instellingen zitten, bijvoorbeeld de baudrate van de com-poort waaraan de GPS ontvanger is gekoppeld, het wachtwoord van de MapPoint account, etcetera.

Aan LisaController is ook de `GPSHandler` en `MapPointService` gekoppeld, deze verzorgen het `GeoLocalization` component van de applicatie. Hierin zitten methoden om de latitudes en longitudes te verkrijgen, om kaarten te genereren.



Figuur 4.11: UML diagram van de serviceprovider

Dit diagram ziet er veel eenvoudiger uit dan die van de cliënt, dit komt door het ontbreken van de `Geolocalization` component. In het `ServiceProviderMain` object zitten de messaging methoden. Verder is het `XmlSerializer` object hieraan gehangen, deze verzorgt het serializen en de-serializen van objecten. De `DatabaseController` blijft over; hierin zitten alle operaties die op de database uitgevoerd kunnen worden.

4.4. GPS in .NET

In deze paragraaf wordt kort het gebruik van GPS in .NET toegelicht.

Voor de integratie van GPS in een .NET applicatie wordt een evaluatieversie van de Franson GPS tools .NET pakket gebruikt. De integratie van GPS in een PocketPC applicatie is zeer eenvoudig. In de GPS module (zie paragraaf 4.1) zitten NMEA listeners geïmplementeerd. Deze parseert dan de NMEA strings, hieruit komen de latitudes en de longitudes. Deze wordt dan lokaal opgeslagen. Bij elke NMEA sentence wordt nagegaan of er 50 meter is afgelegd. Is dat het geval, dan wordt de nieuwe latitude en longitude wederom opgeslagen in de `trail` array. Dit gebeurt in het geval dat de gebruiker een nieuwe route maakt. Maar als het een bewandeling van een route betreft, dan wordt de nieuwe latitude en longitude naar de serviceprovider verzonden. Daarbij wordt ook nagegaan of de verzonden positiegegevens in de buurt is van de meegezonden locatie. Is dat het geval, dan wordt de locatiegegevens naar de PDA verzonden. Op die manier wordt onnodig dataverkeer voorkomen.

De afstand wordt berekend aan de hand van de onderstaande formule:

$$D = \arccos(\sin(\frac{\pi}{180} * lat1) * \sin(\frac{\pi}{180} * lat2) + \cos(\frac{\pi}{180} * lat1) * \cos(\frac{\pi}{180} * lat2) * \cos(\frac{\pi}{180} * lon1 - \frac{\pi}{180} * lon2)) \quad [4.1]$$

Waarbij de waarden *lat*1, *lat*2, *lon*1 en *lon*2 respectievelijk de coördinaten van de twee punten zijn. Is *D* groter dan 50, dan worden de latitude en de longitude opgeslagen.

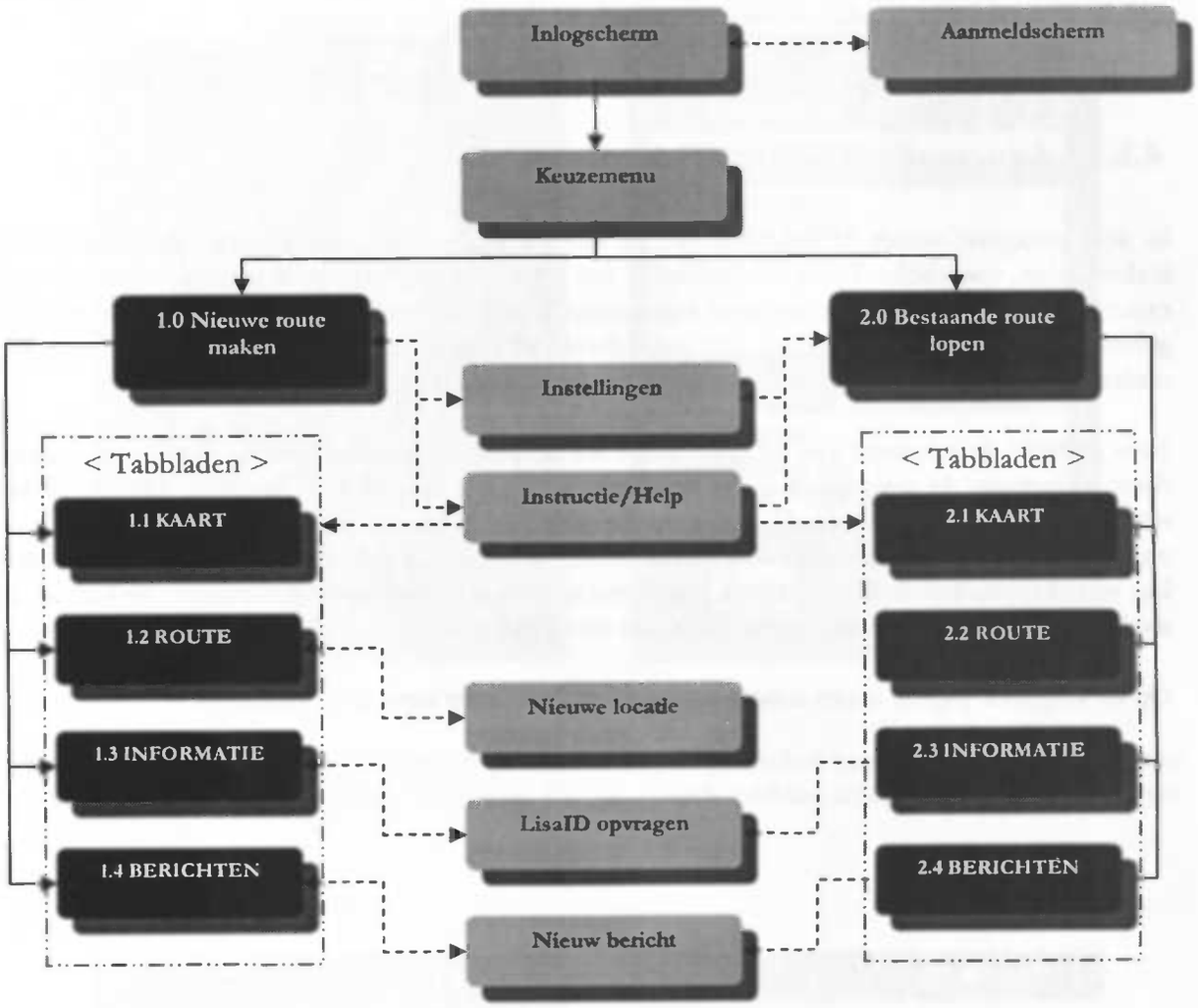
Voor de code van het bovenstaande wordt verwezen naar appendix VII.

4.5. Structuur van de applicatie

In deze paragraaf wordt de structuur van de applicatie, de schermen waarmee de gebruiker te maken krijgt, toegelicht. Dit is voornamelijk het werk geweest van Sven Schultz, hij heeft veel expertise op het gebied van cognitieve ergonomie. Zijn kennis wordt in dit project in de praktijk gebracht, ook heeft Sven Schultz een gebruikersonderzoek uitgevoerd. De resultaten van dat onderzoek is verwerkt in het uiteindelijke ontwerp, wat hieronder wordt gepresenteerd.

Er is getracht de structuur van de applicatie voor de gebruiker zo eenvoudig mogelijk te maken door gedurende de ontwikkeling de structuur zo lineair mogelijk te houden. Dat betekent, wanneer de gebruiker zich in de applicatie bevindt dat er maximaal 1 niveau naar beneden of naar boven kan worden genavigeerd. Het doel hiervan is dat de gebruiker qua oriëntatie niet snel kan en zal verdwalen in de schermen. Qua structuur weet de gebruiker direct waar hij zich in de applicatie bevindt en daarbij hoe hij verder en terug kan gaan.

Op de volgende pagina is een schematische weergave die de structuur weergeeft.



Er zijn 2 lijnen mogelijk binnen de applicatie:

- 1e lijn: Inloggen en 'Nieuwe route maken': + 1.x
- 2e lijn: Inloggen en 'Bestaande route lopen': + 2.x

Onderliggende niveaus (maximaal 1)

Figuur 4.12: Structuur van de applicatie.

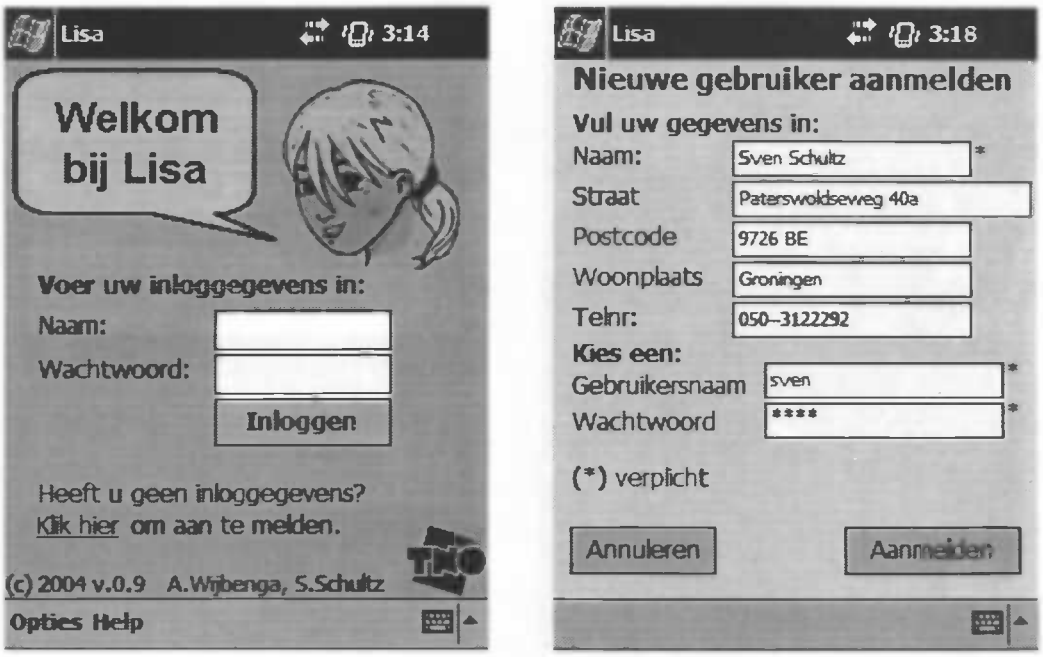
Als de gebruiker eenmaal is aangemeld, dan kunnen er zich twee scenario's voordoen: de gebruiker gaat een nieuwe route maken of de gebruiker gaat een bestaande route lopen. Dat zijn dus de twee lijnen. Na de keuze wordt de gebruiker naar de tabbladen gebracht, waarbij de gebruiker kan wisselen tussen de bladen: kaart, route, informatie en berichten.

Ook is er aandacht gegeven aan het instructiescherm. Deze is overal bereikbaar bij alle tabbladen in de vorm van een submenu onderin het scherm.

4.6. De Grafische User Interface (GUI)

In deze paragraaf wordt de GUI weergegeven in de vorm van screenshots. Zie daarvoor de volgende pagina's.

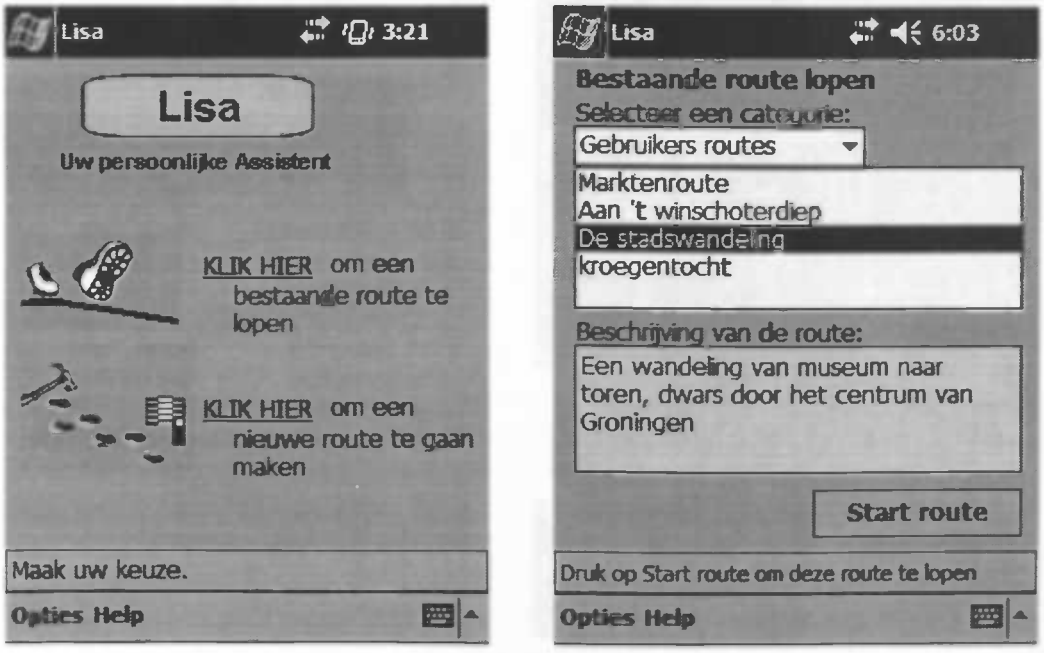
Inloggen en gebruikersregistratie



Figuur 4.13: Inloggen en gebruikersregistratie.

Bij het inloggen hoeft de gebruiker slechts zijn/haar gebruikersnaam en wachtwoord te invoeren. De gegevens komen overeen met de username en password van de Jabber server. Indien de gebruiker geen credentials heeft, kan hij/zij zich als nieuwe gebruiker aanmelden.

RouteLopen of RouteMaken en route selecteren



Figuur 4.14: RouteLopen of RouteMaken en route selecteren.

Zoals eerder in dit verslag is vermeld, wordt er onderscheid gemaakt tussen RouteLopen en RouteMaken. Dit komt tot uitdrukking in de eerste screenshot. In de tweede screenshot is er gekozen voor het bewandelen van een route. De geselecteerde route is: 'De stadswandeling'.

De kaart en de route

Figuur 4.15: De kaart en de route.

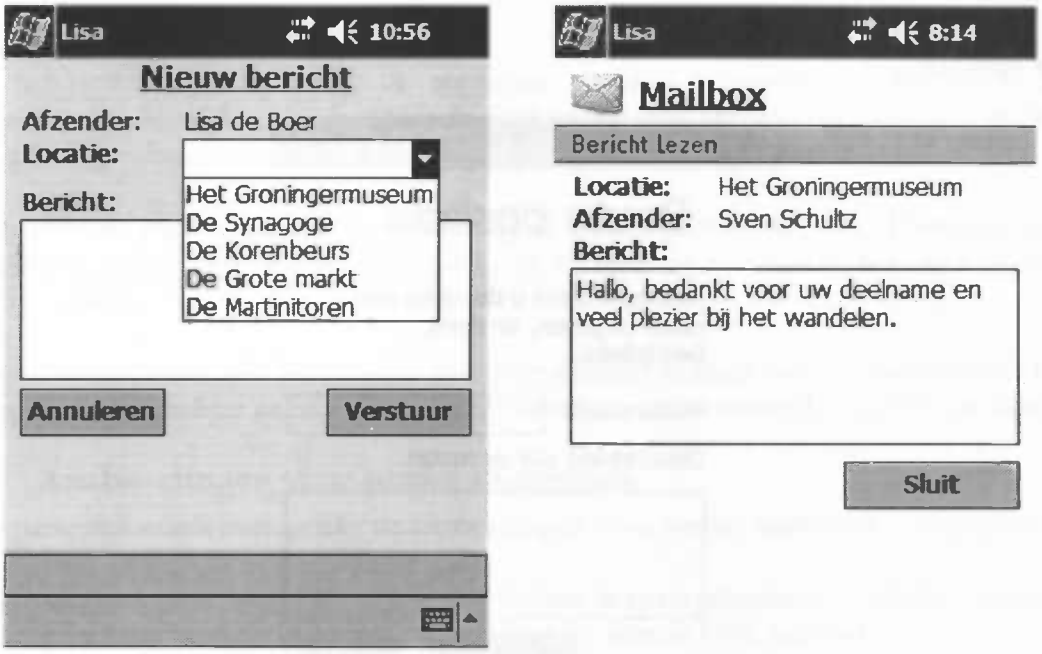
Op de kaart zijn de locaties zelf en het pad zichtbaar, de gebruiker kan aan de hand van de kaart zich oriënteren in een stedelijke omgeving. Op het tweede tabblad zitten de NAW gegevens van de gehele route.

De locaties en de mailbox

Figuur 4.16: De locaties en de mailbox.

In het tabblad 'Locaties' kan de gebruiker de informatie over de locaties lezen. Ook kan de gebruiker hier de LisaID invoeren. In het tabblad 'Mailbox' zitten de locatiegerelateerde berichten.

Nieuw bericht en een bericht lezen



Figuur 4.17: Nieuw bericht en een bericht lezen.

In de eerste screenshot wordt een nieuw bericht aangemaakt, de gebruiker kan dan de locatie selecteren waaraan het nieuwe bericht gerelateerd is. In het tweede screenschot wordt een bericht van Sven Schultz bij het Groninger Museum getoond.

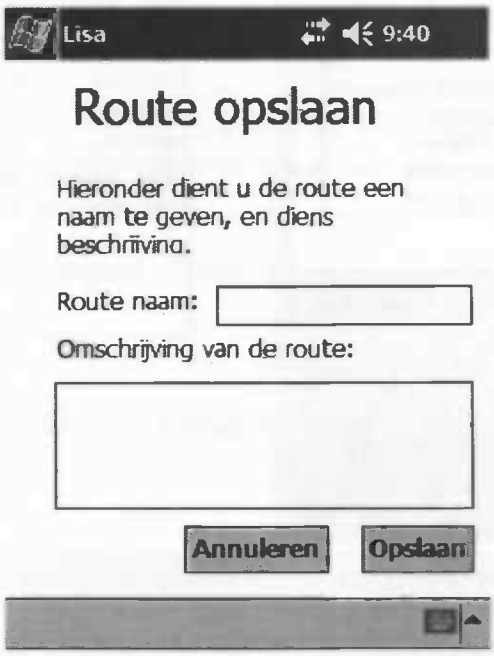
Bestaande of een nieuwe locatie

Figuur 4.18: Bestaande of een nieuwe locatie.

Bij het toevoegen van een nieuwe locatie aan een route kan de gebruiker kiezen uit bestaande locaties of een nieuwe locatie. Mocht de gebruiker een bestaande locatie willen, dan kan hij selecteren uit de listbox in de eerste screenshot. Bij een nieuwe locatie dient de gebruiker de

NAW gegevens en de omschrijving van dat locatie te invoeren, dit is te zien in de tweede screenshot.

Route opslaan



Figuur 4.19: Route opslaan.

Ten slotte wordt de gehele route opgeslagen, dit gebeurt in bovenstaande screenshot. De gebruiker dient de naam van de nieuwe route in te voeren en ook de daarbijbehorende omschrijving.

Hoofdstuk 5. Lisa en haar toekomst

Inleiding
Nu is het project *Lisa* voltooid, de applicatie werkt. De productontwikkeling is in de voorafgaande hoofdstukken uitgebreid uit de doeken gedaan. Ondanks het intensieve onderzoek en ontwikkeling vallen er altijd dingen te verbeteren of anders aan te pakken.

In dit hoofdstuk wordt ingegaan op eindconclusie van de probleemstelling. Daarnaast worden aanbevelingen gedaan voor verder onderzoek, ter verbetering van *Lisa* en het ontwikkelen van een *LB* applicatie.

Ook worden hier eventuele features die geïmplementeerd kunnen worden beschreven. Er zijn veel mogelijkheden, echter de projectuur en het aantal manuren waren de beperkende factoren.

5.1. Aanbevelingen voor verder onderzoek

In deze paragraaf wordt puntsgewijs de aanbevelingen voor verder onderzoek weergegeven.

- Onderzoek naar mogelijkheid(en) om Galileo te gaan gebruiken (in 2008). Galileo biedt hogere nauwkeurigheid en meer mogelijkheden dan de GPS techniek.
- Onderzoek naar in hoeverre AGPS daadwerkelijk mogelijk wordt in Nederland. Dit kan interessant zijn voor *LB* diensten op een GSM toestel, waardoor ontwikkelde applicaties breed ingezet kunnen worden.
- Onderzoek naar koppeling van een RFID reader op de PDA, of dat de PDA een 'tag' krijgt. Er is voorspeld dat RFID in de toekomst overal gebruikt zal worden.
- Onderzoek naar andere GIS dan MapPoint. Dit is echter niet noodzakelijk, aangezien TNO ICT momenteel MapPoint voor een jaar mag gebruiken (MSDN licentie).
- Onderzoek naar een dynamische XML serializer, dat arraylists kan serializen waardoor het dataverkeer gereduceerd kan worden. Dit brengt kostenbesparing met zich mee.
- Onderzoek naar WiFi lokalisatie in grote steden met een voldoende aantal hoeveelheid hotspots om lokalisatie te kunnen realiseren. Dit vanwege het groeiende aantal WiFi hotspots in de grote steden.
- Onderzoek naar open source GPS software, en het ontwikkelen ervan. TNO ICT is dan niet afhankelijk van COTS software.
- Onderzoek naar implementatie mogelijkheden van het XMPP protocol in Compact .NET. TNO ICT is dan niet afhankelijk van COTS software.
- Onderzoek naar mogelijkheden voor het verbeteren van de GUI. Het Compact Framework biedt niet veel mogelijkheden om de GUI te ontwerpen. Het schijnt dat Compact Framework versie 2.0 veel meer mogelijkheden biedt.
- In *MiloWare* de `XnearY` functie analyseren, en verbeteren.
- Onderzoek naar de scanning techniek: Near Field Communication (NFC). Deze techniek is de laatste maanden erg in opkomst. Ook zou onderzoek naar *LB* via Bluetooth aanbevelenswaardig zijn, aangezien Bluetooth een zeer gangbare techniek is.

5.2. Features voor Lisa in de toekomst

In deze paragraaf wordt puntsgewijs de toekomstige features voor *Lisa* gepresenteerd.

- Video on location: de gebruiker kan zelf video's maken, en deze aan de locatie koppelen. De route-bewandelaars kunnen dan deze video's bekijken. Dit geldt ook voor de 'foto on location' feature.

- Toevoeging van een Instant Messaging client, bijvoorbeeld een MSN client. Waardoor de gebruiker online kan chatten met andere gebruikers. Op de buddy-list is dan zichtbaar waar de bewandelaar zich bevindt.
- Meer instel mogelijkheden. De gebruiker kan met deze feature zelf bepalen of de berichten binnen mogen komen, en van wie ze afkomstig zijn, van welke datum, etcetera. De gebruiker zou meer selectiemogelijkheden moeten hebben, bijvoorbeeld het selecteren van routes op sleutelwoorden.
- Een 'binnenshuis'-module kan worden opgenomen, zodat *Lisa* zowel buiten en binnen te gebruiken is. Bijvoorbeeld een museumtour, buitenshuis wordt de route aan de hand van de GPS positie gepresenteerd, binnenshuis een rondleiding aan de hand van de WiFi lokalisatie.

Zo zijn er vele features te bedenken. Vooral de eerste twee kunnen van grote toegevoegde waarde zijn.

5.3. Conclusie

In deze paragraaf wordt ingegaan op de vervulling van de probleemstelling.

De probleemstelling luidt:

> *Op welke wijze kan een nieuwe* Location Based Service *applicatie op een Pocket PC (PPC) door TNO ICT zo optimaal mogelijk worden ontwikkeld, waarbij rekening gehouden wordt met de mogelijkheden en beperkingen van de bestaande technieken?*

Er wordt hierna gekeken naar de ontworpen architectuur, of het voldoet aan de gestelde functionele requirements.

Architectuur versus functionele requirements en de technieken

Uit analyse van de ontworpen architectuur en architecturen voor LBS applicaties in het algemeen [A2], blijkt dat de systemen uit overeenkomende componenten bestaan. De componenten zijn:

- Ontsluiting en weergave van locatie informatie:
  Een *location based* applicatie moet de locatie van het device kunnen ontsluiten, dat kan op verschillende manieren (zie hoofdstuk 2). Bij *Lisa* wordt de GPS techniek gebruikt. De architectuur voldoet daarmee aan de functionele requirement 1, *het moet een lokalisatie techniek bevatten.* De weergave van de locatie wordt verzorgd door MapPoint. MapPoint biedt de mogelijkheid om kaarten weer te geven aan de hand van een gegeven longitude en latitude. De architectuur voldoet ook daarmee aan functionele requirement 6, *het moet de beschikking hebben over geografische kaartgegevens.*
- Ondersteuning van heterogene devices:
  De architectuur moet zodanig zijn dat externe devices opgenomen kunnen worden in het systeem. In het geval van *Lisa* betreft het dus een scanning techniek (en ook de GPS ontvanger). Omdat de applicatie op een PDA draait, biedt de PDA veel uitbreidingsmogelijkheden. De in dit verslag beschreven architectuur vormt daartoe geen beperking. Dus de architectuur voldoet aan de functionele requirement 2, *het moet de beschikking hebben over een scan methode.*
- Een *LB* platform:
  Omdat de PDA geen voldoende rekenkracht en capaciteit (geheugen) biedt, is een platform noodzakelijk. Op dat platform kan worden bijgehouden waar de devices zich bevinden. Daarbij is een database onontbeerlijk. Bij *Lisa* wordt daarvoor het *MiloWare*

platform gebruikt. De ontworpen architectuur voldoet dus aan de requirements 5 en 7, *beveiliging in de vorm van gebruikers authenticatie is gewenst* en *de locaties en de routes moeten 'ergens' worden opgeslagen.*

- Communicatie methode:
  Omdat een *LB* applicatie vaak mobiel is, is een communicatie methode met het platform noodzakelijk. De infrastructuur is reeds aanwezig in de vorm van de UMTS/GPRS netwerk. Echter de keuze voor het protocol dient overwogen te nemen. Voor *Lisa* is de keuze op het XMPP protocol gevallen, zie hoofdstuk 3. De architectuur vormt dus geen beperking voor de requirements 3 en 4, *de PDA moet draadloos kunnen communiceren met het LBS platform* en *de informatie moet binnen een acceptabele tijd over en weer worden verzonden tussen de cliënt en de server.*

De architectuur komt de functionele requirements tegemoet. En vanwege de overeenkomstigheid met andere architecturen is de architectuur van *Lisa* reproduceerbaar. Echter indien een andere lokalisatie methode wenselijk is, dan dient bij de architectuur rekening gehouden te worden met of de positie cliënt-sided of via het netwerk wordt ontsloten.

In dit verslag is het onderzoek naar de verschillende lokalisatie en communicatie methoden uitvoerig beschreven. Momenteel biedt het GPS systeem de meest nauwkeurige lokalisatie en het is meest geschikt als lokalisatie techniek voor de PDA, Pocket PC systeem. Voor de communicatie is het XMPP protocol om verschillende redenen gekozen, echter de webservice methode is ook een goed alternatief.

Conclusie
Om een LBS applicatie voor op de PDA te ontwikkelen kan de ontworpen architectuur worden gehanteerd.

Daarbij worden de volgende technieken gebruikt:

- GPS als lokalisatie methode.
- Een serviceprovider en *MiloWare* als *LB* platform.
- Het XMPP protocol voor de communicatie.
- Microsoft MapPoint voor de geografische kaartgegevens.

Voor TNO ICT in de toekomst kan het project *Lisa* als referentiekader worden gebruikt. De ontworpen architectuur kan als richtlijn worden gebruikt bij toekomstige LBS applicaties. Ook kan met behulp van dit verslag in korte tijd worden nagegaan welke lokalisatie technieken om handen zijn. De bron code is zo modulair mogelijk gestructureerd, zodat TNO ICT (deel) componenten van de code kan gebruiken voor toekomstige applicaties.

Tenslotte is de applicatie *Lisa* uitermate geschikt om opgenomen te worden in het portfolio van TNO ICT, aangezien *Lisa* reeds enige bekendheid heeft opgebouwd in de marketing wereld van Groningen (Groninger Museum, V.V.V.) en bij Surfnet. Tevens biedt *Lisa* de mogelijkheid om een echte *Location Based Service* te demonstreren. *Lisa* is als het ware een innovatieve kaleidoscoop van verschillende technieken en features: *location based* informatie, *location based* messaging, *location based* navigatie, webservices, Jabber protocol, "RFID", UMTS, GPRS, GPS, etcetera.

Gebruikte afkortingen
(in willekeurige volgorde)

ECEF: *Earth-Centered-Earth-Fixed*

WiFi: *Wireless Fidelity*

UMTS: *Universal Mobile Telecommunication System*

GPRS: *General Packet Radio Service*

RFID: *Radio Frequency Identification*

XMPP: *Extensible Messaging and Presence Protocol*

GPS: *Global Positioning System*

AGPS: *Assisted Global Positioning System*

AoA: *Angle of Arrival*

TDoA: *Time Difference of Arrival*

ToA: *Time of Arrival*

PDA: *Personal Digital Assistent*

NMEA: *National Marine Electronics Association*

GSM: *Global System for Mobile Communications*

2G/3G: 2nd Generation / 3rd Generation

WLAN: *Wireless Local Area Network*

SOAP: *Simple Object Access Protocol*

LB(S): *Location Based (Services)*

COTS: *Commercial off-the-shelf.*

OSI: *Open Systems Interconnection*

IRC: *Internet Relay Chat*

SMS: *Short Message Service*

SMTP: *Simple Mail Transfer Protocol*

AIM: *AOL Instant Messenger*

ICQ: *I Seek You*

MSN: *Microsoft Network*

LISA: *Location based Informatie Systeem Assitent*

UDDI: *Universal Description Discovery and Integration*

SOAP: *Simple Object Access Protocol*

XML: *Extensible Markup Language*

WSDL: *Web Services Description Language*

WSFL: *Web Services Flow Language*

NAW: *Naam Adres en Woonplaats*

GIS: *Geographic Information System*

DBMS: *DataBase Management System*

Geraadpleegde literatuur- en bronnenlijst

Websites:

- [W1]: *TNO Telecom*, http://www.telecom.tno.nl/, februari 2004.
- [W2]: *Extreme Programming*, http://extremeprogramming.org/, september 2004.
- [W3]: *Wikipedia*, http://nl.wikipedia.org/wiki/hoofdpagina, oktober 2004.
- [W4]: *Microsoft*, http://www.microsoft.com/, september 2004.
- [W5]: *Galileo*, http://europa.eu.int/comm/dgs/energy_transport/galileo/intro/index_en.htm, oktober 2004.
- [W6]: *Codeproject.com*, http://www.codeproject.com/, december 2004.
- [W7]: *Global Positioning System: Advanced Topics in Mathematics at CHSS*, http://mail.colonial.net/~abeckwith/gps.html, februari 2005.
- [W8]: *Ekahau Inc.*, http://www.ekahau.com, februari 2005.
- [W9]: *Vereniging voor Statistiek en Onderzoek (VOS)*, http://www.vsonet.nl/index.php?id=2788&catid=2, november 2004.
- [W10]: *Radius Server*, http://www.open.com.au/radiator/.
- [W11]: *GSM World*, http://www.gsmworld.com/technology/gprs/intro.shtml, november 2004.
- [W12]: *Research Activities: UMTS*, http://www.nt.tuwien.ac.at/mobile/projects/UMTS/en/, februari 2005.
- [W13]: *International Engineering Consortium*, http://www.iec.org/online/tutorials/umts/topic01.html, februari 2005.
- [W14]: *Jabber.org (XMPP)*, http://www.jabber.org/ietf/attic/draft-ietf-xmpp-im-04.html, november 2004.
- [W15]: *Microsoft MapPoint*, http://www.microsoft.com/mappoint, oktober 2004.
- [W16]: *MySQL*, http://www.mysql.org, november 2004.
- [W17]: *Franson GPStools .NET*, http://franson.com/gpstools/, december 2004.
- [W18]: *SIP center*, http://www.sipcenter.com, februari 2004.

Literatuur:

- [L1]: *Software Engineering, 6th Edition*, Ian Sommerville, 2001 Addison Wesley, ISBN 0-201-39815-X.
- [L2]: *Design and Use of Software Architectures*, Jan Bosch, 2000 Addison Wesley, ISBN 0-201-67494-7.
- [L3]: *Professional C#, 2nd Edition*, Simon Robinson, K. Scott Allen, Ollie Cornes, Jay Glynn, Zach Greenvoss, Burton Harvey, Christian Nagel, Morgan Skinner and Karli Watson, 2002 Wrox Press, ISBN 1-86100704-3.
- [L4]: *.NET Enterprise Development in C#: From Design to Deployment*, Matthew Reynolds and Karli Watson, 2002 Wrox Press, ISBN 1-86100591-1.
- [L5]: *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification Second Edition*, Klaus Finkenzeller, 2003 Wiley, ISBN 0-470-84402-7.
- [L6]: *GPS: Theory and Practice, Fifth revised edition*, B. Hofmann-Wellenhof, H. Lichtenegger and J. Collins, 2000 Springer-Verlag/Wien New York, ISBN 3-211-83534-2.
- [L7]: *GPRS demystified*, John Hoffman, 2003 McGraw-Hill TELECOM, ISBN 0-07-138553-3.

- [L8]: *UMTS the Fundamentals*, B. Walke, P. Seidenberg, M. P. Althoff, 2003 Wiley, ISBN 0-470-84557-0.
- [L9]: *Networks: Internet • Telephony • Multimedia, Convergences and complementarities*, Daniel Hardy, Guy Malléus, Jean-Noël Méreur, 2002 Springer-De Boeck, ISBN 3-540-00559-5.
- [L10]: *Business Data Communications 4th Edtion*, William Stallings, 2001 Prentice Hall, ISBN 0-13-088263-1.
- [L11]: *World Wide Wi-Fi, Technological Trends and Business Strategies*, Teik-Kheong, Benny Bing, 2003 Wiley-Interscience, ISBN 0-471-46356-6.
- [L12]: *Distributed Systems: Principles and Paradigms*, Andrew S. Tanenbaum, Maarten van Steen, 2002 Prentice-Hall, ISBN 0-13-088893-1.
- [L13]: *Wireless Communications and Networks*, William Stallings, 2002 Prentice Hall, ISBN 0-13-040864-6.

Artikelen en papers:

- [A1]: *An Architecture for User Location in Heterogeneous Mobile Networks*, Maarten Wegdam, Jeroen van Bemmel, Ko Lagerberg and Peter Leijdekkers, 2004 Proceedings of the 7th IEEE International Conference on High Speed Networks and Multimedia Communications Vol. 3079, ISBN 3-540-22262-6. http://research.it.uts.edu.au/arn/files/papers/hsnmc04userlocationwegdam.pdf.
- [A2]: *An Architecture for Location-based Service Mobility Using the SIP Event Model*, Ron Shacham, Henning Schulzrinne, Wolfgang Kellerer, 2004 Department of Computer Science, Columbia University New York, http://www.sigmobile.org/mobisys/2004/context_awareness/papers/mob_paper2.pdf.
- [A3]: *Localizability Analysis for GPS/Galileo Receiver Autonomous Integrity Monitoring*, Steve Hewitson, Hung Kyu Lee and Jinling Wang, 2004 Journal of Navigation, 57:245-259 Cambridge University Press. http://www.gmat.unsw.edu.au/snap/publications/hewitson_etal2004a.pdf.
- [A4]: *GPS primer, a student guide to the Global Positioning System*, 2003 The Aerospace Corporation, http://www.aero.org/education/primers/gps/GPS-Primer.pdf.
- [A5]: *Integrated Coastal Hydrography (Invest to Save Budget) Project, Emerging Technologies Report*, 2004 Integrated Coastal Hydrography, http://www.coastalhydrography.com/ICH%20Emerging%20Technologies%20Final%20Report%20(26%20March%202004).pdf.
- [A6]: *The Galilei Project, GALILEO Design Consolidation*, 2003 Galilei, http://europa.eu.int/comm/dgs/energy_transport/galileo/doc/galilei_brochure.pdf.
- [A7]: *Standardization of Mobile Phone Positioning for 3G Systems*, Yilin Zhao (Motorola Inc.), 2002 IEEE Communications Magazine, http://www.comsoc.org/livepubs/cil/public/preview/pdf/zhao.pdf.
- [A8]: *Cell-ID location technique, limits and benefits: an experimental study*, Emiliano Trevisani, Andrea Vitaletti, 2003 Università di Roma "La Sapienza", http://wmcsa2004.lancs.ac.uk/Slides/wmcsa2004-06-trevisani-cellidlocation.pdf.
- [A9]: *Positioning of Mobile Devices*, Ng Ping Chung, 2003 Oresund Summer University Geo-Information System *Project Report*, http://wmcsa2004.lancs.ac.uk/Slides/wmcsa2004-06-trevisani-cellidlocation.pdf.

- [A10]: *Mobile Positioning vs. Privacy*, Mikko Heikkinen, 2004 Helsinki University of Technology, http://www.netlab.hut.fi/opetus/s38042/s04/Presentations/02112004 Heikkinen/Heikkinen paper.pdf.
- [A11]: *A wireless LAN-based indoor positioning technology*, Z. Xiang, S. Song, J. Chen, H. Wang, J. Huang, X. Gao, 2004 IBM Corp., http://www.research.ibm.com/journal/rd/485/xiang.pdf.
- [A12]: *GPRS/EDGE presentation*, Karam Jinane, Peignot Estelle, Abou Aun Mayssam, 2002 DESS R2M, http://dessr2m.adm-eu.uvsq.fr/gprs-edge.pdf.
- [A13]: *Wireless Presence and Instant Messaging*, Yanna Vogiazou, 2002 The Open University (UK), http://www.jisc.ac.uk/uploaded documents/tsw 02-07.pdf.
- [A14]: *Extensible Messaging and Presence Protocol (XMPP)*, Mikko Laukkanen, 2004 University of Helsinki, http://www.cs.helsinki.fi/u/kraatika/Courses/IPsem04s/xmpp.pdf
- [A15]: *Architectural Considerations for Presence and Instant Messaging Infrastructure: Comparing XMPP and SIP/SIMPLE*, Jabber Inc., 2004 Jabber Inc., http://www.jabber.com/index.cgi?CONTENT ID=55.
- [A16]: *A definition of Web Services*, Simon Watson, Graham Dunlop, 2002 Intercea Inc., http://www.intercea.co.uk.

Appendices

Appendix I: Opdrachtomschrijving TNO ICT en projectorganisatie

Afstudeeropdracht TNO ICT

LBS pocket PC applicatie

Titel stage/afstudeeropdracht: LBS pocket PC applicatie
Duur van stage/afstuderen: 6 Maanden
Manager: Evert v/d Akker
Begeleider: Karen Groenink, Arnoud de Jong/Henk Enting
Locatie: Groningen

INLEIDING
TNO Telecom is een Research & Development Center op het gebied van telecommunicatie en IT, met belangrijke klanten als o.a. KPN. Het heeft als uitdaging om dié producten, diensten en processen te ontwikkelen die een substantiële bijdrage leveren aan het resultaat van deze bedrijven. Met een sterke mix aan hoogwaardige kennis en kunde op technologisch, sociaal-cultureel, en bedrijfseconomisch gebied, is TNO Telecom in staat om zowel nieuwe dienstconcepten te ontwikkelen alsook deze vooraf te toetsen op klantwaarde en economische haalbaarheid. Binnen TNO Telecom wordt op dit moment veel onderzoek gedaan naar Location Based Services (positie gebaseerde) diensten. Inmiddels is er een platform ontwikkeld waarmee LBS diensten kunnen worden aangeboden. LBS diensten zijn natuurlijk bij uitstek geschikt om aan te bieden op een mobiel device, zoals een Pocket PC of mobiele telefoon. Lokalisatie gebeurt in het specifieke geval van ons platform via het WiFi netwerk.

Een ontwikkelomgeving waarmee binnen TNO Telecom veel gewerkt wordt is .Net. Ook het LBS platform is hierin ontwikkeld. .Net biedt ook een lokalisatie server, Mappoint. Met Mappoint is het mogelijk om geografische data grafisch weer te geven.

OPDRACHTOMSCHRIJVING
Ontwerp en ontwikkel een LBS dienst in .NET die gebruik maakt van het LBS-platform. De dienst moet gaan draaien op een Pocket PC.
Aan de ene kant communiceert de dienst met het LBS platform. Het platform voorziet de cliënt van locatie-informatie, bijvoorbeeld waar iemand zich bevindt en meer. Daarnaast communiceert de applicatie met de Mappoint server. In beide gevallen gebeurt de communicatie via webservices.
Via het ontwerpproces, dat zal bestaan uit het maken van een functioneel ontwerp, grafisch ontwerp en objectmodel, zal een uiteindelijke keuze gemaakt worden wat er definitief gebouwd zal gaan worden.

De dienst zal aan een aantal eisen moeten voldoen:
- Heldere en gebruiksvriendelijke user interface.
- Gemakkelijk uitbreidbaar met nieuwe diensten.
- Veilig; gebruikers willen immers niet dat hun locatiegegevens op straat komen te liggen.
- Bestaat voor een deel uit herbruikbare (generieke) modules, zoals:
  - Communicatiemodule
  - Securitymodule
  - Kaartmodule

- ...

De applicatie zal de volgende diensten moeten kunnen aanbieden:
- Bericht op locatie achterlaten voor een persoon of groep.
- Locatie vinden van bijv. een restaurant.
- ...

Opgeleverd wordt een werkende applicatie op een pocket pc, die voldoet aan bovenstaande eisen. Daar hoort de volgende documentatie bij:
- Functioneel ontwerp.
- Grafisch ontwerp.
- Data/Object model
- Code.

WIE ZOEKEN WIJ?
We zoeken twee mensen met elk hun eigen specialiteit:
- Iemand die het ontwerp van de User Interface voor zijn rekening neemt; hierbij hoort natuurlijk ook de gebruiksvriendelijkheid van de applicatie.
- Daarnaast willen we graag een technisch iemand met programmeerervaring (liefst .Net), die de applicatie zal bouwen.

Vanzelfsprekend is dat beiden nauw met elkaar samenwerken.
We verwachten dat je pragmatisch ingesteld bent, initiatief neemt en zowel zelfstandig als in een team kunt werken. Ook is het van belang dat je jouw kennis kunt overbrengen op technische en niet-technische collega's en je kunt je inleven in de business kansen van deze dienst.

WAT BIEDEN WE?
Je komt te werken in een jonge innovatieve omgeving met zowel TNO Telecom collega's als andere studenten. Wij staan open voor jouw ideeën en initiatieven als bijdrage in een operationeel project. En wellicht kunnen we je na afloop verwelkomen als nieuwe collega!

Oplevering

Het volgende zal worden opgeleverd aan het eind van de afstudeerperiode, naast dit afstudeerverslag.

1. Afstudeerverslag met daarin (o.a.):
   a. Verslag literatuuronderzoek naar de methoden, kenmerken en toepasbaarheid van lokalisatie/communicatie technieken.
   b. Dienstomschrijving, uitwerking, ontwerpdocumentatie.
      De documentatie zal bestaan uit:
      i. Beschrijving van gebruiker en functionele requirements.
      ii. Omschrijving rapid prototyping.
      iii. Functioneel ontwerp.
      iv. Architectuur.
      v. Data/object model.
   c. Omschrijving Microsoft MapPoint Webservices.
   d. Aanbevelingen voor toekomstig onderzoek en ontwikkeling.
2. De broncode (op CD-ROM).
3. Gecompileerde code, draaiend op een PPC, installatie CD plus de gebruikshandleidingen en een installatiehandleiding.

Projectorganisatie en rolverdeling

Communicatie

Zowel Albert Wijbenga en Sven Schultz zullen hun werk documenteren en deze naar elkaar toe ter beschikking stellen. Tijdens de wekelijkse gesprekken worden de rollen van voorzitter en notulist afwisselend vervuld.

Sven Schultz en Albert Wijbenga zullen elkaar zo veel mogelijk op de hoogte houden van alle informatie die hun ter ore is gekomen, dit ook in verband met de ontwikkelmethode. Daaronder wordt o.a. verstaan de feedback van de afstudeerbegeleiders, ontdekkingen die van belang zijn voor het te ontwikkelen product.

Research, ontwerp en ontwikkeling

De verantwoording van het technische aspect van de ontwikkeling ligt geheel bij Albert Wijbenga, de verantwoording van het ergonomische aspect ligt geheel bij Sven Schultz.

Albert Wijbenga en Sven Schultz zullen elkaar zo veel mogelijk op de hoogte houden van onderzoeksresultaten. Daaronder wordt o.a. verstaan de marktonderzoeken, de gebruikersonderzoeken, store-checks, interviews, literatuurstudie, en try-outs.

Vanwege het multidisciplinaire aspect in het team wordt als ontwikkelmethode eXtreme Programming gehanteerd. Deze methode is geschikt voor applicatieontwikkeling in kleine en multidisciplinaire teams. Voor een schematische weergave hiervan zie appendix II.

Het zwaartepunt van het gebruikersonderzoek ligt voornamelijk bij Sven Schultz. Albert Wijbenga zal hierbij technische ondersteuning verlenen, in de vorm van installatie van prototypes die als subject worden gehanteerd bij het onderzoeken. Daarnaast zal Albert Wijbenga naar aanleiding van de resultaten de nieuwe requirements implementeren in de prototypes.

Appendix II: Ontwikkelmethode

In de ontwerpfase werd gezocht naar een geschikte ontwerpmethode voor het ontwikkelen van een Pocket PC applicatie. Na deskresearch en oriënterende gesprekken met software ontwikkelaars binnen TNO is besloten dat *Extreme Programming* de meest geschikte software ontwikkelmethode voor de opdracht is. Deze techniek speciaal voor de ontwikkeling van software in een klein team bestaande uit mensen met verschillende disciplines. Onderlinge communicatie over voortgang speelt een belangrijke rol, doordat de functionaliteiten in een korte doorlooptijd worden ontwikkeld. Dit lijkt op *rapid prototyping*, echter hierbij wordt in mindere mate rekening gehouden met de verschillende disciplines die aanwezig zijn in een team.

Deze methode, ook wel 'XP' genoemd is ontwikkeld door Kent Beck. De optimale kracht van XP komt voort uit het in samenhang toepassen van twaalf 'best practices', doordat deze elkaar versterken.

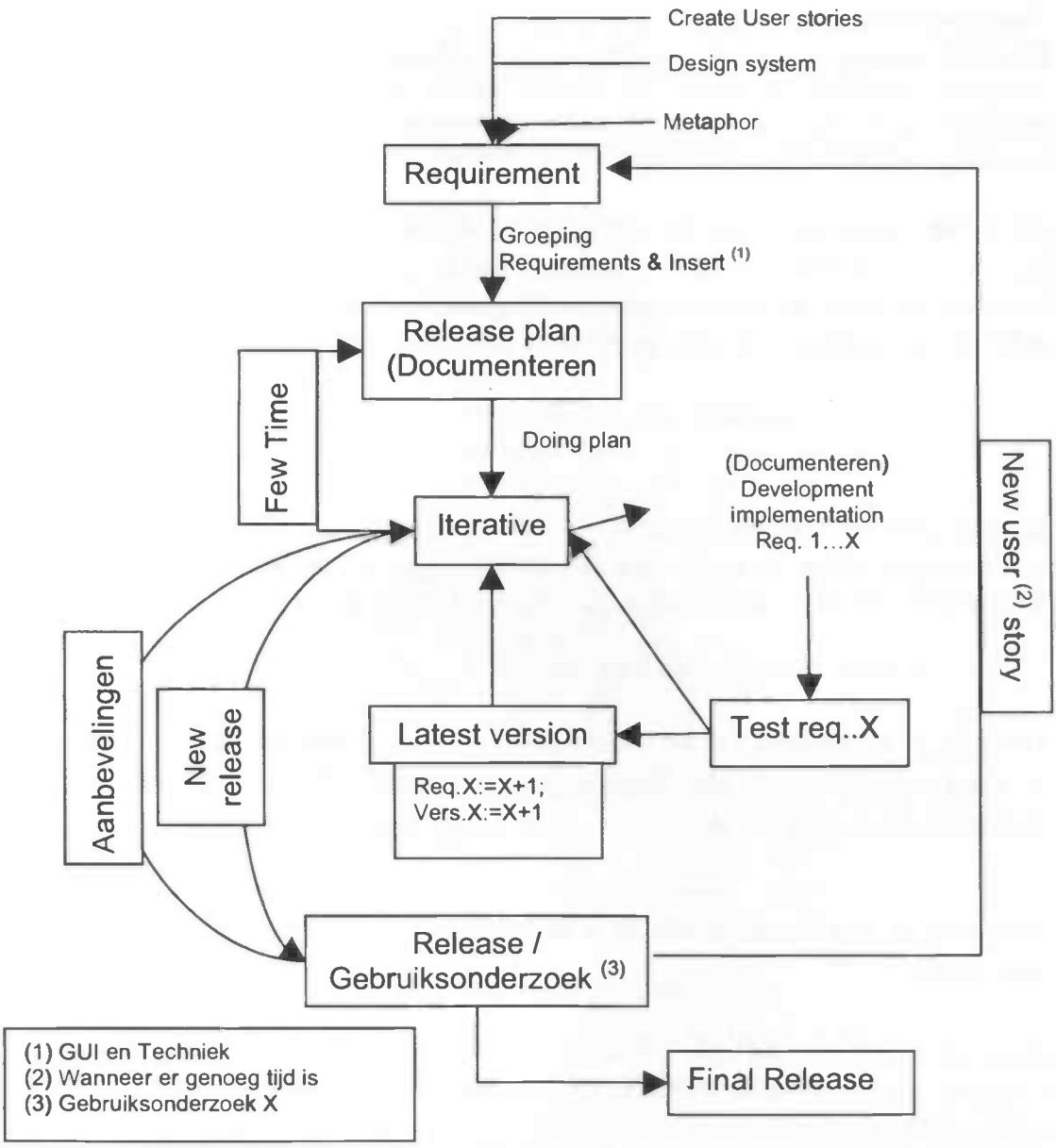
XP is gebaseerd op de volgende twaalf '*best practices*':

1. Gebruik een zo simpel mogelijk ontwerp dat werkt.
2. Schrijf eerst de test code ('unit test') voordat je functionaliteit codeert.
3. Continue herstructurering ('*refactoring*') van de programmacode.
4. Continue integratie van alle programmacode.
5. Iedere ontwikkelaar heeft gelijke rechten over alle programmacode.
6. Iedere ontwikkelaar gebruikt dezelfde codeerstandaard.
7. Programmeurs werken in paren ('*pair programming*').
8. De klant is onderdeel van het ontwikkelteam en moet dus continue voor vragen beschikbaar zijn.
9. De software wordt in een vaste regelmaat in releases van beperkte omvang aan de klant opgeleverd voor beoordeling.
10. Ontwikkelaars plannen aan de hand van kleine brokjes functionaliteit ('*user stories*').
11. Alle teamleden (ontwikkelaars en de klant) delen een gemeenschappelijk beeld van het systeem ('*methaphor*').
12. Overwerken is een uitzondering.

Tijdens de ontwikkelfase wordt getracht zo analoog mogelijk aan het XP principe te werken, achteraf is in de praktijk gebleken dat dit zeer goed werkt. Toch dient hier opgemerkt te worden dat vele zaken mondeling zijn beslist.

Na de ontwerpfase begon het eXtreme Programmeren. Stap voor stap, door middel van korte iteraties, zijn alle functionaliteiten langzaam in de dienst opgenomen. Eerst werden de verschillende user interfaces ontwikkeld, zodat er een bepaalde framework ontstond. Vervolgens zijn er functies gekoppeld aan de interfaces, zodat er interactie tussen de gebruiker en het systeem ontstond. Bij de ontwikkeling van de applicatie is zeer nauw samengewerkt tussen de twee afstudeerders. Het resultaat van de een van de basis voor de andere, een minpunt hierbij is dat het team erg afhankelijk van de voortgang van elkaar wordt. Gelukkig heeft dit geen consequenties gehad bij de uitvoering van deze opdracht.

Op de volgende pagina is een overzicht opgenomen waarbij in een (aangepaste) schematische weergave het Extreme Programming proces is uitgewerkt.



Figuur I.1: Extreme Programming

Naast XP zijn er nog een aantal ontwikkelmethode maar deze waren voor ons minder geschikt. Voor verdere informatie over deze methode wordt verwezen naar [W2].

Appendix III: GPS doorgelicht

Code pseudorange

Het GPS concept houdt onder andere in dat de observaties (van de ontvanger) niet exact zijn, d.w.z. ze bestrijken een range, het interval tussen het tijdsverschil van de klokken van de satellieten en de receiver. Dit wordt nader '*pseudorange*' genoemd. De ranges zijn dus de '*biases*' van klokverschillen tussen de satelliet en de ontvanger.

Zij t^S het emissietijdstip van het signaal vanuit de satelliet, en t_R het ontvangsttijdstip van het signaal bij de receiver. De tijdsvertragingen van de satellietklok en de klok van de ontvanger, waar de GPS tijd als uitgangspunt wordt gehanteerd, zijn respectievelijk δ^S en δ_R. Het tijdsverschil is gelijk aan Δt. Dus geldt de vergelijking,

$$\begin{aligned}\Delta t = t_R - t^S &= [t_R(\text{GPS}) + \delta_R] - [t^S(\text{GPS}) + \delta^S] \\ &= \Delta t(\text{GPS}) + \Delta\delta.\end{aligned} \qquad \text{[III.1]}$$

In [2.0] geldt dat $\Delta t(\text{GPS}) = t_R(\text{GPS}) - t^S(\text{GPS})$ en $\Delta\delta = \delta_R - \delta^S$. De term $\Delta\delta$ is gelijk aan de klokvertraging tussen de satelliet en de ontvanger. Als het tijdsverschil Δt vermenigvuldigd is met de lichtsnelheid c, dan verkrijgt men de pseudorange R, dus

$$R = c\Delta t = c\Delta t(\text{GPS}) + c\Delta\delta = \rho + c\Delta\delta. \qquad \text{[III.2]}$$

De range ρ is berekend van de werkelijke reistijd van de signaalemissie. Met andere woorden, ρ komt overeen met de afstand tussen de positie van de satelliet op tijdstip $t^S(\text{GPS})$ en de positie van de antenne van de receiver op het tijdstip $t_R(\text{GPS})$.

Aangezien ρ een functie is met twee verschillende variabelen, wordt ρ omgevormd in een Taylor serie;

$$\begin{aligned}\rho = \rho(t^S, t_R) &= \rho(t^S, (t^S + \Delta t)) \\ &= \rho(t^S) + \upsilon(t^S)\Delta t.\end{aligned} \qquad \text{[III.3]}$$

Waar υ de tijdsafgeleide van ρ is, of de radiale snelheid van de satelliet (relatief ten opzichte van de ontvanger). Alle gebruikte tijdstippen in [III.3] zijn expressies in GPS systeemtijd.

De GPS satellieten zijn 20.000 km. hoog van de Aarde gepositioneerd, en de signalen worden om de milliseconde verzonden. De maximale radiale snelheid van de GPS satellieten is $\upsilon \approx 0{,}9\,\text{km}\,\text{s}^{-1}$, en de reistijd van het signaal van de satelliet is ongeveer 0,07 s. De afwijking kan, volgens [III.3], dus tot 60 m. bedragen. Echter de laatste GPS ontwikkelingen hebben geleid tot een nauwkeurigheid van 3 m. tot 0,3 m. afwijking.

Locatie positionering met code pseudorange

Op een bepaald tijdstip t kan de pseudorange worden gemodelleerd naar:

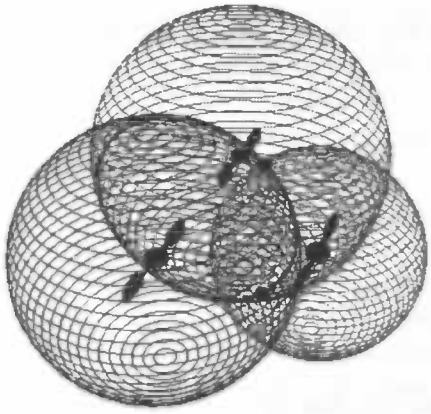
$$R_i^j(t) = \rho_i^j(t) + c\Delta\delta_i^j(t). \qquad \text{[III.4]}$$

Hier is $R_i^j(t)$ de berekende coderange tussen het observatiepunt i en de satelliet j, $\rho_i^j(t)$ is de geometrische afstand tussen de satelliet en het observatiepunt, en c is de lichtsnelheid. De laatste component, $\Delta\delta_i^j(t)$ is de klokbias welke de gecombineerde klok offsets van de receiver en de satelliet klokken representeert, waarbij de GPS tijd als uitgangspunt wordt gehanteerd.

De afstand zit impliciet in $\rho_i^j(t)$, in vgl. [2.3], en kan expliciet worden genoteerd als

$$\rho_i^j(t) = \sqrt{(X^j(t) - X_i)^2 + (Y^j(t) - Y_i)^2 + (Z^j(t) - Z_i)^2}. \qquad \text{[III.5]}$$

Waar $X^j(t)$, $Y^j(t)$ en $Z^j(t)$ de coördinaten zijn van de geocentrische positie van de satelliet op tijdstip t, en X_i, Y_i en Z_i zijn de drie ECEF coördinaten van het observatiepunt i (zie Figuur 2.1).



Figuur 2.1: Het snijpunt van de 4 vlakken levert de coördinaten van de positie op de Aarde.

Nu zullen de details van de klokbias $\Delta\delta_i^j(t)$ nader worden toegelicht.

Gegeven een bepaald tijdstip, een enkele positie i is automatisch geïmpliceerd. Elke satelliet levert een onbekende klokbias, welke is aangeduid als j bij de klokcomponent. Als we de klokbias van het observatiepunt i verwaarlozen, hebben we vier onbekenden in de vergelijking ter berekening van de pseudo afstand van de eerste satelliet. Er zijn drie plaats coördinaten en één klokbias van deze satelliet. Elke toegevoegde satelliet levert één vergelijking met dezelfde plaats coördinaten maar met een 'nieuwe' satelliet klokbias.

Gelukkig is de satelliet klokinformatie bekend en wordt broadcast verzonden in de vorm van drie polynominale coëfficiënten a_0, a_1 en a_2 met een referentie tijd t_c. Vandaar dat de vergelijking,

$$\delta^j(t) = a_0 + a_1(t - t_c) + a_2(t - t_c)^2, \qquad \text{[III.6]}$$

ons in staat stelt de berekening van de satelliet klokbias op tijdstip t te doen. Ondanks dat [III.6] een groot deel van de satelliet klokbias wegneemt, blijft er toch een kleine foutmarge over.

De gecombineerde bias component $\Delta\delta_i^j(t)$ kan in tweeën worden gesplitst,

$$\Delta\delta_i^j(t) = \delta_i(t) - \delta^j(t), \qquad \text{[III.7]}$$

waar de satelliet gerelateerde component bekend is door [III.6] en de ontvanger gerelateerde component $\delta_i(t)$ onbekend blijft. Substitueer [III.7] in [III.4] en verplaats de satelliet gerelateerde component naar links, dan krijgen we de vergelijking om de pseudo afstand te berekenen,

$$R_i^j(t) + c\delta^j(t) = \rho_i^j(t) + c\delta_i(t). \qquad \text{[III.8]}$$

Het rechtergedeelte bevat nog steeds vier onbekenden. Om de vier onbekenden te achterhalen, moet het aantal observaties gelijk of meer dan het aantal onbekenden zijn. Het bewijs volgt hierna.

Zij het aantal observaties is $n_j n_t$ waar n_j het aantal satellieten is, en n_t het aantal tijdstippen. Voor de positionering zijn de drie coördinaten en de klok bias van de ontvanger onbekend. Dus het aantal onbekenden is $3 + n_t$, dit geeft ons de vergelijking

$$n_j n_t \geq 3 + n_t. \qquad \text{[III.9]}$$

Welke de volgende relatie impliceert

$$n_t \geq \frac{3}{n_j - 1}. \qquad \text{[III.10]}$$

Het minimale aantal satellieten om aan de ongelijkheid te voldoen is $n_j = 2$, waarbij het aantal observatie tijdstippen $n_t \geq 3$ is. Voor $n_j = 4$, wordt de oplossing $n_t \geq 1$ verkregen. Dit reflecteert de feature van GPS dat het bliksemsnel de positie van een ontvanger kan bepalen, waar de vier onbekenden op elk tijdstip worden opgelost als er minstens vier satellieten getraceerd kunnen worden. Verdere wiskundige details over GPS zijn te vinden in [L6]. En voor factoren die van invloed zijn op de performance van GPS wordt verwezen naar [W3].

Frequenties

Het GPS signaal gaat over verschillende frequenties, ze zullen kort worden omschreven.

- L1 (1575,42 MHz):
  Voor het publiek, de 'coarse/acquisition' (C/A) code, en een encrypted code P(Y).
- L2 (1227,60 MHz):
  Bevat de P(Y) code. De encryptie sleutels om gebruik te kunnen maken van P(Y) zijn in beheer bij de regering van de VS en worden enkel toegekend voor militaire doeleinden.
- L3 (1381,05 MHz):
  Dit signaal wordt (wederom) gebruikt voor militaire doeleinden, in dit geval voor het detecteren van raketten, nucleaire explosies en andere '*high-energy*' infrarood signalen.
- L4 (1841,40 MHz):
  Wordt bestudeerd voor additionele ionosferische correctie, voor de andere signalen.
- L5 (1176,45 MHz):
  Dit signaal wordt gebruikt voor het detecteren van belangrijke (civiele) personen, de zogenaamde '*safety-of-life*' signaal.

Kloksignaal
In bovengenoemde frequenties wordt het kloksignaal verzonden, dit bestaat uit de GPS tijd, wat het aantal seconden vanaf middernacht op 5 januari 1980 inhoudt. De klokken op de satellieten zijn ook onder invloed van de speciale en de algemene relativiteit, die de klokken 38 microseconden per dag trager maakt. Dit wordt door de elektronica op de satellieten aangepast. Dit is trouwens één van de weinige bewijzen van de relativiteitstheorie van Einstein in de praktijk!

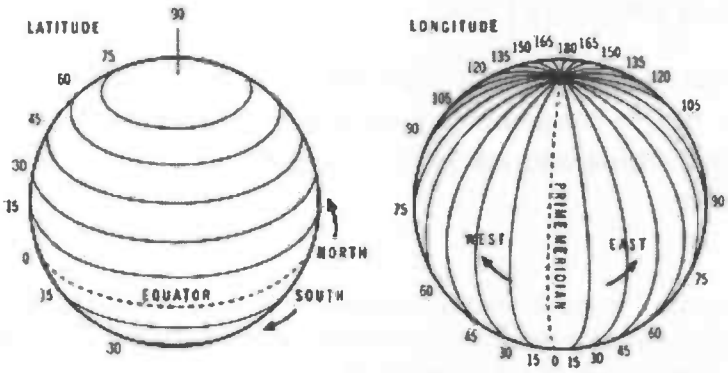
NMEA
In de GPS industrie is er een standaard voor het verwerken van GPS data. De hardware in de ontvangers ontvangt, decodeert en verwerkt het tot 'begrijpelijke' data. De National Marine Electronics Association (NMEA) heeft hiertoe de standaard ontwikkeld, voor de applicaties die gebruik maken van GPS ontvangers.

De NMEA data is in de vorm van kommagescheiden zinnen, een voorbeeld:

```
$GPRMC,040302.663,A,393.7,N,10506.6,W,0.27,358.86,200804,,*1A
```
[III.11]

Deze ene zin bevat bijna alles wat een GPS applicatie nodig heeft: latitude, longitude, snelheid, ligging, satelliettijd, versie-informatie en richting (kompas). In hoofdstuk 4 van dit verslag wordt ingegaan op de werking van de sentence parser.

Latitude en Longitude
Waar het bij *LBS* in combinatie met GPS allemaal om gaat, zijn de coördinaten. De coördinaten die door de GPS ontvangers worden geleverd zijn in latitudes en longitudes.
Een latitude is de breedtegraad van de Aarde, d.w.z. de hoek tussen de Noordpool/Zuidpool en de equator. En een longitude is de lengtegraad, d.w.z. de hoek tussen de nulmeridiaan en de punt in het oosten of in het westen. Zie figuur 2.2.



Figuur 2.2: Latitude en Longitude

De GPS ontvanger berekent de longitude en de latitude aan de hand van de vectoren tussen de satellieten en het middelpunt van de Aarde. Op die manier kunnen de hoeken (latitude en longitude) berekend worden via goniometrie [W7].

Appendix IV: WiFi lokalisatie algoritme

Het (statistische) algoritme ter positiebepaling

Laat de set $L = \{l_1, l_2, ..., l_n\}$ de pregeselecteerde posities (markeringsposities) in een bepaald gebied zijn en $A = \{a_1, a_2, ..., a_m\}$ de accesspoints van de WiFi netwerk (3.3). Voor elke l_i de signaalsterkte $s\,(0 \le s \le 100)$ gedetecteerd vanuit accespoint a_j geldt de waarschijnlijkheids distributie $p_i^j(s)$, waar geldt

$$\sum_{s=0}^{100} p_i^j(s) = 1. \qquad \text{[IV.1]}$$

Voor een onbekende locatie x wordt aangenomen dat het geobserveerde signaal $O = \{o_1, o_2, ..., o_m\}$ waar o_j de signaalsterkte van accespoint a_j is. Dan wordt de waarschijnlijkheid $P(l_i \mid O)$ voor elke positie l_i. De kunst is dus het vinden van een positie, l_i, waar de waarschijnlijkheid $P(l_i|O)$ maximaal is. In wiskundige termen is dat

$$P(l_i \mid O) = \frac{P(O \mid l_i)P(l_i)}{P(O)}, \qquad \text{[IV.2]}$$

waar $P(O \mid l_i)$ de conditionele waarschijnlijkheid van observatie O op positie l_i is; $P(O)$ de normaliseringconstante is; en $P(l_i)$ is de (voorkeur) waarschijnlijkheid, waar de positie l_i de correcte is. Dat laatste kan als constant worden verondersteld, of worden afgeleid uit metingen/berekeningen.
De waarschijnlijkheid $P(O \mid l_i)$, de O op positie l_i, kan weergegeven worden als

$$P(O \mid l_i) = \prod_{j=1}^{m} P(o_j \mid l_i = \prod_{j=1}^{m} p_i^j(o_j). \qquad \text{[IV.3]}$$

Uit de vergelijkingen [IV.2] en [IV.3] blijkt dat de waarschijnlijkheidsdistributie $p_i^j(s)$ en de waarschijnlijkheid $P(l_i)$ de dominante factoren zijn bij de positieberekening. Deze twee factoren moeten van tevoren bekend zijn, om ze te verkrijgen wordt in twee fasen gewerkt: de *training* fase en de *working* fase.

Training fase

Tijdens de training fase worden samples verzameld om L te vullen. In het algemeen zou de markeringsposities zorgvuldig geselecteerd moeten worden, zo dat ze gelijkmatig staan verdeeld in het gebied. Uit onderzoek [A10] is gebleken dat de meetrichting (NZWO) de signaalsterkte zelf ook beïnvloedt. Dus voor elke sample worden de vier metingen, elk een andere richting, in L opgenomen voor één fysieke positie. Deze verzamelde signalen worden gebruikt om $p_i^j(s)$ te bepalen. Deze factor is dominant, het is dus van belang de signalen veelvuldig en zorgvuldig te 'trainen'. Hiertoe is een methode in [A10] beschreven.

Working fase

Tijdens de working fase detecteert de mobiele device een signaal van elke access point, en gebruikt het model (2.3.1) om de positie real-time te berekenen. Hoewel, vanwege de aanwezigheid van de variatie van signaalsterktes in omgevingen, die binnenshuis zijn, is het onmogelijk om de foutmarge volledig te elimineren. Deze kan echter wel verkleind worden door

de kennis over de inrichting van gebouwen te benutten, daartoe is ook een algoritme ontworpen, zie daarvoor [A10].

Appendix V: Use-cases

In deze bijlage worden de use-cases van *Lisa* behandeld, er wordt onderscheid gemaakt tussen de cliënt use-cases en de serviceprovider use-cases.

Nota bene: deze cases zijn in de ontwerpfase opgesteld.

Cliënt use-cases

- De gebruiker meldt zich aan en logt in
- De applicatie kan zijn eigen locatie bepalen
- De gebruiker kiest een route
- De gebruiker loopt een route
    - De gebruiker ziet route op grafisch kaart
    - De gebruiker leest gegevens over een locatie
    - De gebruiker vraagt informatie van zijn gewenste route op
    - De gebruiker scant een RFid TAG
        - De gebruiker voert RFid TAG handmatig in
    - De gebruiker ontvangt een bericht
    - De gebruiker maakt een nieuw bericht

- De gebruiker maakt een eigen route
    - De gebruiker kiest een locatie
    - De gebruiker ontvangt mogelijke locaties in de buurt van de dienst
    - De gebruiker slaat nieuwe locatie op in de database
    - De gebruiker staat zijn nieuwe route op in de database

- De gebruiker raadpleegt de helpfunctie betreffende het huidige scherm
- De gebruiker logt uit de dienst

Serviceprovider use-cases

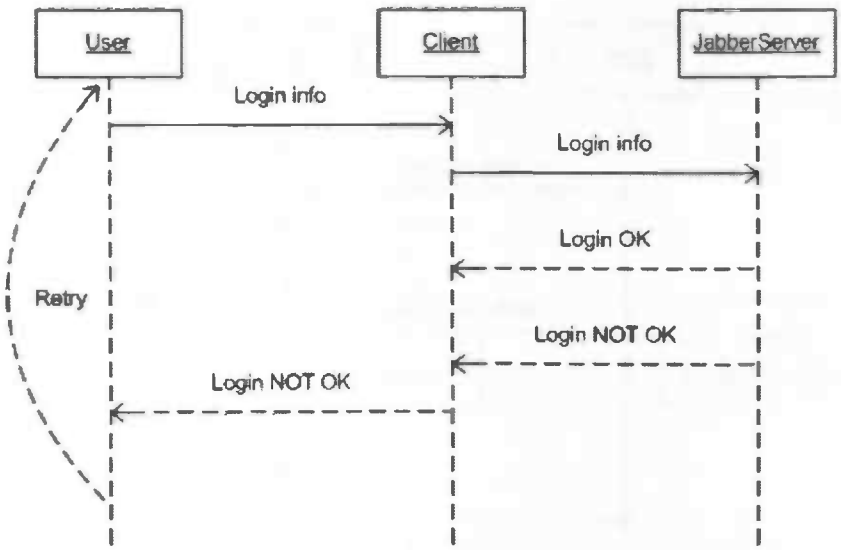
- De server meldt zich aan bij de Jabber server en logt in
- De server ontvangt berichten van de gebruikers
- De server handelt de berichten van de gebruikers af
    - De server levert gegevens over locatie uit DB
    - De server levert routes uit DB
    - De server levert berichten uit DB
    - De server levert kaarten uit DB
    - De server zet nieuwe gegevens over locatie in DB
    - De server zet nieuwe routes in DB
    - De server zet nieuwe berichten in DB
    - De server zet nieuwe kaarten in DB
- De server roept webservices aan van MiloWare
    - De server zoekt alle locaties in de buurt van gegeven coördinaten op
    - De server haalt de locatie ID van objecten uit de database van MiloWare

Op de volgende pagina worden de cliënt use-cases nader uitgewerkt.

Cliënt use-cases

Prioriteit = Must have, should have, could have

Use-case:	**1.1**
Naam	: Inloggen op de server
Omschrijving	: De gebruiker logt in op de jabber server
Prioriteit	: Must have
Vereiste	: ✓ Verbinding met de dienst aanwezig ✓ De gebruiker moet nog niet ingelogd zijn ✓ Gebruikersnaam en code moeten bekend zijn bij gebruiker en server
Beschrijving	: **Gebruiker** / **Systeem** Klik op applicatie Startapplicatie op Geef inlogscherm Voer naam en wachtwoord in Klik op aanmelden Controleer invoer Geeft hoofdpagina weer
Resultaat	: Gebruiker is ingelogd
Uitzonderingen	: ✓ Gebruikersgegevens zijn onbekend of onjuist ✓ Gebruik is al ingelogd ✓ Verbinding valt weg tijdens inloggen



De user voert login informatie, en applicatie logt in bij server

Het volgende omschrijft de dingen die fout kunnen gaan;

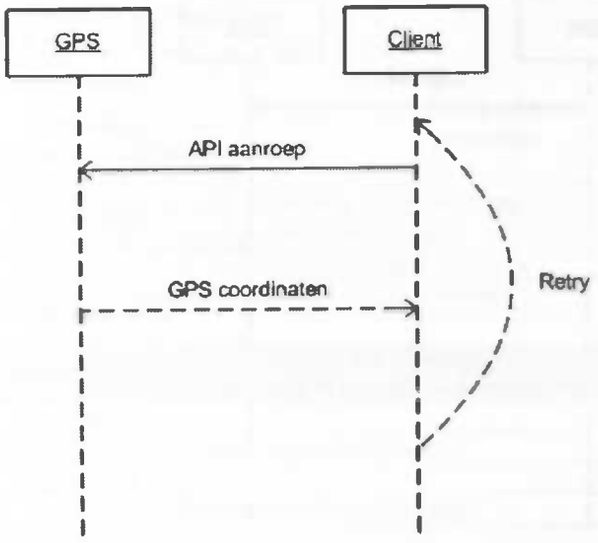
1. Als de gebruiker een onbekende username invoert, dan krijgt de cliënt een user_error terug, die aangeeft dat de username onbekend is.

2. Als de gebruiker een foutief wachtwoord (bij een juiste username) invoert, dan krijgt de cliënt een pwd_error terug, die aangeeft dat de password fout is.

Op die manieren kan de cliënt de juiste waarschuwing (in de vorm van een pop-up bericht) krijgen.

Heeft de user de juiste username en password combinatie ingevoerd, dan wordt de *Lisa* dienst beschikbaar gezet voor de desbetreffende user.

Use-case: 1.2		
Naam	:	Eigen locatie bepalen
Omschrijving	:	De applicatie is in staat haar eigen locatie bepalen
Prioriteit	:	Must have
Vereiste	:	✓ GPS ontvanger moet aangesloten zijn op de PDA ✓ GPS ontvanger moet in verbinding staan met satellieten, '*open sky*'.
Beschrijving	:	**Gebruiker** / **Systeem** De applicatie vraagt om de 50 meter de huidige positie van de gebruiker op Bufferen van locatiegegevens Coördinaten worden 2x per minuut verstuurd mits deze zijn veranderd.
Resultaat	:	Gebruiker kan nu gebruik maken van de dienst omdat het systeem weet waar hij/zij zich bevindt.
Uitzonderingen	:	✓ GPS heeft geen ontvangst

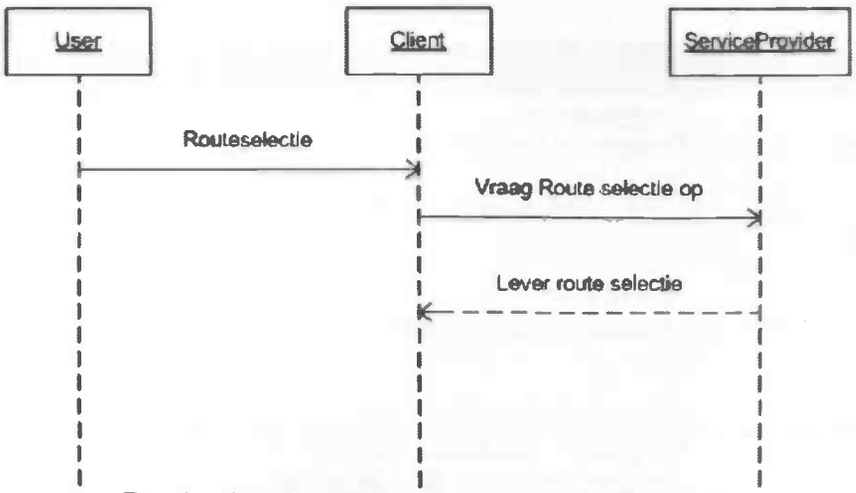


De applicatie vraagt om GPS coördinaten

De cliënt vraagt middels de GPS API de coördinaten van de GPS receiver. De GPS coördinaten worden opgeslagen als een `Positie` object, welke leesbaar is voor MapPoint. De cliënt vraagt continu om de 50 meter de coördinaten van de GPS ontvanger.

Dit is een must have, aangezien het een *LB* applicatie is, de eigen locatie is daarbij van essentieel belang.

Use-case: 1.3		
Naam	:	Route kiezen
Omschrijving	:	De gebruiker kiest een route
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met de dienst aanwezig ✓ Coördinaten zijn bekend ✓ Beginpunt van de route maximaal 1,5 km van de huidige positie van de gebruiker
Beschrijving	:	**Gebruiker** / **Systeem** Klikt op "kies een route". Geeft "Kies een route" scherm weer. Geeft een lijst met route categorieën weer. Selecteert een categorie. Geeft een lijst met mogelijke routes weer. Selecteert een route. Klikt op "Toon beschrijving". Open nieuw scherm met beschrijving van de route. Klikt op sluit scherm. Scherm wordt gesloten Klikt op start. Geeft "Routebegeleiding" scherm weer, tabblad "Kaart".
Resultaat	:	Gebruiker heeft een route geselecteerd en kan beginnen met het lopen van de route onder begeleiding van de applicatie
Uitzonderingen	:	✓ Geen routes beschikbaar binnen 1,5 km vanaf de huidige locatie ✓ GPS coördinaten zijn niet bekend ✓ Geen verbinding met de dienst
Bijzonderheden		Routes worden gerangschikt op volgende bij: Populair → Aantal keer gelopen Officieel → Datum Gebruiker → Datum



De gebruiker selecteert de routecategorie en de SP levert dat

De gebruiker selecteert de route categorie, de Client stuurt dat door naar de ServiceProvider. De SP haalt de routes uit de Database en stuurt dat door naar de Client, zodat de gebruiker kan zien welke routes beschikbaar zijn binnen een bepaalde straal.

Na de selectie komt de gebruiker in het tabblad 'Kaart', waarin de gebruiker op een plattegrond kan zien wat de route is en waar hij/zij zich bevindt. Hierbij vindt geen communicatie plaats, aangezien de kaart al is opgehaald vanuit de server (zit in de row). Met de data wordt een nieuwe bitmap aangemaakt en weergegeven.

Dit is een must have aangezien het bewandelen van routes het speerpunt is van de gehele applicatie.

Use-case: 1.4a (Kaart)		
Naam	:	Route lopen – Kaart
Omschrijving	:	De gebruiker loopt een route
Prioriteit	:	Must have
Vereiste	:	✓ Coördinaten moeten bekend zijn bij gebruiker ✓ Verbinding met de dienst aanwezig
Beschrijving	:	**Gebruiker** / **Systeem** Klikt op tabblad "Kaart" / Geeft een grafische weergave van de route weer. Klikt op besturingselement / Geeft aanpassingen grafisch weer. Loopt de afgelegde route Klikt op "Tekst" voor tekstuele begeleiding / Geeft tekstuele begeleiding van de route weer.
Resultaat	:	De gebruiker kan zich oriënteren d.m.v. een kaart.
Uitzonderingen	:	✓ Gebruiker loopt verkeerde route
		✓ Geen communicatie met MapPoint server
		✓ Coördinaten gebruiker zijn niet bekend

Use-case: 1.4b (Locatie)		
Naam	:	Locatiegegevens
Omschrijving	:	De gebruiker leest gegevens over de locaties
Prioriteit	:	Should have
Vereiste	:	✓ Verbinding met de dienst aanwezig
Beschrijving	:	**Gebruiker** / **Systeem** Klikt op tabblad "Locaties" / Tabblad "Locaties" weergeven.
Resultaat	:	De gebruiker kan de locaties zien waar de route langs komt met daarbij de adresgegevens. Ook de RFid die bij een locatie horen worden weergegeven.
Uitzonderingen	:	✓ Database kan niet bereikt worden

Use-case: 1.4c (Informatie)		
Naam	:	Informatiegegevens
Omschrijving	:	De gebruiker vraagt informatie van zijn gewenste route op
Prioriteit	:	Should have
Vereiste	:	✓ Verbinding met de dienst aanwezig
Beschrijving	:	

Gebruiker	**Systeem**
Klikt op tabblad "Informatie"	
	Tabblad "Informatie" weergeven. Selecteert standaard dichtstbijzijnde locatie Toont beschrijving van geselecteerde locatie
Selecteert andere locatie	
	Toont beschrijving van geselecteerde locatie
Scrollt naar beneden	
	Toont overige informatie van de beschrijving

Resultaat	:	De gebruiker kan de locaties zien waar de route langs komt met daarbij de adresgegevens. Ook de RFid die bij een locatie horen worden weergegeven.
Uitzonderingen	:	✓ Database kan niet bereikt worden

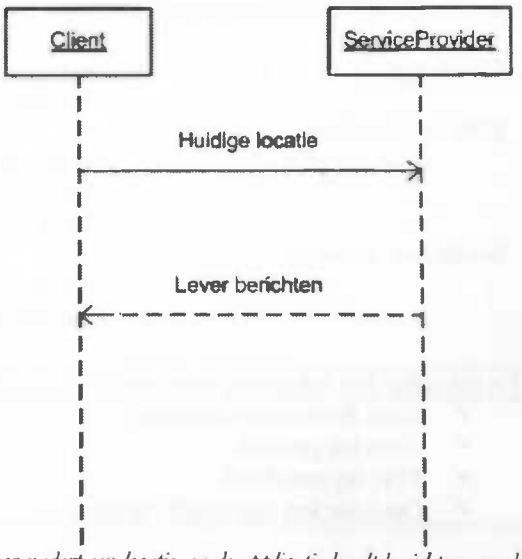
Use-case: 1.4d (Scannen RFid)		
Naam	:	Scannen RFid
Omschrijving	:	De gebruiker scant een RFid TAG
Prioriteit	:	Could have
Vereiste	:	✓ RFid tag aanwezig ✓ RFid reader aangesloten op de PDA ✓ Verbinding met de dienst aanwezig ✓ Informatie in database aanwezig
Beschrijving	:	

Gebruiker	**Systeem**
	Tabblad "Informatie" weergeven.
Klikt op "Scannen"	
	Activeert RFid reader. Scant object Toon beschrijving object.
Scrollt naar beneden	
	Toont overige informatie van de beschrijving

Resultaat	:	De gebruiker kan informatie lezen van RFid objecten
Uitzonderingen	:	✓ Geen RFid reader aanwezig. ✓ Geen tag gescand. ✓ RFid tag onbekend. ✓ Database kan niet bereikt worden.

Use-case:	**1.4e (Alternatief RFid scannen)**
Naam	: Invoeren RFid code
Omschrijving	: De gebruiker voert RFid TAG handmatig in
Prioriteit	: Should have
Vereiste	: ✓ RFid tag code zichtbaar ✓ Verbinding met de dienst aanwezig ✓ Informatie in database aanwezig
Beschrijving	: **Gebruiker** / **Systeem** Toont handmatige invoerscherm voor RFid. Voert RFid code in. Toont overige informatie van beschrijving.
Resultaat	: De gebruiker kan handmatig informatie opvragen en lezen van een RFid object
Uitzonderingen	: ✓ RFid tag onbekend. ✓ Database kan niet bereikt worden.

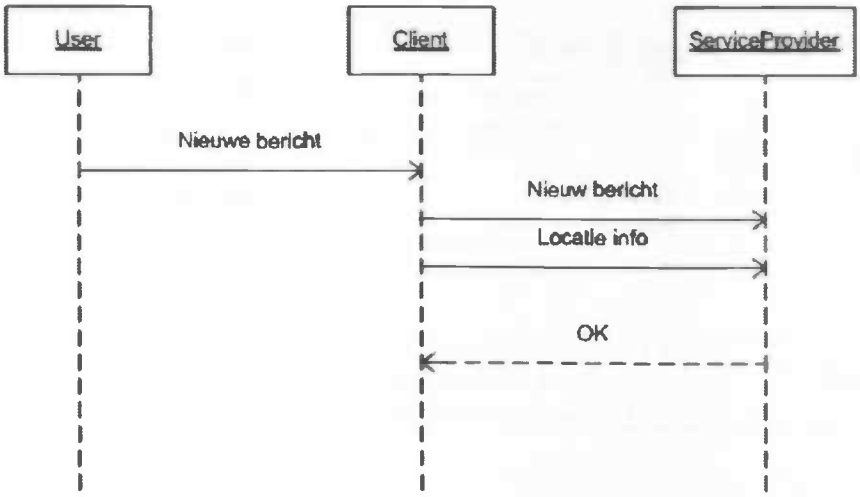
Use-case:	**1.5**
Naam	: Bericht ontvangen
Omschrijving	: De gebruiker ontvangt een bericht
Prioriteit	: Should have
Vereiste	: ✓ Verbinding met de dienst aanwezig
Beschrijving	: **Gebruiker** / **Systeem** Geeft indicatie van een nieuwe bericht Klikt op indicatie Geeft tabblad berichten weer Selecteert gewenste bericht om te lezen Toont geselecteerde bericht
Resultaat	: De gebruiker kan nieuwe berichten lezen
Uitzonderingen	: ✓ Geen verbinding met Dienst



De gebruiker nadert een locatie, en de applicatie haalt berichten van die locatie op.

Als de gebruiker een bepaalde locatie nadert, en die komt overeen met een locatie in de route, dan haalt de cliënt de plaatsgebonden berichten op. De gebruiker kan aan het tabblad zien of er nieuwe berichten zijn.

Use-case:	**1.5a**	
Naam	:	Bericht versturen
Omschrijving	:	De gebruiker maakt een nieuw bericht
Prioriteit	:	Should have
Vereiste	:	✓ Verbinding met de dienst aanwezig
Beschrijving	:	
		Gebruiker / **Systeem**
		Klikt op button
		Geeft invoer mogelijkheid voor nieuw bericht
		Voert tekst voor bericht in. Drukt op bericht versturen
		Slaat bericht op.
Resultaat	:	De gebruiker kan berichten versturen
Uitzonderingen	:	✓ Geen verbinding met Dienst



De gebruiker voert nieuwe bericht in, en de applicatie stuurt dat door naar SP

Als de gebruiker bij een locatie een bericht wil achterlaten, dan moet de gebruiker de bericht invoeren, en op 'Verzenden' klikken. De locatie informatie en de inhoud van dat bericht wordt dan verzonden naar de SP, en wordt tenslotte opgeslagen in de database.

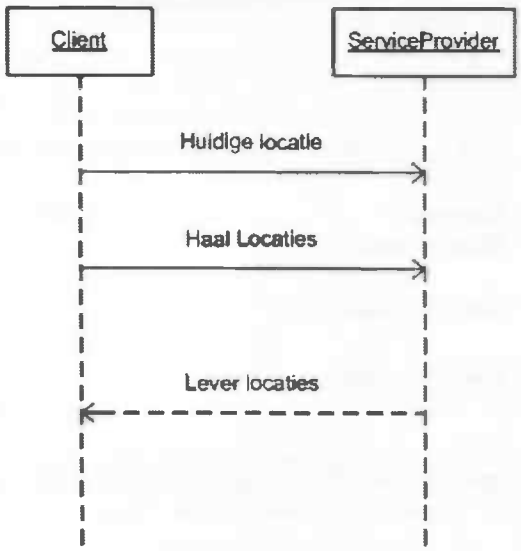
Use-case: 1.6		
Naam	:	Eigen route maken
Omschrijving	:	De gebruiker maakt een eigen route
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met de dienst aanwezig ✓ Coördinaten van de gebruiker moeten bekend zijn.
Beschrijving	:	**Gebruiker** / **Systeem** Geeft hoofdscherm weer Klikt op button Geeft feedback Geeft nieuw scherm weer Begint met lopen van eigen route
Resultaat	:	Gebruiker kan beginnen met het maken van een eigen route
Uitzonderingen	:	✓ Geen verbinding met de dienst ✓ Geen ontvangst met GPS ✓ Ontvangt geen routekaart

Use-case: 1.6a		
Naam	:	Locatie kiezen
Omschrijving	:	De gebruiker kiest een locatie
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met de dienst aanwezig ✓ Coördinaten van de gebruiker moeten bekend zijn.
Beschrijving	:	**Gebruiker** / **Systeem** Klikt op locatie Toont wizard voor aanmaken van een locatie
Resultaat	:	Er wordt een wizard gestart om locatie toe te voegen
Uitzonderingen	:	✓ Geen verbinding met de dienst ✓ Geen ontvangst met GPS

Use-case: 1.6b		
Naam	:	Locatie in de buurt
Omschrijving	:	De gebruiker ontvangt mogelijke locaties in de buurt van de dienst
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met de dienst aanwezig ✓ Coördinaten van de gebruiker moeten bekend zijn.
Beschrijving	:	

Gebruiker	**Systeem**
	Toont de wizard Toont mogelijke locaties in de buurt van de gebruiker
Selecteert een locatie	
	Slaat selectie op Toont 2e scherm van wizard Vraagt om bevestiging
Klikt op invoegen	
	Voegt locatie toe aan route Sluit wizard

Resultaat	:	Bestaande locatie toevoegen aan eigenlijk route
Uitzonderingen	:	✓ Geen verbinding met de dienst ✓ Geen ontvangst met GPS

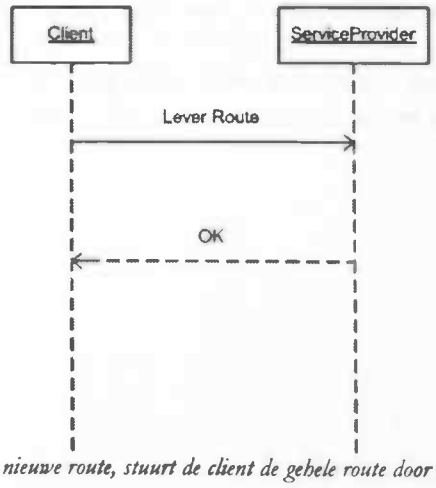


De applicatie haalt alle locaties binnen een bepaalde straal

Als de gebruiker een bestaande locatie wil toevoegen aan zijn route, dan haalt de applicatie alle bestaande locaties binnen een bepaalde straal van de SP vandaan.

Use-case: 1.6c		
Naam	:	Opslaan nieuwe locatie
Omschrijving	:	De gebruiker slaat nieuwe locatie op in de database
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met de dienst aanwezig ✓ Coördinaten van de gebruiker moeten bekend zijn.
Beschrijving	:	**Gebruiker** / **Systeem** Toont de wizard Klikt op button om nieuwe locatie aan te maken Toont 2e scherm van wizard Klikt op invoerveld Toont invoermethode Voert gegevens in Klikt op button
Resultaat	:	Nieuwe locatie aanmaken en toevoegen aan eigen route
Uitzonderingen	:	✓ Geen verbinding met de dienst ✓ Geen ontvangst met GPS

Use-case: 1.6d		
Naam	:	Opslaan eigen route
Omschrijving	:	De gebruiker slaat zijn nieuwe route op in de database
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met de dienst aanwezig ✓ Coördinaten van de gebruiker moeten bekend zijn.
Beschrijving	:	**Gebruiker** / **Systeem** Klikt op opties Toont opties Klikt op opslaan route Toont vraag om bevestiging Geeft bevestiging Slaat route op
Resultaat	:	Eigen route wordt opgeslagen en is beschikbaar voor andere gebruikers
Uitzonderingen	:	✓ Geen verbinding met de dienst ✓ Geen ontvangst met GPS



Bij creatie nieuwe route, stuurt de client de gehele route door naar de SP

Als de gebruiker een nieuwe route heeft gemaakt (aan de hand van locaties), dan stuurt de applicatie de gehele route door naar de SP. De SP slaat de route tenslotte op in de Database.

Use-case: 1.7		
Naam	:	Help raadplegen
Omschrijving	:	De gebruiker raadpleegt de helpfunctie betreffende het huidige scherm
Prioriteit	:	Should have
Vereiste	:	✓ Beschikbaarheid van helpfunctie bij huidige scherm
Beschrijving	:	**Gebruiker** / **Systeem** Klikt op help Toont help scherm van huidige scherm Scroll door tekst (eventueel) Klikt op knop om de sluiten
Resultaat	:	Vraag van gebruiker wordt beantwoord en hij kan verder met de dienst.
Uitzonderingen	:	✓ Beschikbaarheid van helpfunctie

Use-case: 1.8		
Naam	:	Uitloggen
Omschrijving	:	De gebruiker logt uit de dienst
Prioriteit	:	Must have
Vereiste	:	✓ Gebruiker moet ingelogd zijn
Beschrijving	:	**Gebruiker** / **Systeem** Toont hoofdmenu Klikt op opties Toont opties Klikt op afmelden Meld gebruiker af Toont aanmeldscherm
Resultaat	:	Gebruiker is afgemeld uit de dienst. PDA is vrij
Uitzonderingen	:	

Als de applicatie afgesloten wordt, dan wordt de gebruiker (en client) ook automatisch uitgelogd bij de Jabber server.

Serviceprovider use-cases

Use-case:	1.9
Naam	: Inloggen op de server
Omschrijving	: De SP logt in op de jabber server
Prioriteit	: Must have
Vereiste	: ✓ Verbinding met server aanwezig ✓ SP moet nog niet ingelogd zijn ✓ Gebruikersnaam en code moeten bekend zijn bij server
Beschrijving	: **SP** / **Jabber Server** Start op Stuurt login info Authoriseert login Stuurt authorisatie terug Ontvangt authorisatie
Resultaat	: SP is ingelogd, en dienst is beschikbaar
Uitzonderingen	: ✓ Geen internet

Use-case:	1.10
Naam	: Ontvangen van berichten (listener)
Omschrijving	: De SP ontvangt berichten van gebruikers
Prioriteit	: Must have
Vereiste	: ✓ Verbinding met server aanwezig ✓ Bron (gebruiker) moet bekend zijn
Beschrijving	: **SP** / **Cliënt** Luistert Stuurt bericht Ontvangt bericht Authoriseert bericht Interpreteert bericht
Resultaat	: SP weet wat de gebruiker wil op dat moment
Uitzonderingen	: ✓ Onjuiste/onbekende parameters

Use-case: 1.11		
Naam	:	Afhandelen van berichten
Omschrijving	:	De SP voert functies a.d.h.v. parameters uit
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met server aanwezig ✓ Parameters bekend
Beschrijving	:	**SP** — **Cliënt** Roept de juiste functie aan
Resultaat	:	De juiste functie wordt aangeroepen
Uitzonderingen	:	✓ Functie niet beschikbaar

Use-case: 1.11a		
Naam	:	Informatie leveren
Omschrijving	:	De server levert gegevens uit DB
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met server aanwezig ✓ DB in de lucht ✓ Parameters bekend
Beschrijving	:	**SP** — **Cliënt** Ontvangt resultaat van functie van 1.11 Stuurt informatie Ontvangt informatie
Resultaat	:	De client wordt voorzien van gevraagde informatie
Uitzonderingen	:	✓ Client uitgelogd

Use-case: 1.11b		
Naam	:	Informatie opslaan
Omschrijving	:	De server slaat gegevens in DB op
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met server aanwezig ✓ DB in de lucht ✓ Parameters bekend
Beschrijving	:	**SP** — **Cliënt** Ontvangt resultaat van functie van 1.11 Zet informatie in DB
Resultaat	:	De DB wordt voorzien van nieuwe informatie
Uitzonderingen	:	

Use-case: 1.12		
Naam	:	Locatiegegevens MiloWare
Omschrijving	:	De SP haalt locatie gegevens op uit MiloWare platform
Prioriteit	:	Must have
Vereiste	:	✓ Verbinding met server aanwezig ✓ MiloWare in de lucht ✓ Parameters bekend
Beschrijving	:	**SP** — **Cliënt**
Resultaat	:	
Uitzonderingen	:	

Appendix VI: MapPoint code

```
public PictureBox MakeRouteMapByMidLocation(PictureBox pbox, double zoom,
double latitude, double longitude, LatLongArray latLong, int length,
TrailOfLatLong toll)
{
      //store latitude and longitude
      centerpoint = new LatLong();
      centerpoint.Latitude = latitude;
      centerpoint.Longitude = longitude;

      //Define the default view
      defaultView = new ViewByHeightWidth();
      defaultView.CenterPoint = new LatLong();
      defaultView.CenterPoint = latLong.LatLongs[((length-
1)/2)].Latilongi;
      defaultView.Width = 50;
      defaultView.Height = 50;

      //Create an instance of the RenderService
      renderService.UserInfoRenderHeaderValue = new
      UserInfoRenderHeader();

      renderService.UserInfoRenderHeaderValue.DefaultDistanceUnit =
            DistanceUnit.Kilometer;
      renderService.Credentials = new
            System.Net.NetworkCredential(myUserName, myPassword);

      //set pushpins for route
      Pushpin[] pushpins = new Pushpin[length + toll.Trail.Length];

      for(int i = 0; i < (length); i++)
      {
            pushpins[i] = new Pushpin();
            pushpins[i].IconDataSource = "MapPoint.Icons";
            pushpins[i].IconName = "4";
            pushpins[i].LatLong = latLong.LatLongs[i].Latilongi;
            pushpins[i].ReturnsHotArea = false;
            pushpins[i].Label = latLong.LatLongs[i].Naam;
      }

      for(int j = length; j < pushpins.Length; j++)
      {
            pushpins[j] = new Pushpin();
            pushpins[j].IconDataSource = "MapPoint.Icons";
            pushpins[j].IconName = "8";
            pushpins[j].LatLong = new LatLong();
            pushpins[j].LatLong.Latitude = toll.Trail[j-length].Latitude;
            pushpins[j].LatLong.Longitude = toll.Trail[j-
                  length].Longitude;
            pushpins[j].ReturnsHotArea = false;
      }

      pushpins[0].IconName = "28";
      pushpins[length-1].IconName = "31";
      ViewByHeightWidth[] myview = new ViewByHeightWidth[1];
      myview[0] = defaultView;

      //setting mapspecifications
      MapSpecification mapSpec  = new MapSpecification();
```

```
        mapSpec.DataSourceName = "MapPoint.EU";
        mapSpec.Views = myview;

        mapSpec.Pushpins = pushpins;

        mapSpec.Options = new MapOptions();
        mapSpec.Options.Format = new ImageFormat();
        mapSpec.Options.Format.Width = pbox.Width;
        mapSpec.Options.Format.Height = pbox.Height;
        mapSpec.Options.Style = MapStyle.Phone;
        mapSpec.Options.Zoom = zoom;

        //Get the map
        MapImage[] mapImages = renderService.GetMap(mapSpec);
        System.IO.Stream streamImage = new
        System.IO.MemoryStream(mapImages[0].MimeData.Bits);
        Bitmap bitmapImage = new Bitmap(streamImage);
        pbox.Image = bitmapImage;
        return pbox;
}
```

Appendix VII: GPS code

Nmea parser

```
private void objParser_Generic(GpsToolsNETcf.NmeaSentence sentence)
{
      //voor als een NMEA sentence binnenkomt

      m_cnt++;
      if(m_cnt == 0 )
      {
            latitudey = latitude;
            longitudex = longitude;
      }
      else
      {
            if(Settings.Demo == "0")
            {
                  afstandz = CalculateDistance(longitude, latitude,
                                                longitudex, latitudey);
            }
            if(afstandz >= Convert.ToDouble(Settings.Range) && client)
            {
                  latitudey = latitude;
                  longitudex = longitude;

                  if(this.firstlocatie)
                  {
                        if(trailteller < 50)
                        {
                              trailteller++;
                              trail.Trail[trailteller].Longitude =
                                                      longitude;
                              trail.Trail[trailteller].Latitude =
                                                      latitude;
                        }
                        else
                        {
                              MessageBox.Show("De route is al 2,5 km lang,
                                    u dient de route nu te opslaan.");
                        }
                  }
                  else
                  {
                        p.Latitude = latitude;
                        p.Longitude = longitude;
                        p.Gebruikersnaam = lc.c.Gebruikersnaam;
                        p.Password = lc.c.Password;

                        Winfessor.SoapBox.Core.Message.MessagePacket mp =
                  new Winfessor.SoapBox.Core.Message.MessagePacket();
                        XmlSerializer xs = new XmlSerializer();
                        mp.Subject = "setpositie";
                        mp.Body = xs.Serialize(p);

                        lc.jabberClientHandler("send", mp);
                  }
            }

      }
```

```
        sentence.Dispose();
}
```

Calculate distance

```
public double CalculateDistance(double el1_longitude, double el1_latitude,
double el2_longitude, double el2_latitude)
{
        double distance = 0;

        // omvormen naar radialen
        double lat1 = DegToRad(el1_latitude);
        double lat2 = DegToRad(el2_latitude);
        double lon1 = DegToRad(el1_longitude);
        double lon2 = DegToRad(el2_longitude);

        Double i =
        Math.Acos(Math.Sin(lat1)*Math.Sin(lat2)+Math.Cos(lat1)*
        Math.Cos(lat2)*Math.Cos(lon1-lon2));

        distance = i * 6378.137;
        return distance;
}

public double DegToRad(Double degrees)
{
        return (Math.PI/180)*degrees;
}
```