Afstudeerscriptie

Voor het afstudeerproject

“Augmented maintenance: garbage control”

Uitgevoerd bij Logica Nederland BV in het kader van het Working Tomorrow programma

Auteur	: Sander van der Wal
Studentnummer	: *******
Datum	: 31-05-2010
Versie	: 1.0
Afstudeerbedrijf	: Logica Nederland BV
Divisie	: Professional Skills & Consulting
Competence	: Working Tomorrow
Bedrijfsbegeleiders	: Herbert Leenstra
Bedrijfsmentor	: Jos Duker
Onderwijsinstelling	: Hogeschool Utrecht
Opleiding	: Technische Informatica
1e Examinator	: Jan Zuurbier

Versiebeheer

Versie	Omschrijving
0.1	Hoofdstuk “Omgeving en organisatie” en “Afstudeeropdracht” beschreven. Hoofdstuk 4 tot en met 4.3.1 beschreven.
0.2	Hoofdstuk “Resultaten onderzoek” beschreven.
0.2.1	Opmerkingen Jan Zuurbier verwerkt. Paragraaf 4.3 uitgewerkt. Paragraaf 5.5 gedeeltelijk herschreven.
0.3	Opmerkingen Jan Zuurbier verwerkt. Hoofdstuk “Functioneel ontwerp” toegevoegd. Hoofdstuk “Technisch ontwerp” toegevoegd. Hoofdstuk 5.5 “client / server communicatie” aangepast.
0.5	Hoofdstuk 7 “Resultaten” toegevoegd. Hoofdstuk 8 “Conclusie en aanbevelingen” toegevoegd. Voorwoord toegevoegd. Inleiding toegevoegd. Managementsamenvatting toegevoegd. Volledig concept.
0.6	Opmerkingen Herbert Leenstra verwerkt.
0.7	Opmaak afronden.
0.7.5	Opmerkingen Jan Zuurbier en Jos Duker verwerkt.
1.0	Afronden document.

Tabel 0.1: Versiebeheer

Adresgegevens

Examinandus

Naam	: Sander van der Wal
Straat / Nummer	: **********
Postcode / Plaats	: **** ** **********
Telefoonnummer	: **********
Emailadres	: **********
	: **********
Onderwijsinstelling	: Hogeschool van Utrecht
Afstudeerrichting	: Technische Informatica
Studentnummer	: *******

Examinator

Naam	: Jan Zuurbier
Emailadres	: **********
Telefoonnummer	: **********
	: **********

Afstudeercoördinator

Naam	: Marten Wensink
Emailadres	: **********
Telefoonnnummer	: **********

Bedrijfsgegevens

Naam	: Logica Nederland BV
Afdeling	: Working Tomorrow
Straat / Nummer	: Merweplein 23
Postcode / Plaats	: 3430 BJ Nieuwegein

Bedrijfsbegeleiders

Naam	: Herbert Leenstra
Telefoonnummer	: **********
	: **********
Emailadres	: **********
Functie	: Projectmanager Working Tomorrow programma

Bedrijfsmentor

Naam	: Jos Duker
Telefoonnummer	: **********
Emailadres	: **********
Functie	: Software Engineer

Competence Manager

Naam	: Mart Schoenmakers
Telefoonnummer	: **********
	: **********
Emailadres	: **********
Functie	: Competence Manager Java & Oracle

Voorwoord

Deze scriptie is geschreven ter afsluiting van mijn afstudeerproject van de opleiding Technische Informatica aan de Hogeschool Utrecht. Het afstudeerproject is uitgevoerd van februari 2010 tot en met juni 2010 binnen de afdeling Working Tomorrow van Logica te Nieuwegein.

Ik wil hierbij de heer Herbert Leenstra bedanken voor het begeleiden van mijn project binnen Logica en de inhoudelijke feedback op de verslagen. Tevens wil ik mijn inhoudelijk begeleider binnen Logica de heer Jos Duker bedanken voor het inhoudelijk begeleiden en geven van feedback tijdens mijn afstuderen. Tot slot wil ik de heer Jan Zuurbier bedanken als docentbegeleider van de Hogeschool Utrecht voor het begeleiden en leveren van feedback tijdens het afstudeerproject.

Naast hen wil ik alle betrokken medewerkers van Logica en medestudenten van Working Tomorrow bedanken voor hun hulp, vrijgemaakte tijd, opmerkingen en gezellige tijd om dit project tot een goed einde te brengen.

Nieuwegein, Mei 2010

Sander van der Wal

Managementsamenvatting

Volle openbare prullenbakken, graffiti op zitbankjes of bushokken met gebroken ruiten. Het zijn veel voorkomende situaties, echter het melden van dergelijke situaties vraagt tijd en betrokkenheid van de burger.
In de huidige situatie zijn er een aantal manieren om de problemen te melden. Burgers kunnen bij de gemeente of onderhoudsdienst terecht, maar moeten dan wel minimaal de straatnaam en een beschrijving van het probleem geven. Hier moet de burger wel eerst opzoeken waar ze het kunnen melden en dit weerhoudt ze ervan om het ook daadwerkelijk te melden.

Uit dit probleem volgde de volgende hoofdvraag:

“Op welke wijze kan de eindgebruiker door middel van een Android applicatie die gebruik maakt van augmented reality, verleidt worden om problemen of schade aan eigendommen te melden bij de onderhoudsdienst?”

Uit onderzoek is gebleken dat een snelle, intuïtief werkende Android applicatie met dynamische content het interessant maken om een applicatie opnieuw te gebruiken. Door de applicatie snel te laten werken, hoeven eindgebruikers niet of nauwelijks te wachten totdat de Android applicatie reageert. Door het intuïtief te maken is er geen ‘help’ of handleiding nodig om de eindgebruiker te leren omgaan met de Android applicatie.

Om de eindgebruikers meer te betrekken bij het melden van problemen dient er interactie te zijn met de gebruiker. Dit wordt toegepast door een soort score systeem. Eindgebruikers verdienen rewards als ze een probleem melden die nog niet gemeld was. Ook krijgen ze een berichtje te zien waarin ze worden bedankt voor het melden van een probleem.

Om de positie van het probleem automatisch te bepalen wordt gebruik gemaakt van de GPS mogelijkheden van de Android telefoon. Door dit proces, hoeft de eindgebruiker zich geen zorgen meer te maken over de plaats van het probleem.

De rewards worden over de reële beelden getekend op de plek waar de eindgebruiker een melding heeft gedaan. Ook is er terug te zien hoeveel rewards de eindgebruiker heeft gekregen.

Als een eindgebruiker een probleem wilt melden dient er in de Android applicatie een nieuwe melding aangemaakt te worden. Dit proces bestaat uit een aantal stappen die doorlopen moeten worden. Als eerste krijgt de gebruiker de gelegenheid om een foto te maken van het probleem. Nadat de gemaakte foto is goedgekeurd door de eindgebruiker krijgt deze een scherm te zien waarin een categorie geselecteerd kan worden. Als er een categorie is geselecteerd kan de gebruiker nog eventueel een opmerking toevoegen. Dit kan van alles zijn wat de eindgebruiker nuttig vindt om te melden. Ondertussen is de positie bepaald van het probleem en kan de melding verstuurd worden naar de server. De gebruiker krijgt hiervan een voortgang te zien in een voortgangsbalk. Als de melding is opgeslagen op de server krijgt de eindgebruiker een bedankje en een reward.

Inhoudsopgave

1 Inleiding

Dit verslag bevat informatie met betrekking tot de afstudeeropdracht van Sander van der Wal met de titel *“Augmented maintenance: garbage control”*.

Overal komt men problemen tegen; volle prullenbakken, ingegooide ruiten van bushokjes of graffiti op de bankjes. Er is nog geen makkelijke optie om dit soort problemen te melden. Men weet vaak niet waar het gemeld kan worden. Een telefoonnummer moet opgezocht worden of op de website van de gemeente staat een pagina waar het kan worden gemeld. De vraag is echter of men dan nog weet waar het probleem zich bevindt.

Om de bovengenoemde situatie te verbeteren is er besloten om een mobiele applicatie te ontwikkelen waardoor het proces minder tijd en moeite in beslag neemt. Om dit te verwezenlijken is er een onderzoek uitgevoerd om te bekijken hoe de applicatie ontwikkeld kan worden, zodat men deze optie van het melden van problemen gaat gebruiken. Ter demonstratie hiervan is er een proof of concept ontwikkeld.

2 Omgeving en organisatie

2.1 Logica

De afstudeeropdracht wordt uitgevoerd bij Logica te Nieuwegein. Logica plc. is op 30 december 2002 ontstaan uit het voormalige Logica plc. (60%) en het voormalige CMG plc. (40%) Beide ICT-dienstverleners zijn van oorsprong Engelse bedrijven, maar CMG was in Nederland veel groter dan de Engelse moeder. Sinds 13 januari 2006 is tevens het Franse Unilog onderdeel van het bedrijf, waarmee het een derde thuismarkt heeft gecreëerd. Logica is een internationale ICT-dienstverlener en heeft wereldwijd momenteel 40.000 werknemers in 39 verschillende landen in dienst. Het behoort, ook qua omzet, tot de internationale top-20 in de ICT-dienstverlening. De omzet uit de traditionele ICT-dienstverlening wordt voornamelijk in Europa en Australië (continent) gehaald. Logica levert diensten op tal van terreinen, zoals management- en ICT-consultancy, systeemontwikkeling en –integratie en neemt voor klanten complete bedrijfsprocessen in beheer. Het bedrijf ontwikkelt en implementeert oplossingen voor klanten over de hele wereld. Zij maakt daarbij gebruik van geavanceerde technologieën die direct doorwerken in de bedrijfsresultaten van de klant. Logica is een beursgenoteerde onderneming met noteringen aan de London Stock Exchange en Euronext Amsterdam.

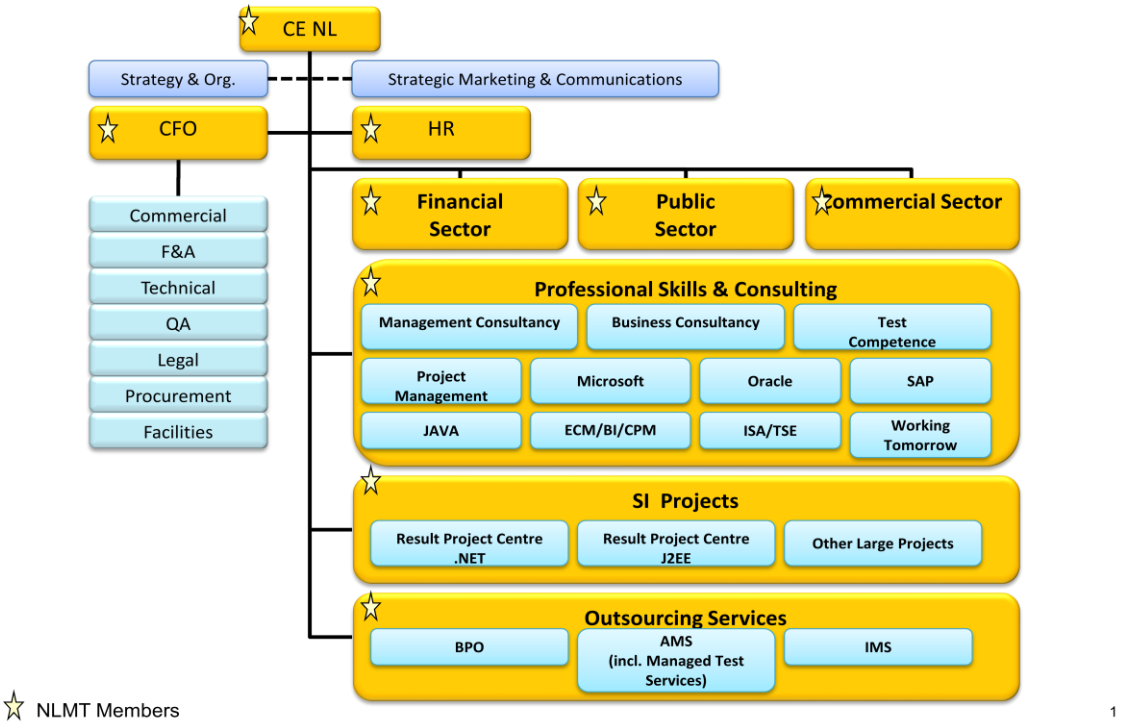
2.2 Working Tomorrow

De opdracht wordt uitgevoerd bij de afdeling Working Tomorrow (WT). Dit is een programma dat opgestart is binnen Logica om plaats te bieden aan studenten die op verschillende locaties binnen het programma aan hun afstudeeropdracht werken. De afstudeeropdrachten hebben een innovatief karakter. Innovatief wat betreft technologie, concept of methodiek. Zo werken er studenten aan agent technologie, G.R.I.D, machine to machine communicatie, aan nieuwe concepten als de thuiscentrale (een centrale die warmwatervoorziening in huis combineert met elektriciteitsopwekking), digitaal papier en intelligente stroom.

Het Working Tomorrow programma heeft 4 hoofddoelen en dit zijn:

1. Een plaats bieden waar studenten uitdagende en innovatieve afstudeerprojecten kunnen uitvoeren.
2. Het werven van toekomstige werknemers.
3. Verhogen van de reputatie van Logica op het gebied van innovatie.
4. Demo’s en prototypen van projecten gebruiken voor het verkrijgen van betaalde opdrachten.

Dit project valt onder de divisie Energy Utilities & Telecom (EUT). Binnen deze divisie wordt nog onderscheid gemaakt tussen verschillende specialismen die competences worden genoemd. Vanuit deze opdracht gezien zal de student binnen het “Working Tomorrow” programma werkzaam zijn in de competence Professional Skills & Consulting.



Figuur 2.1: Organogram van Logica

2.3 Begeleiding

Het project wordt door een aantal personen begeleidt, welke komen vanuit Logica en de Hogeschool van Utrecht. Hieronder zullen alle betrokken begeleiders worden genoemd.

De algemene begeleiding wordt verzorgt door de projectmanager van het Working Tomorrow programma te Nieuwegein, Herbert Leenstra. Hij houdt zich bezig met de organisatorische kant van het afstudeerproject. Tevens houdt hij zich bezig met het reviewen van de gemaakte documentatie.
Voor de inhoudelijke begeleiding is er een mentor binnen Logica aangesteld. Deze rol wordt vervuld doos Jos Duker, software engineer Java. Hij heeft kennis van het Java vakgebied waar het afstudeerproject binnenvalt en kan dus ondersteuning bieden voor de inhoudelijke kant van het project.

Daarnaast wordt er een twee wekelijkse werkoverleg georganiseerd waarin twee afstudeerders ter oefening een presentatie houden over hun afstudeeropdracht. Daarnaast is er de mogelijkheid om algemene zaken of problemen te behandelen.
Uiteindelijk is er ook nog een drie wekelijkse voortgangsbespreking, of vaker indien de afstudeerder dit wenst, met Herbert Leenstra waarin de voortgang van het project wordt besproken.

Vanuit de Hogeschool van Utrecht is er een docentbegeleider aangesteld. Deze begeleider begeleidt de afstudeerder tijdens het afstudeertraject en houdt de voortgang van het project in de gaten. Daarbij biedt de docentbegeleider tijd beschikbaar om documenten te reviewen.

3 Afstudeeropdracht

Voor deze afstudeeropdracht zullen een aantal termen gebruikt worden om ondersteuning te bieden aan het definiëren van de afstudeeropdracht. Hieronder zijn de definities van de termen beschreven.

Augmented reality
Augmented reality is kort gezegd het laten ‘samensmelten’ van rechtstreekse reële beelden met virtuele beelden. Het is een oudere techniek die al wordt toegepast bij tv-uitzendingen. Een voorbeeld hiervan zijn de reclameborden langs het voetbalveld. Echter, door smartphones en de sensoren die ze bevatten, is het sinds 2009 steeds vaker gebruikt.

Android
Android is een open source besturingssysteem voor mobiele apparaten. Dit besturingssysteem is gebaseerd op Linux en het Java platform en is vrij jong[1] (eerste Android telefoon in Nederland kwam uit op 30 januari 2009)[2]. Het feit dat Android open source is, biedt het de mogelijkheid dat ook externe ontwikkelaars applicaties voor Android kunnen maken. Daarnaast is het bekend dat augmented reality is toe te passen op Android telefoons. Een bekend voorbeeld is Layar[3]. Hierdoor is Android zeer geschikt voor dit afstudeerproject.

Android applicatie
Het te ontwikkelen product waar de belanghebbenden mee te maken krijgen is de Android applicatie. Wat deze Android applicatie moet gaan doen volgt in het vervolg van dit hoofdstuk.

Eindgebruiker
De eindgebruiker is iemand die direct gebruik maakt van de mogelijkheden van de te maken Android applicatie. De eindgebruiker moet in het bezit zijn van een mobiel apparaat die op Android draait.

Onderhoudsdienst
De term onderhoudsdienst heeft hier betrekking op een bedrijf of onderdeel van een bedrijf dat onderhoud pleegt aan eigendommen of eigendommen van het bedrijf dat hen inhuurt.

Problemen
Bij problemen moet er gedacht worden aan situaties die het gebruik van objecten belemmeren. Denk hier bijvoorbeeld aan volle prullenbakken, graffiti of schade aan objecten.

3.1 Probleemstelling

Volle openbare prullenbakken, graffiti op zitbankjes of bushokken met gebroken ruiten. Het zijn veel voorkomende situaties, echter het melden van dergelijke situaties vraagt tijd en betrokkenheid van de burger.
In de huidige situatie zijn er een aantal manieren om de problemen te melden. Burgers kunnen bij de gemeente of onderhoudsdienst terecht, maar moeten dan wel minimaal de straatnaam en een beschrijving van het probleem geven. Hier moet de burger wel eerst opzoeken waar ze het kunnen melden en dit weerhoudt ze ervan om het ook daadwerkelijk te melden. Daarnaast wordt stadsmeubilair regelmatig gecontroleerd op problemen[4]. Dit wordt gedaan door de werknemers van het onderhoudsbedrijf, terwijl ze beter ingezet kunnen worden om reparaties en onderhoud uit te voeren.

3.2 Opdrachtomschrijving

Het in paragraaf 3.1 vastgestelde probleem kan worden opgelost door een mobiele applicatie te ontwikkelen welke de burger verleidt om problemen te melden, zonder dat ze zich zorgen hoeven te maken waar ze het moeten melden. De mobiele applicatie kan geïnstalleerd worden op telefoons die op Android draaien. De burgers komen overal en hebben zelf overlast van de problemen en kunnen door de applicatie ter plekke het probleem melden.

Voor de beschrijving van de Android applicatie wordt het werkgebied ervan beperkt tot het stadsmeubilair dat valt onder het beheer van JCDecaux[5]. JCDecaux is een bedrijf dat gemeenten stadsmeubilair biedt en zorgt voor de plaatsing, onderhoud en reparatie. In ruil hiervoor krijgen ze het recht om reclameruimte te exploiteren. Voor dit bedrijf is gekozen, omdat het veel stadsmeubilair in Utrecht exploiteert. Utrecht ligt tegen Nieuwegein aan (locatie van uitvoering van het project), waardoor dichtbij getest kan worden. JCDecaux is niet de opdrachtgever en heeft geen enkele invloed op dit project. Het bedrijf is gekozen om voor het proof of concept een werkgebied af te bakenen.

Dit afstudeerproject levert een Android applicatie op waarmee de eindgebruiker aan kan geven wat het probleem is. De eindgebruiker kan de camera van zijn mobiele telefoon richten op het object van JCDecaux wat onderhoud nodig heeft en met een druk op de knop wordt de nodige informatie verstuurd. Afhankelijk van de locatie van het stadsmeubilair, komt de informatie terecht bij het juiste kantoor van JCDecaux. Het kantoor van JCDecaux ontvangt de foto met locatie data en ziet waar de foto is genomen. Zij kunnen dit doorgeven aan het personeel dat vervolgens een adequate reactie daarop kan nemen.

Door augmented reality toe te passen is de Android applicatie aantrekkelijk te maken voor de eindgebruiker. Een idee is om de omgeving waar de gebruiker zich bevindt van invloed te zijn op virtuele beelden die tevoorschijn komen.
Als de burger het probleem meldt bij de onderhoudsdienst kan het adequater worden verholpen. Door het toevoegen van beeld en een korte beschrijving is er tevens bekend wat het probleem is, hierdoor kan het probleem efficiënter worden verholpen. Hierdoor hoeven de onderhoudsdiensten minder personeel in te zetten om het meubilair te controleren.

3.3 Hoofdvraag

De centrale vraag die naar aanleiding van de probleemstelling naar voren komt luidt:

“Op welke wijze kan de eindgebruiker door middel van een Android applicatie die gebruik maakt van augmented reality, verleidt worden om problemen of schade aan eigendommen te melden bij de onderhoudsdienst?”

3.4 Deelvragen

Om een concreet antwoord op de hoofdvraag te kunnen geven, zijn vier deelvragen geformuleerd:

1. *“Hoe werkt het Android platform wat betreft de applicatie opbouw en sensoren?”*
2. *“Hoe kan augmented reality worden toegepast op de Android applicatie?”*
3. *“Welke elementen maakt de Android applicatie aantrekkelijk voor de eindgebruiker?”*
4. *“Wat zijn de mogelijkheden om de onderhoudsdienst op de hoogte te brengen van de problemen?”*

3.5 Projectgrenzen

Om te voorkomen dat er onduidelijkheid bestaat over de scope van het project en het te maken product binnen de vastgestelde tijd afkomt, zijn er een aantal projectgrenzen opgesteld:

1. Er wordt geen zorg geboden aan de opgeleverde producten na afronding van het project.
2. Er wordt geen rekening gehouden met eisen van de eindgebruiker van de server applicatie.
3. Er zal geen implementatie in een live omgeving van de Android- en server applicatie plaatsvinden.
4. Er zal een simpele server applicatie gemaakt worden ter demo dat er gecommuniceerd wordt.
5. De Android applicatie is alleen te gebruiken in de open lucht en wordt beperkt tot het stadsmeubilair van JCDecaux in de stad Utrecht.
6. De Android applicatie wordt gemaakt voor de Nederlandse markt.

4 Aanpak

Tijdens het afstudeerproject is gebruik gemaakt van een tweetal onderdelen van Dynamic Systems Development Method[6] (DSDM). Dit is gedaan, omdat het toepassen van DSDM te omvangrijk is voor een project dat in een periode van een halfjaar afgerond diende te worden. De twee gebruikte onderdelen zijn MoSCoW (zie paragraaf 4.1) en timeboxing (zie paragraaf 4.2).

4.1 MoSCoW

MoSCoW is een acroniem voor de manier waarop prioriteiten aan de vereisten worden toegekend. De rest van het woord staat voor:

***M**ust have*: voor vereisten die fundamenteel zijn voor het systeem.

***S**hould have*: voor belangrijke vereisten die bij een ontwikkeling binnen een minder strak tijdschema waarschijnlijk als verplicht aangemerkt zouden worden.

***C**ould have*: voor vereisten die makkelijk uit de te ontwikkelen stap kunnen worden weggelaten.

***W**ould like to have but will not have this time around*: voor waardevolle vereisten die kunnen wachten totdat de verdere ontwikkeling plaats vindt.

4.2 Timeboxing

Binnen DSDM wordt een timebox beschouwd als de tijd tussen de begin- en einddatum van een periode waarvan aan het eind ‘iets’ wordt afgeleverd. Dit ‘iets’ kan bijvoorbeeld zijn een analysemodel, deel van een frontend of functionaliteit. Dit levert een aantal vaste deadlines op van producten die worden afgeleverd.

Timeboxing wordt uitgevoerd in combinatie met de MoSCoW methode (zie paragraaf 4.1). Dit gebeurt door in elke timebox een aantal functionaliteiten te plannen. Deze functionaliteiten krijgen een prioriteit volgens de MoSCoW methode. Binnen elke timebox moet dus de *must have* afgerond zijn en bij een minder strakke planning worden vervolgens de *should have* functionaliteiten ontwikkelt. Hierdoor wordt voorkomen dat er kostbare tijd wordt gestoken in de niet *must* have functionaliteiten.

In de planning (zie bijlage A: *Plan van aanpak*) is dit toegepast door fasen en deadlines voor (deel)producten vast te leggen.

4.3 Uitvoering

Het afstudeerproject werd in aantal fasen doorlopen. Hierdoor was het te doorlopen traject overzichtelijker. In de volgende paragrafen zullen alle fasen beschreven worden.

4.3.1 Opstartfase

In de opstartfase is er gewerkt aan het concreet opzetten van het project. Hierin is de opdracht vastgesteld en is er een planning gemaakt voor de duur van het project.
Aan het eind van deze fase is er een goedgekeurd plan van aanpak opgeleverd aan de bedrijfsbegeleider, docentbegeleider en de afstudeeradministratie van de Hogeschool Utrecht.

4.3.2 Onderzoeksfase

Na afronding van de opstartfase, lag er een concreet plan klaar voor het aanpakken van het afstudeerproject. De volgende stap was het zoeken naar antwoorden op de deelvragen (zie paragraaf 3.3). De antwoorden leidde, gezamenlijk, uiteindelijk tot het antwoord op de hoofdvraag van het afstudeerproject.

Om specifieker op de deelvragen in te gaan en het onderzoek te leiden waren er een aantal onderzoeksvragen gedefinieerd. Om het overzichtelijk te houden zijn de onderzoeksvragen in een aantal onderwerpen onderverdeeld. Voor dit afstudeerproject waren de volgende onderzoeksvragen per onderwerp gedefinieerd:

- **Android**
  - Wat is een aanbevolen ontwikkelomgeving voor Android?
  - Hoe is een Android applicatie opgebouwd?
  - Hoe zijn de sensoren te bereiken vanuit Android?
  - Hoe is een graphical user interface (GUI) te maken in Android?
- **Augmented reality**
  - Hoe is augmented reality toe te passen op de Google Phone 1 (G1)?
- **Lokalisatie**
  - Hoe werkt het digitaal kompas?
  - Welke manieren zijn er om de locatie te bepalen?
  - Wat is de beste manier om de locatie te bepalen?
- **Uitnodigend**
  - Wat maakt een applicatie interessant voor de eindgebruiker?
  - Wat zijn manieren, door het toepassen van augmented reality, om de applicatie uitnodigend te maken?
- **Communicatie**
  - Hoe moet de communicatie tussen de Android applicatie en server applicatie opgezet worden?

De resultaten die voortkwamen uit de hierboven genoemde onderzoeksvragen zijn uitgewerkt in een onderzoeksrapport. In hoofdstuk vijf worden de resultaten besproken. In bijlage B is het hele onderzoeksrapport bijgesloten.

4.3.3 Ontwerpfase

Na de onderzoeksfase was de benodigde kennis opgedaan. Deze kennis werd gebruikt om de applicaties te ontwerpen. In deze fase zijn twee documenten opgeleverd waarin staat beschreven hoe de applicaties eruit gaan zien.

Het eerste document is een functioneel ontwerp hierin zijn de volgende zaken behandeld:

- Beschrijving van de functionaliteiten;
- Relevante data;
- Datastroom tussen de relevante componenten;
- Concept screenshots.

Het functioneel ontwerp diende ter grondslag voor het tweede document; het technisch ontwerp. In het technisch ontwerp zijn onder andere de volgende zaken behandeld:

- Gebruikte hard- en software;
- Gebruikte protocollen;
- Gebruikte framework;
- Architectuur;
- Werking van het systeem door middel van sequence diagrammen.

4.3.4 Realisatiefase

Met de kennis van Android uit het onderzoeksrapport, het functioneel- en technisch ontwerp werd er begonnen aan de realisatiefase. Het proces is incrementeel en iteratief doorlopen.

Per iteratie is er gekeken of het eerder gemaakt ontwerp nog overeenkwam, dit om incompatibiliteit tussen de incrementen te voorkomen. Vervolgens werd het increment gerealiseerd en getest. Na het succesvol testen werd het increment geïmplementeerd in het deelproduct. Als het increment werkte in het deelproduct, werd de volgende iteratie uitgevoerd.

4.3.5 Afrondingsfase

In deze fase wordt een demonstrator van de applicaties gemaakt waarmee ten alle tijde een demo kan worden gegeven. Ook de factsheet moet bijgewerkt worden met de resultaten van het project.

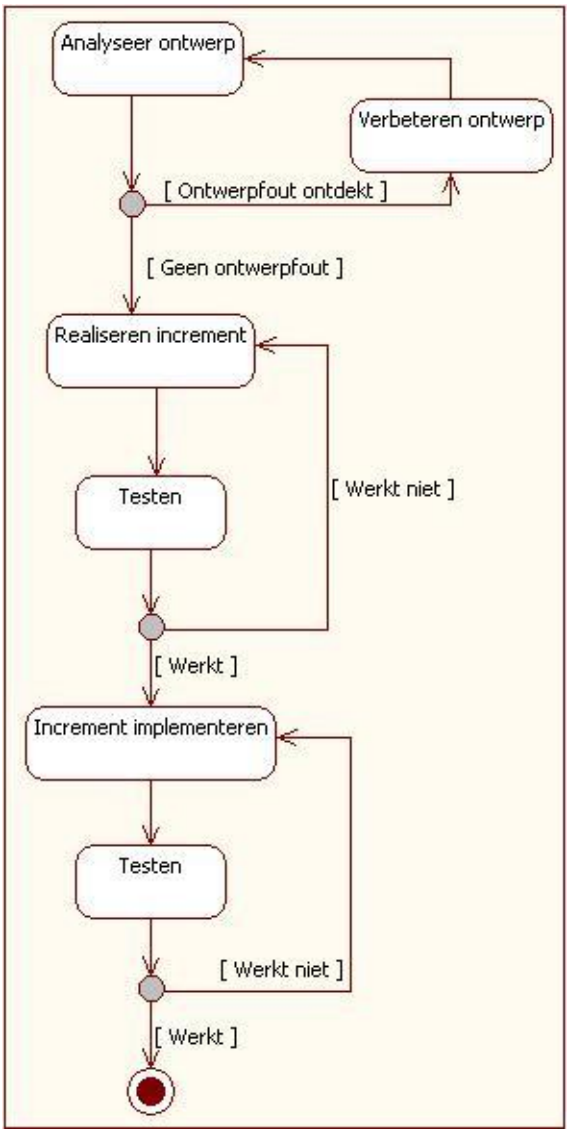


Diagram 4.1: Cyclus van een iteratie

5 Resultaten onderzoek

In dit hoofdstuk worden de bevonden resultaten van het onderzoek besproken. In de volgende paragrafen wordt ingegaan op de kernpunten van het onderzoek. Het volledige onderzoeksrapport is bijgesloten als bijlage B.

5.1 Ontwikkelomgeving

De aanbevolen ontwikkelomgeving bestaat uit een drietal componenten. Het betreft hier een Integrated Development Environment (IDE), Android Development Tools (ADT) en de G1 telefoon.

5.1.1 Integrated Development Environment (IDE)

In de Android Developer guide[7] van Google is veel ondersteuning te vinden voor Eclipse[8] als IDE. Eclipse is een open source framework voor software ontwikkelomgevingen. Door middel van een plugin (ADT) is de IDE zeer geschikt voor het ontwikkelen van Android applicaties.

5.1.2 Android Development Tools (ADT)

ADT is een plugin voor Eclipse, welke extensies toevoegt aan de IDE. Deze extensies maken het ontwikkelen van Android applicaties soepeler en sneller. Met de debugging tool die meekomt is het mogelijk om de applicatie te debuggen. In deze modus valt onder andere proces informatie (en eventuele foutmeldingen) te zien van de applicatie. Als er een Android telefoon is aangesloten en de bijbehorende driver geïnstalleerd, dan wordt de applicatie automatisch op de telefoon gedraaid.

5.1.3 G1 telefoon

Om de Android applicatie te testen op een echte telefoon is de G1 beschikbaar gesteld. Dit is het eerste mobiel die op Android draait. Het toestel beschikt over onder andere een GPS sensor, digitaal kompas en camera. Deze drie onderdelen zijn nodig om de Android applicatie te laten functioneren zoals deze bedoeld is.

5.2 Het Android platform

Bij Android staat het gebruik van elementen uit andere applicaties, mits deze applicaties dit toestaan, centraal. Daarom is Android uit een aantal componenten opgebouwd die gestart kunnen worden vanuit elk applicatieproces. In de volgende paragrafen wordt hierop kort ingegaan.
Dit hoofdstuk geeft antwoord op de deelvraag:

"Hoe werkt het Android platform wat betreft de applicatie opbouw en sensoren?"

Voor de sensoren wordt u doorverwezen naar bijlage B: *Onderzoeksdocument*.

5.2.1 Activities

Een activity presenteert een venster met daarin één of meerdere views. De visuele inhoud van een venster word bepaald door de structuur van een aantal view-objecten. De parent view organiseert de layout van zijn children. De children views zijn user interface elementen.

5.2.2 Services

Een service component wordt gebruikt als achtergrondproces. Het wordt gebruikt voor langlopende taken. Hier kan bijvoorbeeld worden gedacht aan het afspelen van muziek terwijl de gebruiker andere applicaties aan het gebruiken is.

5.2.3 Content providers

Een content provider maakt het mogelijk om bepaalde data beschikbaar te stellen tussen verschillende applicaties. De data staat opgeslagen in het bestandssysteem of de ingebouwde SQLite database. Zo kunnen bijvoorbeeld contacten in de telefoon opgevraagd worden door andere applicaties zonder het apart in te voeren.

5.2.4 Broadcast receivers

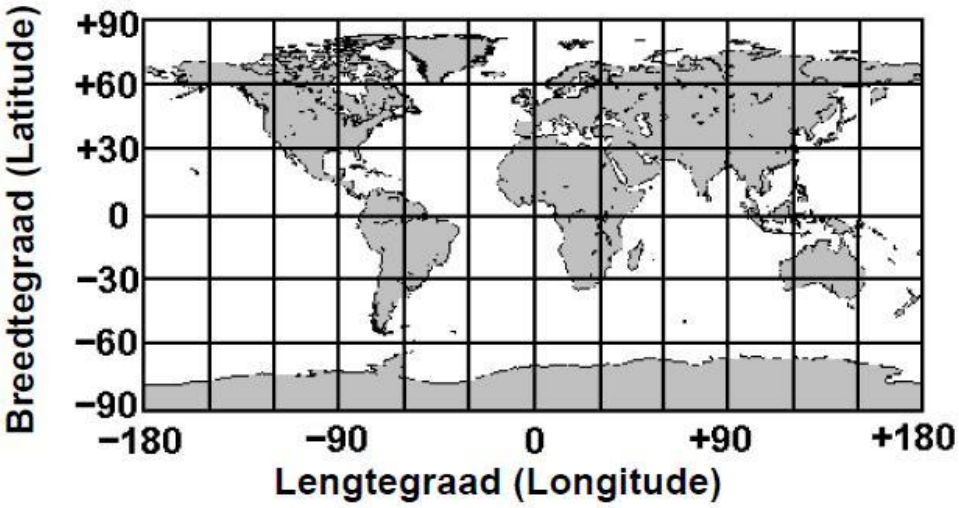
Een broadcast receiver is een component dat niets anders doet dan het ontvangen en reageren op broadcast announcements. Broadcast announcements kunnen van Android zelf komen of van andere applicaties. Applicaties die toegang hebben tot de receivers kunnen tot bepaalde acties overgaan als er een announcement wordt ontvangen.

5.3 Locatie bepalen

De locatie van het G1 toestel wordt aangegeven door de combinatie van een lengte- en breedtegraad.

De lengtegraad lijnen lopen verticaal over de aardbol. De nullijn is de Greenwich meridiaan. De lengtegraad geeft de locatie in de west-oost richting aan. Ten westen van de Greenwich meridiaan worden de graden negatief gemeten en ten oosten positief (zie figuur 5.1).

De breedtegraad lijnen lopen horizontaal over de aardbol. De nullijn is de evenaar. De breedtegraad geeft de locatie in noord-zuid richting aan. Ten noorden van de evenaar zijn de waarden positief en ten zuiden van de evenaar zijn de waarden negatief (zie figuur 5.1).



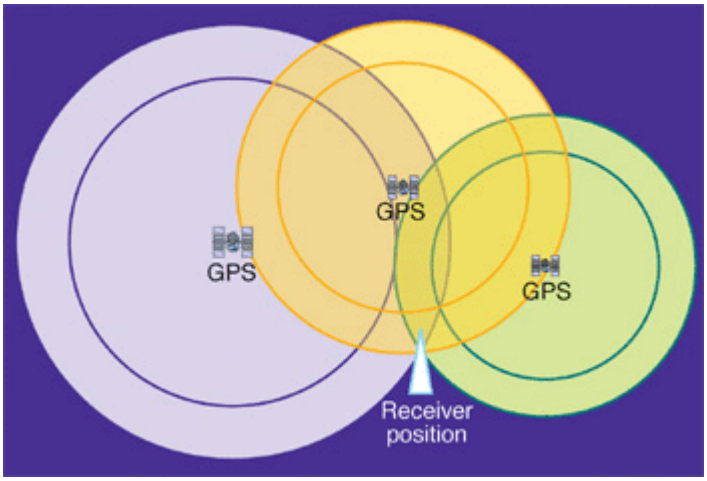
Figuur 5.1: Positiebepaling lengte- en breedtegraad

(Bron: Van Eenbergen, F. (2009). "Augmented Hyves op Google Android")

Voor het bepalen van de lengte- en breedtegraad zijn een tweetal methoden beschikbaar die toe te passen zijn op het Android platform. Deze zullen beide beschreven worden en met elkaar vergeleken.

5.3.1 Locatie bepalen via GPS

Om de locatie te kunnen bepalen via het GPS systeem zijn er minimaal 3 satellieten nodig. Deze satellieten zenden radiosignalen uit die door de GPS ontvanger worden opgevangen. Door de tijd te berekenen tussen ontvangst en uitzenden van het radiosignaal kan de ontvanger de afstanden tot de satellieten bepalen en kan aan de hand van de afstanden, de positie bepalen.



Figuur 5.2: Locatie bepalen via GPS
(Bron: http://www.aero.org/publications/crosslink/summer2002/02.html)

Voordelen:

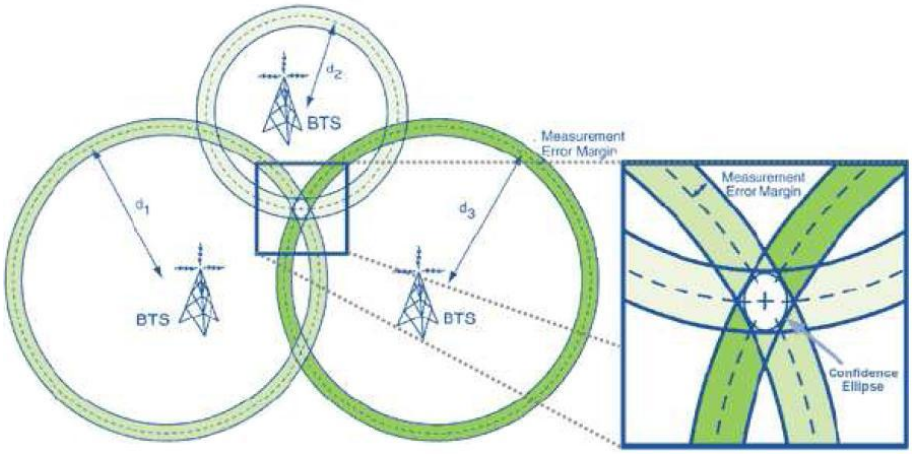
- Het werkt over bijna de hele wereld.
- Het haalt gemiddeld een nauwkeurigheid tussen de tien en twintig meter[9].

Nadelen:

- De tijd om een locatie te bepalen kan oplopen tot drie minuten[10].
- Binnen gebouwen is meestal geen GPS signaal te ontvangen.
- De GPS ontvanger gebruikt veel stroom, waardoor de accu van de telefoon snel leegloopt.

5.3.2 Locatie bepalen via GSM masten

Van deze techniek kan gebruik worden gemaakt indien er toegang is tot de data waar aan elk ID van een GSM mast een lengte- en breedtegraad is gekoppeld. Net zoals bij GPS wordt de tijd gebruikt om de afstand te berekenen. In combinatie met de locatie van de GSM mast kan de locatie worden vastgesteld van het toestel.



Figuur 5.3: Locatie bepalen via GSM
(Bron: Van Eenbergen, F. (2009). "Augmented Hyves op Google Android")

Voordelen:

- Werkt in theorie overal waar GSM-masten staan.
- Er wordt redelijk snel een locatie verkregen[11].
- Lage energie verbruik.
- Werkt in gebouwen.

Nadelen:

- Lage nauwkeurigheid welke varieert tussen de 500 en 3000 meter.
- Minimaal drie GSM-masten nodig.

5.3.3 Locatie bepalen via WLAN

Bij positiebepaling via WLAN (**W**ireless **L**ocal **A**rea **N**etwork) wordt een database met WLAN access points geraadpleegd. Aan de hand van het SSID (**S**ervice **S**et **Id**entifier) van een access point kan uit de database de locatie worden opgehaald. Hiermee kan vervolgens de locatie van het apparaat worden berekent.

Voordelen:

- Snel opvragen van locatie.
- Kost minder stroom.
- Werkt in gebouwen.

Nadelen:

- Verkregen locatie (van de access point) kan verouderd zijn als een access point is verplaatst, maar niet doorgegeven aan de database.
- Meerdere access points nodig om een accurate locatie te bepalen.
- Zeer beperkt beschikbaar in de open lucht

5.3.4 Conclusie

De deelvraag luidde:

"Hoe kan augmented reality worden toegepast op de Android applicatie?"

Voor augmented reality is locatiebepaling nodig. Naar aanleiding van het onderzoek is er om een aantal redenen voor GPS gekozen.
WLAN valt al snel af, omdat deze maar zeer beperkt beschikbaar is. Daarbij komt dat er een database beschikbaar moet zijn om de lengte- en breedtegraad van de access point te achterhalen. Ook de locatiebepaling via GSM masten is afgevallen. Hiervoor zijn twee redenen. Ten eerste is de nauwkeurigheid te laag om het te kunnen gebruiken. Daarnaast is (ook) hier data nodig die aan elke GSM mast een lengte- en breedtegraad heeft gekoppeld. Grote voordelen van locatiebepaling via GPS is het feit dat deze veel nauwkeuriger is dan de andere twee methoden. Het bepalen van de locatie via GPS kan (vooral bij de eerste keer) weliswaar lang duren, maar daartegenover staat de beschikbaarheid van het netwerk over bijna de hele wereld.

5.4 Uitnodigend

Als de Android applicatie in staat is om te ondersteunen in het rapporteren van problemen en schade is nog niet bepaald of de applicatie zal worden gebruikt. De Android applicatie zal aantrekkelijk moeten zijn voor de burgers om het te gaan gebruiken. Hier zijn een aantal eigenschappen mee gemoeid die van invloed zijn om de Android applicatie goed te laten functioneren. Een goed functionerende applicatie zal immers sneller worden geaccepteerd en is aantrekkelijker dan een minder functionerend model.

In de volgende paragrafen zal dieper worden ingegaan op de eigenschappen die van invloed zijn op een applicatie.

5.4.1 Eigenschappen

Om de belangrijke eigenschappen voor mobiele applicaties te bepalen zijn een aantal meningen[12] [13] [14] [15] [16] [17] van personen bekeken. Na analyse hiervan kwamen een aantal eigenschappen naar voren, welke onderverdeeld zijn in functionele en overige eigenschappen.

5.4.1.1 Functionele eigenschappen

Graphical User Interface (GUI)
Een gebruiker neemt geen tijd om een handleiding door te lezen hoe een mobiele applicatie gebruikt moet worden. De interface moet dus intuïtief te gebruiken zijn. Ook het geven van teveel opties in één scherm moet vermeden worden, dit kan verwarrend overkomen op de gebruiker.

Foutmeldingen
Het is onmogelijk om alle fouten te voorspellen in een applicatie, daarom is het belangrijk om foutmeldingen op de juiste manier weer af te vangen en weer te geven. Een gebruiker weet niet wat er aan de hand is en foutmeldingen moeten daarom informeren wat het probleem is en wat de gebruiker kan doen om het te herstellen of voorkomen.

Doel
De applicatie moet uiteindelijk helpen bij het makkelijker uitvoeren van taken. De applicatie moet het niet ingewikkelder maken.

5.4.1.2 Overige eigenschappen

Snelheid
Bij snelheid van een (mobiele) applicatie wordt bedoeld de tijd die nodig is om acties van de gebruiker af te handelen. De Android applicatie is niet bedoeld om gebruikers langdurig te vermaken, maar om gebruikers de mogelijkheid te geven om op eenvoudige manier een probleem te melden.

Gebruik maken van de mogelijkheden
Het trekt gebruikers als een applicatie gebruik maakt van de mogelijkheden die het G1 toestel biedt. Hier moet dus worden gedacht aan onder andere de GPS sensor en digitaal kompas. Door gebruik van de sensoren wordt het geen statische applicatie, maar biedt het diversiteit afhankelijk van de positie.

Dynamische content
Door gebruik te maken van de gegevens die de sensoren bieden of gebruik te maken van internet is het mogelijk om dynamische content te bieden. Dit prikkelt de gebruiker om de Android applicatie vaker te gebruiken.

5.4.2 Augmented reality

Om te achterhalen hoe augmented reality een applicatie aantrekkelijk maakt zijn een drietal '*top augmented reality applications*' lijsten vergeleken[19, 20, 21]. Uit alle drie de lijsten is te zien dat het toevoegen van relevante informatie over de omgeving via augmented reality naar voren komt. Bij dit project draait het om hoe een applicatie aantrekkelijk te maken, zodat de burgers problemen en schade gaan melden. Er kan daarom een laag worden gerealiseerd die de status weergeeft van een eerder gemeld probleem. Hieraan kan de eindgebruiker zien wat de status is van het probleem, bijvoorbeeld: 'Wordt gerepareerd'. In één van de drie lijsten komen wel een paar applicaties voor die spelelementen hebben. Wat toegepast kan worden is het verzamelen van punten.

5.4.3 Conclusie

In dit hoofdstuk is er gekeken naar de volgende deelvraag:
"Welke elementen maakt de Android applicatie aantrekkelijk voor de eindgebruiker?"

Om op algemeen gebied de applicatie interessant te maken voor de eindgebruikers, moet deze aan een aantal eigenschappen voldoen. Technisch gezien moet de applicatie snel reageren om zo min mogelijk tijd van de eindgebruiker te vragen en maakt gebruik van de mogelijkheden van de sensoren waar dit mogelijk is in de applicatie. Om de eindgebruiker te motiveren om de applicatie te blijven gebruiken is gebruik van dynamische content een goede optie.

Functioneel moet de GUI zo simpel mogelijk en intuïtief worden opgebouwd. Dit om hulp bij het gebruik van de Android applicatie zo minimaal mogelijk te houden. Daarnaast dienen foutmeldingen zo te zijn geformuleerd dat de eindgebruiker weet waar hij aan toe is als er iets onverwachts gebeurd.

Veel applicaties voegen een laag toe aan de reële beelden die informatie tonen die relevant is voor de eindgebruiker. In dit project is dat toe te passen door de status van de melding weer te geven. Hierdoor wordt tevens de kans op dubbele meldingen verkleind, want de eindgebruiker weet dat het is gemeld.
Als laatst is het combineren van een maatschappelijke relevante applicatie met spelelementen een extra stimulering. In dit project is het kunnen verdienen van punten een goede aanvulling om er een verleidelijke applicatie van te maken.

5.5 Client / server communicatie

Om de locatie en de foto van de Android applicatie naar de server over te kunnen sturen moet er communicatie mogelijk zijn. Voor dit project is er gekeken naar XMPP (E**x**tensible **M**essaging and **P**resence **P**rotocol) en HTTP (**H**yper**t**ext **T**ransfer **P**rotocol). In de volgende paragrafen wordt kort op beide protocollen ingegaan en wordt gekeken en welke het beste past bij het project.

5.5.1 Extensible Messaging Presence Protocol (XMPP)

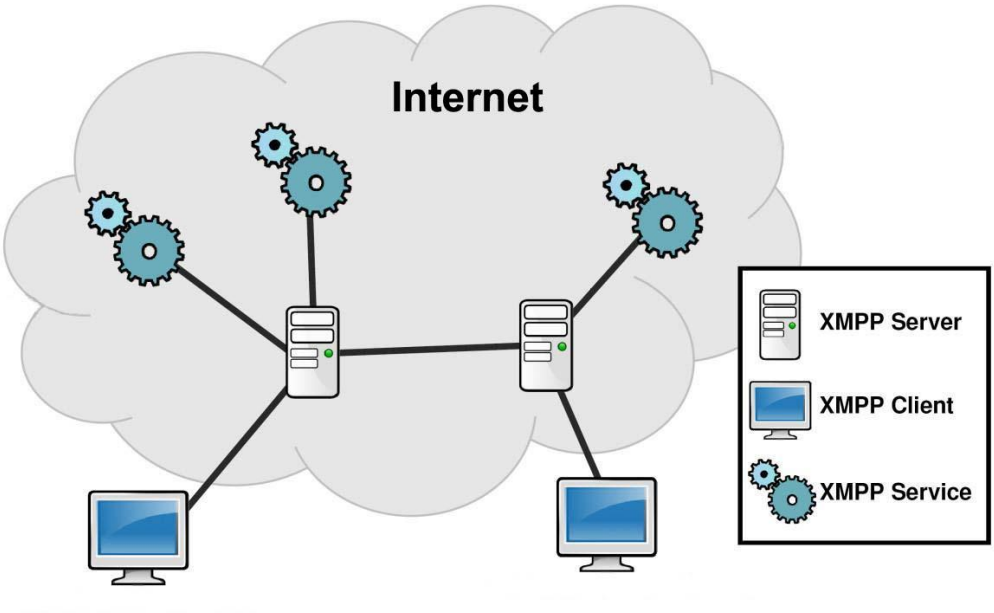
XMPP[18] is een open standaard gebaseerd op XML (E**x**tensible **M**arkup **L**anguae). Van origine was het bedoeld voor real-time communicatie voor instant messaging en presence notification. Door verschillende uitbreidingen is XMPP steeds uitgebreider in te zetten voor onder andere multi-party chat en voice / video calls.
Bij XMPP wordt gebruik gemaakt van een stateful verbinding, wat betekent dat er een constante verbinding is en bij het verzenden van een bericht niet opnieuw een verbinding tot stand hoeft te worden gebracht. Deze eigenschap kan worden toegepast bij het versturen van updates (status van melding) of als alle rewards van de server opgevraagd dienen te worden. Door de steeds kleine berichten die worden verstuurd is er wel alvast data binnen. Mocht de verbinding wegvallen dan is niet gelijk alle data verloren (HTTP request bevat wel alle data).

In de volgende situatie wil *bob@example.com* een bericht versturen naar *alice@logica.com*. Bobs XMPP client verstuurt het bericht naar de *example.com* server. *Example.com* server opent een verbinding met de *logica.com* server. Vervolgens zal de *logica.com* server het bericht afleveren aan de client van Alice (zie figuur 5.4).

Figuur 5.4: XMPP chat voorbeeld

De XMPP services hebben betrekking op uitbreidingen die aan de servers toegevoegd kunnen worden. In de bovengenoemde situatie zal een XMPP service kunnen zijn dat Bob een leesbevestiging wil van Alice.

In figuur 5.5 is een voorbeeld te zien van een XMPP netwerk.



Figuur 5.5: Voorbeeld van een XMPP netwerk[19]

5.5.2 HiGrids

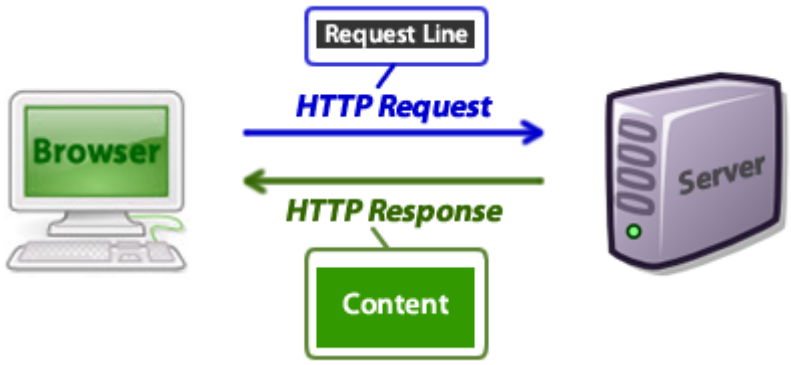
Als XMPP gebruikt zou worden, zoals in paragraaf 5.5.1 omschreven, dan moet vrij veel gedaan worden aan de server kant voor dit project. Echter, binnen Logica is een project bezig onder de titel ‘HiGrids’. In HiGrids wordt gebruik gemaakt van XMPP als communicatie protocol tussen de clients.

Om te bepalen of het mogelijk is om beide projecten met elkaar te combineren is gekeken naar de eigenschappen van de server kant van het HiGrids project.
HiGrids maakt gebruik van accounts voor elke gebruiker. Dit betekent theoretisch dat de eindgebruiker van de Android applicatie zich eerst moet aanmelden alvorens hij iets kan melden. Echter, is de server in staat om een account te creëren zonder de eindgebruiker erbij te betrekken.
Door gebruik van XML zijn berichten makkelijker te parsen. Dit komt door de structuur van XML bestanden. Dit kan voornamelijk van toepassing zijn voor de notificaties voor de rewards.
Een andere eigenschap van de XMPP server is dat berichten aan de hand van een namespace worden gerouteerd. Hierdoor is het vrij makkelijk om berichten door te sturen naar de juiste server. In dit project is het toe te passen door bijvoorbeeld meldingen die te maken hebben bushokjes door te sturen naar de busmaatschappij.
Om de foto’s te versturen naar de server is er een extra uitbreiding nodig voor XMPP. In het HiGrids project is er nog geen mogelijkheid om bestanden te versturen.

5.5.3 HTTP

HTTP is een protocol die een standaard heeft beschreven hoe clients data aanvragen en hoe servers hierop moeten reageren. Het wordt gebruikt door programma's om te communiceren via het wereldwijde web. In tegenstelling tot XMPP moet bij elk bericht een nieuwe verbinding worden opgezet.

In afbeelding 5.6 is te zien hoe de communicatie tussen een client en een server verloopt.



Figuur 5.6: HTTP protocol[20]

In Android is er al standaard de mogelijkheid om HTTP connecties op te zetten en data via deze connecties te versturen.
Er is geen identificatie nodig hierdoor kan iedereen zonder zich aan te melden gebruik maken van de dienst. Hierdoor is het wel minder toegankelijk om te weten welke gebruiker een probleem meldt.

5.5.4 Conclusie

De deelvraag die is onderzocht luidde:

> *"Wat zijn de mogelijkheden om de onderhoudsdienst op de hoogte te brengen van de problemen?"*

Het doel van de applicatie is dat iedere eindgebruiker op elk moment een probleem kan melden. XMPP is account gebaseerd en dat betekent dat elke keer als een eindgebruiker een probleem wil melden zich moet aanmelden. XMPP servers zijn in staat om een account te maken voor gebruikers die zich voor het eerst aanmelden. Deze account gegevens kunnen vervolgens gebruikt worden om bij te houden wie welke meldingen doet en welke rewards heeft gekregen.

Door de stateful verbinding bij XMPP wordt er eenmalig afgesproken tussen server en client welke definities worden gebruikt (bijvoorbeeld: versie, encoding). Hierna volgen steeds kleine XML berichten. Dit maakt het makkelijker om het berichtenverkeer bij te houden bij de server en de Android applicatie. Wat resulteert in een nettere source code van de Android applicatie en betere onderhoudbaarheid, omdat de XML berichten object georiënteerd zijn. Bij HTTP dient er in Android een HTML (**H**yper**T**ext **M**arkup **L**anguage) formulier opgebouwd te worden welke dan vervolgens verstuurd wordt. Dit is minder goed leesbaar en te onderhouden dan een object georiënteerde opmaak in XML.

Doordat niet alle data in één response wordt verstuurd, is er bij een verbreking van de verbinding niet alle data verloren. Dit voorkomt onnodige data overdracht, omdat de binnengekomen data ook niet opnieuw opgevraagd / verstuurd hoeft te worden.

Het versturen van de foto is een belangrijk onderdeel van de Android applicatie. Bij XMPP moeten plugins het versturen van bestanden mogelijk maken. Als dit via HTTP gaat is er al de mogelijkheid om dit te doen.

Op grond van de bovengenoemde aspecten is er besloten om voor XMPP te gaan. Er is hier voor gekozen, omdat XMPP over eigenschappen beschikt die het systeem overzichtelijker maakt. Daarnaast is in combinatie met het HiGrids project een beter toekomstperspectief als er voor XMPP word gekozen. Redenen hiervoor zijn de standaard berichten routering van de XMPP server en de nettere source code. Ook sluit het project mooi aan op de mogelijkheden van HiGrids.

6 Ontwerp

In dit hoofdstuk wordt het ontwerp van het systeem beschreven. Het is opgedeeld in twee hoofdstukken: functioneel ontwerp en technisch ontwerp.
In het functioneel ontwerp (paragraaf 6.1) worden de functionaliteiten van het systeem (Android applicatie en server applicatie) behandeld. Er zijn een aantal diagrammen gemaakt ter ondersteuning voor de ontwikkeling van beide applicaties.
In het technisch ontwerp (paragraaf 6.2) worden de technieken behandeld om de functionaliteiten uit het functioneel ontwerp te ontwikkelen. Ook hier zijn een aantal diagrammen gemaakt ter ondersteuning van het ontwikkelproces.

6.1 Functioneel ontwerp

6.1.1 Functionaliteiten

In tabel 6.1 is een overzicht te zien met alle functionaliteiten voor het systeem. Er zijn vier categorieën met ieder een aantal functionaliteiten. Aan elke functionaliteit is een MoSCoW prioriteit gegeven.

Iteratie	MoSCoW
Camera: • Beeld ontvangen • Beeld weergeven • Camera bedienen (inzoomen, uitzoomen) • Filters ten behoeve van beeldbewerking	 Must have Must have Should have Could have
Lokalisatie: • Verkrijgen positie • Kompas implementeren • Positie binnen gebouwen	 Must have Must have Should have
Augmented Reality: • Popup box voor probleemomschrijving • Interactie met de gebruiker (bv: beloning toekennen) • Status probleem • Problemen in de buurt laten zien • Kompas weergeven (plaatje van kompas + kijkrichting) • Highlighten object	 Must have Must have Should have Could have Could have Would have
Communicatie: • Verzenden / ontvangen locatie en probleemomschrijving • Weergeven locatie en probleemomschrijving (server) • Verzenden / ontvangen foto • Ophalen problemen in de buurt (Android applicatie) • Positie weergeven op Google maps (server)	 Must have Must have Must have Should have Could have

Tabel 6.1: Functionaliteiten volgens MoSCoW

Diagram 6.1 geeft een overzicht van alle requirements die het systeem heeft voor een werkbare proof of concept welke alle *must have* prioriteiten bevat.

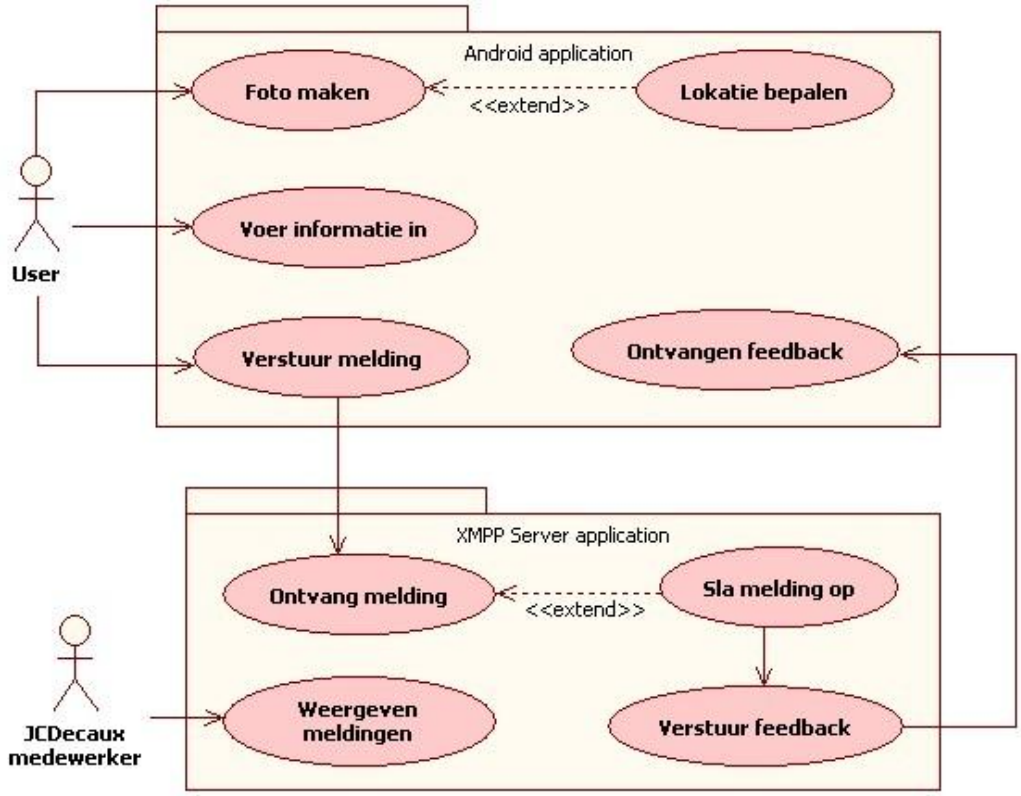


Diagram 6.1: Use case diagram:

In de volgende tabellen wordt elke use case beschreven aan de hand van een aantal labels. Dit geeft inzicht hoe het systeem werkt.

Naam	Foto maken
Samenvatting	Er wordt een foto gemaakt van het probleem
Actor	User
Precondities	1. Optie nieuwe melding geselecteerd 2. GPS is ingeschakeld
Beschrijving	1. De gebruiker richt de camera op het probleem 2. De gebruiker klikt op het camera icoontje; foto wordt gemaakt 3. De applicatie vraagt of de foto goed is 4. Voer use case “Locatie bepalen” uit 5. Voer use case “Voer informatie in” uit
Uitzonderingen	[Foto wordt afgekeurd door gebruiker] Gebruiker moet nieuwe foto maken. [Camera kapot] Als de Android applicatie geen verbinding met de camera kan krijgen, geef dit in een melding weer.
Postconditie	Gebruiker heeft een foto gemaakt

Tabel 6.2: Use case 'Foto maken'

Naam	Voer informatie in
Samenvatting	Gebruiker voegt extra informatie toe
Actor	User
Precondities	1. Er is een foto gemaakt
Beschrijving	1. Gebruiker selecteert categorie 2. Gebruiker voert opmerking in 3. Gebruiker klikt op ‘Verstuur melding’
Uitzonderingen	[Geen categorie gekozen] Laat de gebruiker weten dat er geen geldige categorie is gekozen. [Geen opmerking ingevoerd] Een opmerking is niet verplicht, ga door naar use case “Verstuur melding”.
Postconditie	Er is een melding met foto en extra informatie

Tabel 6.3: Use case 'Voer informatie in'

Naam	Verstuur melding
Samenvatting	Het systeem verstuurt de melding naar de server
Actor	User
Precondities	1. Er is op de knop ‘Verstuur melding’ geklikt
Beschrijving	1. Android applicatie controleert of locatie is bepaald 2. Android applicatie verstuurt foto, informatie en locatie naar de server
Uitzonderingen	[Locatie is (nog) niet bepaald] Laat de gebruiker weten dat de locatie wordt bepaald. Als er na vijf seconden geen locatie is bepaald geef dit aan bij de gebruiker. Use case “Locatie bepalen” informeert de gebruiker over het mogelijke probleem. [Server kan niet worden bereikt] Geef aan de gebruiker door dat de melding niet verstuurd kan worden. De Android applicatie slaat de melding lokaal op en verstuurt deze, zodra er een verbinding met de server is.
Postconditie	Melding is verstuurd naar de server

Tabel 6.4: Use case 'Verstuur melding'

Naam	Locatie bepalen
Samenvatting	Systeem bepaalt locatie
Actor	User
Precondities	1. Gebruiker heeft een foto gemaakt
Beschrijving	1. Android applicatie berekent de locatie van de telefoon 2. Android applicatie bepaalt de kijkrichting
Uitzonderingen	[Locatie kan niet worden berekent] Geef aan de gebruiker mogelijke problemen en oplossingen weer. Stuur de gebruiker terug naar de use case “foto maken”.
Postconditie	Locatie is bepaald

Tabel 6.5: Use case 'Locatie bepalen'

Naam	Ontvangen feedback
Samenvatting	Ontvangen van de feedback
Actor	User
Precondities	1. Verstuurde melding is ontvangen door de server 2. Server heeft feedback verstuurd
Beschrijving	1. Data uitlezen uit de verbinding 2. Ontvangen feedback weergeven
Uitzonderingen	[Fout tijdens ontvangen] Stuur bericht terug naar de server applicatie dat de overdracht is mislukt. De server stuurt vervolgens nog een keer de feedback. Na drie pogingen krijgt de gebruiker een bericht dat het ontvangen is mislukt. Na vijf minuten (of als de Android applicatie weer wordt gestart) wordt het opnieuw geprobeerd.
Postconditie	Feedback is geregistreerd en weergegeven

Tabel 6.6: Use case 'Ontvangen feedback'

Naam	Ontvang melding
Samenvatting	Server ontvangt de melding
Actor	User
Precondities	1. Android applicatie heeft de gemaakte melding verstuurd
Beschrijving	1. Server ontvangt de melding 2. Controleer of de melding alle verplichte data (foto, locatie en categorie) bevat.
Uitzonderingen	[Fout tijdens ontvangen] Stuur bericht naar de Android applicatie dat de overdracht is mislukt. De Android applicatie verstuurt de melding nog een keer. Na drie pogingen wordt de melding lokaal opgeslagen. Na vijf minuten of als de Android applicatie opnieuw word gestart, wordt het opnieuw geprobeerd.
Postconditie	Melding is ontvangen

Tabel 6.7: Use case 'Ontvang melding'

Naam	Sla melding op
Samenvatting	Sla de gegevens van de ontvangen melding op in de database
Actor	JCDecaux medewerker
Precondities	1. Melding is ontvangen
Beschrijving	1. Sla de ontvangen data van de melding op in de database 2. Bepaal de soort reward of feedback die de gebruiker krijgt 3. Sla in de database op welke soort reward er verstuurd moet gaan worden
Uitzonderingen	[Gebruiker heeft de soort reward al] Reward wordt geregistreerd zonder problemen. [Reward kan niet bepaald worden of probleem is al gemeld] Verstuur een feedback bericht waarin de gebruiker wordt bedankt en de reden waarom er geen reward is toegekend.
Postconditie	De melding is opgeslagen

Tabel 6.8: Use case 'Sla melding op'

Naam	Verstuur feedback
Samenvatting	Verstuur bericht naar de Android applicatie
Actor	JCDecaux medewerker
Precondities	1. Melding is opgeslagen in de database
Beschrijving	1. Vraag de soort reward op uit de database 2. Verstuur de reward / feedback naar de Android applicatie
Uitzonderingen	[Telefoon kan niet bereikt worden] Geef in database aan dat de reward niet verzonden is. Na drie pogingen stopt de server met het versturen van de reward / feedback. Als de Android applicatie weer verbinding heeft met de server, wordt het opnieuw geprobeerd.
Postconditie	Feedback / reward is verzonden naar de Android applicatie

Tabel 6.9: Use case 'Verstuur feedback'

Naam	Weergeven meldingen
Samenvatting	Geef de opgevraagde meldingen weer
Actor	JCDecaux medewerker
Precondities	1. Server heeft aanvraag voor weergeven van meldingen binnen gekregen
Beschrijving	1. Vraag de gevraagde meldingen op uit de database 2. Geef de meldingen weer
Uitzonderingen	[Er zijn geen meldingen] Geef aan dat er geen meldingen zijn.
Postconditie	De melding is opgeslagen

Tabel 6.10: Use case 'Weergeven meldingen'

6.1.2 Data flow

Het hele systeem bevat een datastroom tussen verschillende componenten. Er zijn vijf componenten die data met elkaar uitwisselen. Ze zijn van belang voor het melden van problemen, geven van rewards en het weergeven van de meldingen. Dit zijn de satellieten, de G1, de XMPP server, de daarbij behorende database en de computers bij JCDecaux.

Melden van een probleem

1. De satellieten zenden (GPS) radiosignalen uit die de G1 opvangt.
2. De eindgebruiker maakt een foto en voert informatie in (categorie, opmerking is optioneel).
3. Bij het verzenden verstuurt de G1 via XMPP een XML bericht met daarin de foto, locatie, categorie en de optionele opmerking.
4. De XMPP server ontvangt het bericht en slaat de melding op in de lokale database.
5. Als de melding is opgeslagen verstuurt de server een request naar de G1 met de feedback / reward.

Weergeven van de meldingen

1. JCDecaux vraagt via een browser om alle meldingen.
2. De server vraagt aan de database alle melding op.
3. De server geeft alle meldingen weer.

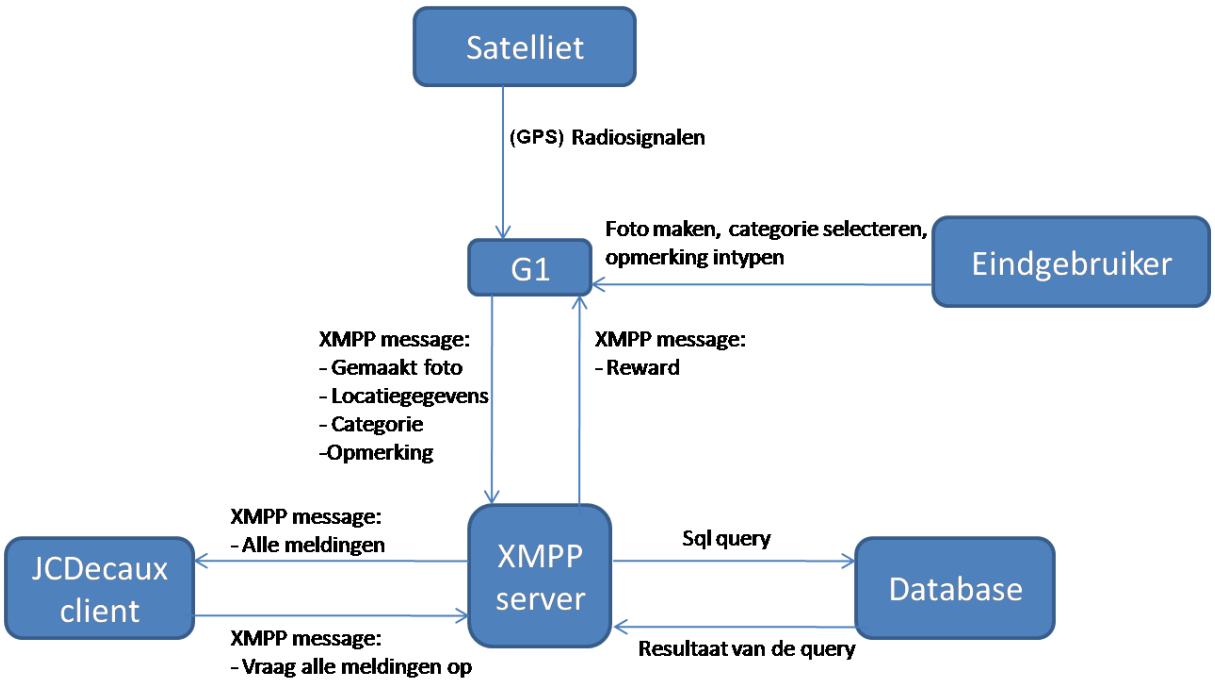


Diagram 6.2: Data flow diagram

6.1.3 Grafische interface

In het hoofdscherm staat alleen een knop om een nieuwe melding aan te maken. Door op deze knop te klikken wordt het proces om een probleem te melden gestart.

Figuur 6.1: Home scherm

Als er op de knop ‘Nieuwe melding’ is gedrukt krijgt de eindgebruiker een scherm te zien met de beelden van de camera. Onderin is een knop te zien met een camera icoontje erin. Door op dit icoontje te klikken wordt er een foto gemaakt en wordt het scherm in figuur 6.3 getoond.

Figuur 6.2: Camera scherm

Om een goede foto van het probleem op te sturen, krijgt de eindgebruiker de vraag of de gemaakte foto gebruikt moet worden. Als de foto wordt afgekeurd (rood kruisje), keert het scherm in figuur 6.2 weer terug. Als de foto gebruikt kan worden (groen vinkje) dan wordt het scherm in figuur 6.4 getoond.

Figuur 6.3: Scherm als er een foto is genomen

De volgende stap is om een categorie te kiezen. Dit helpt om de meldingen te categoriseren. Door op het grijze pijltje die naar beneden wijst te klikken, komt een lijst met categorieën in het scherm. Hiervan moet een categorie gekozen worden. Door op de knop ‘Voltooien’ te klikken, wordt de eindgebruiker naar de laatste stap geleid.

Het gele balkje bovenin geeft de voortgang weer.

Figuur 6.4: Categorie kiezen

Als de eindgebruiker nog iets kwijt wil over het probleem, kan deze stap gebruikt worden om een kleine tekst in te typen. Door op het vak te tikken, komt een scherm naar voren. In dit scherm kan een tekstje getypt worden. Dit scherm mag leeg gelaten worden.
Als de eindgebruiker om ‘Verstuur melding’ klikt wordt het verzend proces opgestart en krijgt de eindgebruiker het scherm in figuur 6.6 te zien.

Figuur 6.5: Opmerking toevoegen

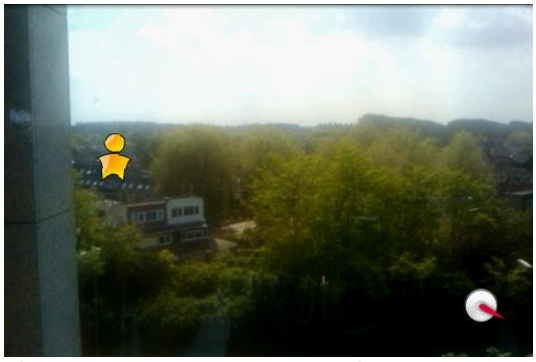
In het laatste scherm van het proces krijgt de gebruiker de voortgang van het verzenden te zien.
Zo weet de gebruiker wanneer de melding volledig is verzonden.

Hierna wordt de reward ontvangen en weergegeven.

Figuur 6.6: Voortgang van het verzenden

Als de reward is ontvangen wordt deze weergegeven op de plek van de melding.

In figuur 6.7 is de ‘buddy’ reward te zien.


Figuur 6.7: Weergeven reward

6.2 Technisch ontwerp

6.2.1 Openfire

Openfire[21] is een XMPP (E**x**tensible **M**essaging and **P**resence **P**rotocol; zie paragraaf 5.5.1) server. De server is geschreven in Java en is hierdoor platformonafhankelijk. Het beheer van de server wordt gedaan via een web interface.
Verder is Openfire uitgerust met een plugin architectuur. Hier is gebruik van gemaakt om meldingen op te slaan en weer te geven. Het opslaan gebeurt in de lokale database van Openfire (zie paragraaf 6.2.4).

6.2.2 ARCA

Het ARCA[22] (**A**ugmented **R**eality **C**ontext-**A**wareness) bestaat uit twee componenten. Een augmented reality component en een context awareness component. De augmented reality component in het framework voorziet de Android applicatie van een augmented reality view. De context awareness component voorziet de functionaliteit voor het weergeven van informatie (bijvoorbeeld de status van een melding). Naast deze twee componenten ondersteunt het framework ook de afhandeling van notificaties van de XMPP server. Om beide componenten te laten werk heeft het framework ook plaatsbepaling geïmplementeerd. Dit voorziet de Android applicatie van GPS locaties en de richting van de camera.

Figuur 6.8: Voorbeeld ARCA implementatie

6.2.3 Architectuur

6.2.3.1 Android applicatie

In diagram 6.3 is de architectuur van de Android applicatie weergegeven. Het bestaat uit een aantal packages. De packages *arca.** komen van het ARCA (**A**ugmented **R**eality **C**ontext **A**wareness; zie paragraaf 6.2.2) framework. Dit framework bestaat uit meer packages en klassen, maar de weergegeven klassen zijn direct van invloed op de *amgc.android* package. De *amgc.android* package bevat het ***A**ugmented **M**aintenaince **G**arbage **C**ontrol* applicatie voor Android.

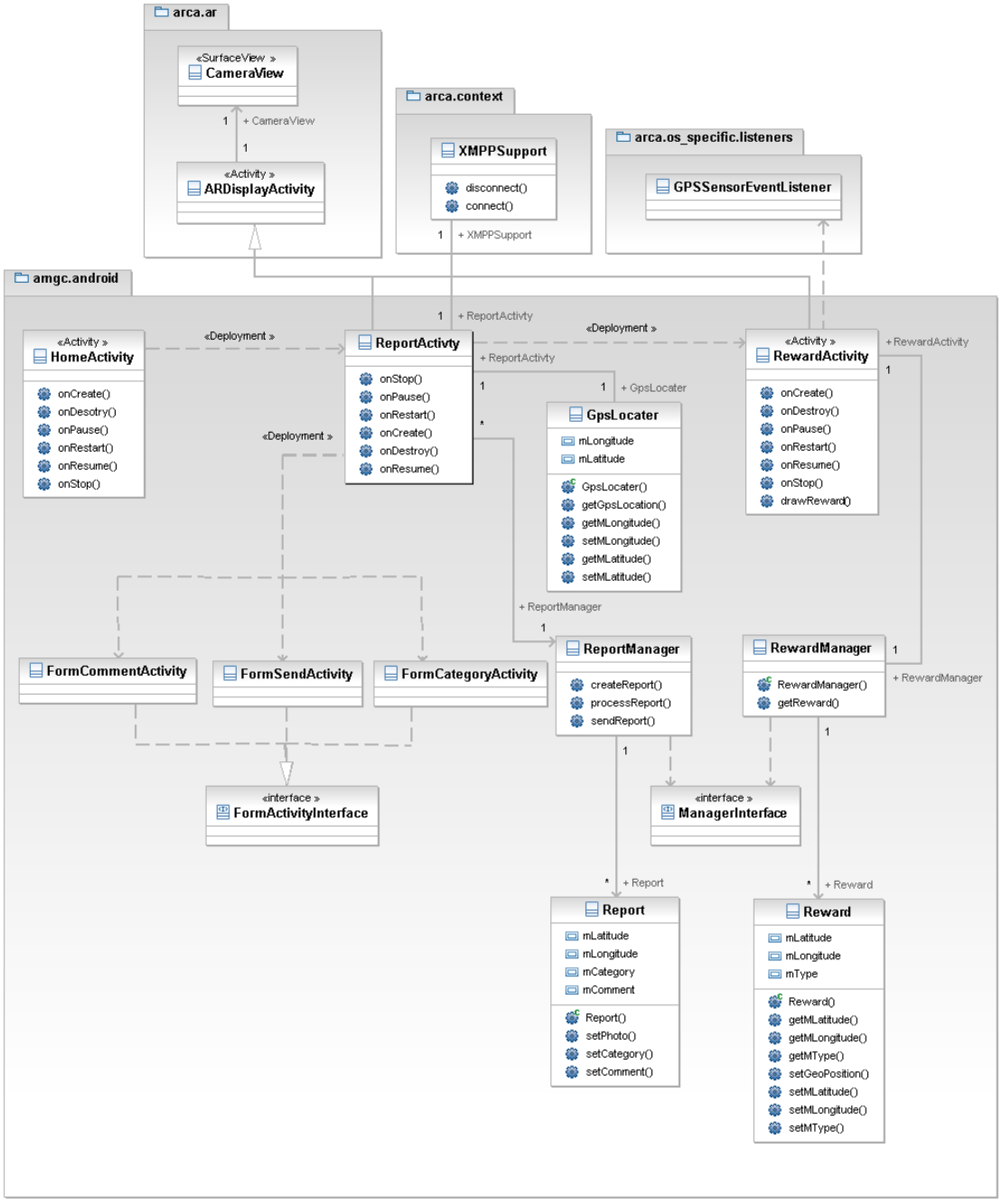


Diagram 6.3: Klassendiagram Android applicatie

arca.ar
Biedt functionaliteit om augmented reality toe te kunnen passen. Dit is mogelijk door het weergeven van de beelden van de camera, de GPS eventlistener en het digitaal kompas zijn geïmplementeerd in het ARCA framework. Ook is er een view om virtueel content op weer te geven over de beelden van de camera.

arca.context
Deze package biedt functionaliteit voor het XMPP protocol en voor het ontvangen van content welke in het virtuele view kan worden weergegeven. Het bevat de klasse *XMPPSupport* welke zorgt voor het juist opzetten van een verbinding met de XMPP server. Via deze klasse is het ook mogelijk om berichten te versturen naar de server. Het ontvangen van berichten wordt afgehandeld via een *NotificationEventHandler*.

amgc.android::HomeActivity
Deze klasse dient voor de weergave van het eerste scherm waarop verschillende UI (User Interface) elementen staan. Het handelt de tab en button events af die mogelijk zijn in dit scherm door de implementatie van de listeners.

amgc.android::ReportActivity
De *ReportActivity* klasse dient voor de weergave van het scherm voor het proces om een melding aan te kunnen maken. Door de implementatie van de *arca.ar::ARDisplayActivity* klasse worden de camerbeelden getoond. Het handelt alle schermen van de *FormActivity* af. Deze schermen sturen data terug en kan door middel van deze data returns het volgende scherm opstarten. Dus de klasse start bijvoorbeeld het scherm op om een categorie te selecteren. Als de categorie is geselecteerd, wordt dit teruggestuurd en wordt het scherm voor de opmerking opgestart.

amgc.android::FormActivity
Deze klasse biedt een interface die geïmplementeerd wordt door de klassen die in het stap voor stap proces van het melden horen. Elk van deze schermen laat een andere stap in het proces zien. De *FormCategoryActivity* geeft de gebruiker de gelegenheid om een categorie te kiezen. De *FormCommentActivity* geeft de gebruiker de gelegenheid om opmerkingen te typen bij de foto. De *FormSendActivity* informeert de gebruiker over de status van het verzenden van de melding. De data van het stap voor stap proces wordt teruggestuurd naar de *ReportActivity* welke het verder afhandelt.

amgc.android::RewardActivity
De RewardActivity dient voor het weergeven van de rewards die zijn verdiend. Dit is mogelijk door de implementatie van de *arca.ar::ARDisplayActivity* klasse. Deze biedt de mogelijkheid voor augmented reality (zie amgc.android::ReportActivity). Ook kan de gebruiker zien hoeveel rewards er verkregen zijn.

amgc.android::GpsLocater
Het ARCA framework biedt nog geen on demand service voor GPS locaties. Deze klasse biedt die mogelijkheid wel. Dit gebeurt door de GPS sensor te activeren en zodra er een locatie beschikbaar is in Android wordt de GPS sensor weer gedeactiveerd en de locatie wordt opgeslagen. Dit wordt alleen gebruikt op het moment dat er een foto wordt genomen, zodat de locatie van de melding kan worden vastgelegd.

amgc.android::ReportManager
De *ReportManager* zorgt voor het beheren van de meldingen. Het zorgt ervoor dat de melding juist opgebouwd en verstuurd wordt. Door de *XMPPSupport* klasse in het ARCA framework kan de *ReportManager* de melding versturen.

amgc.android::RewardManager
De *RewardManager* zorgt voor het beheren van de rewards. Door met de binnengekomen data een reward object te maken is het mogelijk de rewards goed te beheren. Zo kunnen de rewards weergegeven worden in de *RewardActivity*.

Report en Reward klassen
Deze klassen stellen de meldingen (report) en rewards voor. De klassen hebben attributen die de eigenschappen van de objecten voorstellen.

6.2.3.2 Server applicatie

In diagram 6.4 is te zien hoe de server applicatie is opgebouwd. Het bestaat uit een vijftal klassen. De klassen *ReportManager, Report, RewardManager* en *Reward* komen overeen met die van de te ontwikkelen Android applicatie.
De klasse *ConnectionManager* zorgt voor het ontvangen van de melding en het versturen van de feedback / reward. Deze klasse beheert de XMPP verbinding, als er data binnenkomt dat overeenkomt met een melding (report tag, foto, locatie, categorie en evetueel een opmerking) wordt de *ReportManager* geactiveerd en wordt de data doorgegeven. Deze zorgt voor het juist opslaan van de data in de database. Als de *ConnectionManager* opdracht krijgt van de *RewardManager* om een reward / feedback te versturen, wordt het bericht opgebouwd en verstuurd.

Als er meldingen worden opgevraagd krijgt de *ConnectionManager* hiervoor een aanvraag binnen. Vervolgens gaat de *ReportManager* alle gevraagde rapporten opvragen uit de database en opbouwen. Hierna krijgt de *ConnectionManager* de opdracht om deze te versturen naar de bron die aanvraag indiende.

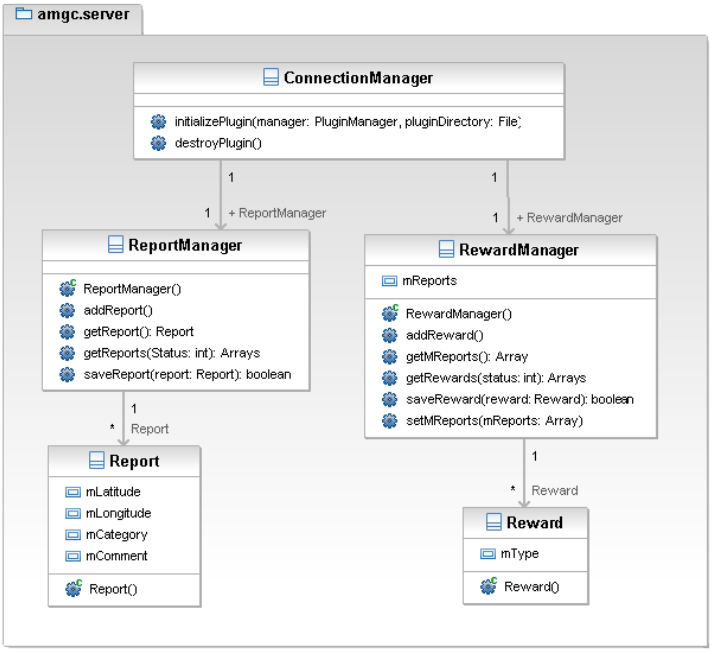


Diagram 6.4: Server klassendiagram

6.2.4 Database

De server applicatie gebruikt de bij Openfire ingebouwde HSQLDB (**H**yper**SQL** **D**ata**B**ase) database. Het systeem bestaat uit vier tabellen. De *reports* tabel slaat alle meldingen op met de daarbij benodigde informatie. De *reports_rewards* tabel houdt per melding bij welke reward bij de melding hoort en of deze verstuurd is. De verschillende soorten rewards worden opgeslagen in de tabel *rewards*. De laatste tabel heet *states* en bevat alle soorten statussen die aan de melding kan worden gekoppeld.
In diagram 4.1 is te zie hoe de database is opgebouwd. Aan elke melding (report) kan maar één status worden gekoppeld en ook maar een enkele reward.

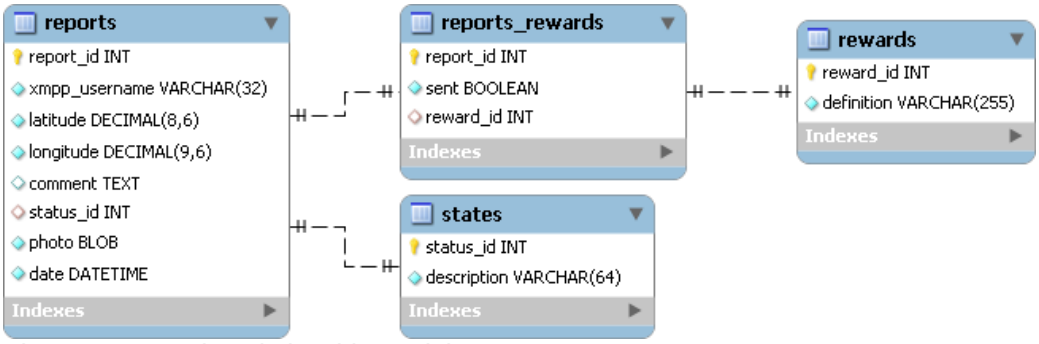


Diagram 6.5: Entity relationship model

reports

Data item	Beschrijving
report_id	Uniek nummer van de melding
xmpp_username	Username waarmee is geregistreerd op de xmpp server
latitude	Breedtegraad
longitude	Lengtegraad
comment	Omschrijving situatie
status_id	Uniek nummer van de status
photo	De gemaakte foto (1024 x 768 pixels)
date	Datum dat de melding is ontvangen

Tabel 6.11: Data item van 'reports'

reports_rewards

Data item	Beschrijving
report_id	Uniek nummer van de melding
sent	Geeft aan of de reward succesvol is verstuurd naar de telefoon
reward_id	Uniek nummer van de soort reward

Tabel 6.12: Data items van 'reports_rewards'

rewards

Data item	Beschrijving
reward_id	Uniek nummer van de reward
definition	Omschrijving van de reward
reward_id	Uniek nummer van de reward
definition	Omschrijving van de reward

Tabel 6.13: Data items van 'rewards'

states

Data item	Beschrijving
status_id	Uniek nummer van de status
description	Tekst van de status; bijvoorbeeld: ‘Wordt gerepareerd’

Tabel 6.14: Data items van 'states'

6.3 Sequence diagrammen

6.3.1 Probleem melden

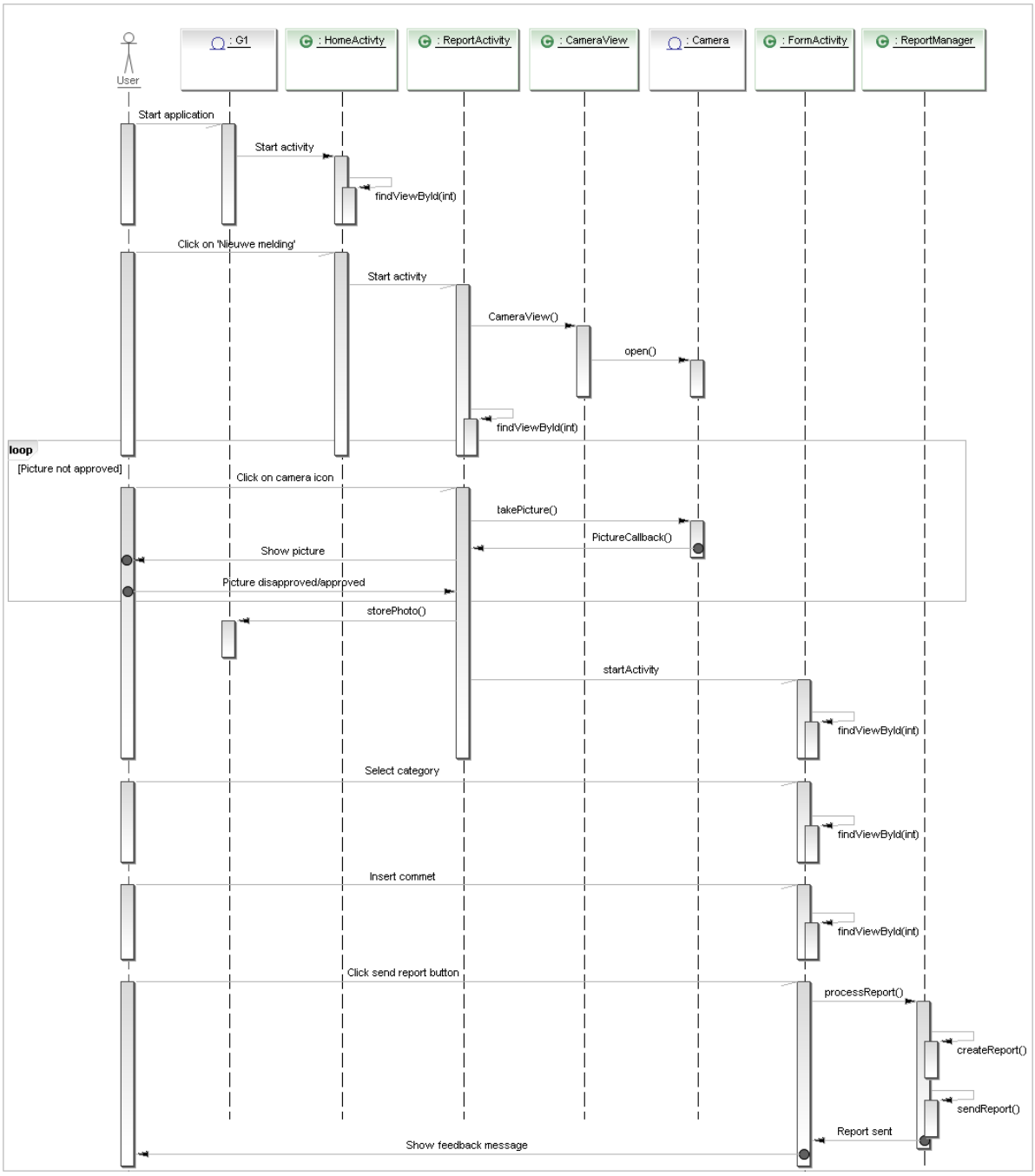


Diagram 6.6: Melden van een probleem

In diagram 6.6 is te zien hoe de interactie verloopt tussen de verschillende klassen om een melding te genereren. De gebruiker start de applicatie waarop het eerste scherm (*HomeActivity)* wordt getoond. De bijbehorende view wordt opgevraagd door de *findViewById()* methode. Als de gebruiker vervolgens het proces voor een nieuwe melding start wordt het volgende scherm (*ReportActivity)* getoond. In dit scherm wordt het camerabeeld getoond, hier zorgt de *CameraView* voor. De view die wordt gevonden door *findViewById()* zorgt voor een balk onderaan in het scherm met een knop waarmee een foto gemaakt kan worden. Als op de knop wordt gedrukt, wordt er een foto gemaakt. Hierna volgt een keuze:

- De foto wordt goedgekeurd en er wordt een scherm getoond (*FormCategoryActivity)* waarin de gebruiker een categorie kan selecteren.
- De foto wordt afgekeurd en de gebruiker krijgt het vorige scherm te zien, waar een nieuwe foto gemaakt kan worden.

Als de categorie is geselecteerd komt er een nieuw scherm (*FormCommentActivity*) waarin de gebruiker een opmerking kan invoeren (niet verplicht). Als de gebruiker de melding verstuurd wordt de *Report* opgemaakt en verstuurd. Tijdens het versturen zorgt de *FormSendActivity* voor een scherm waarop de voortgang van het verzenden te zien is. Daarna krijgt de gebruiker een bedankt berichtje te zien.

6.3.2 Ontvangen melding

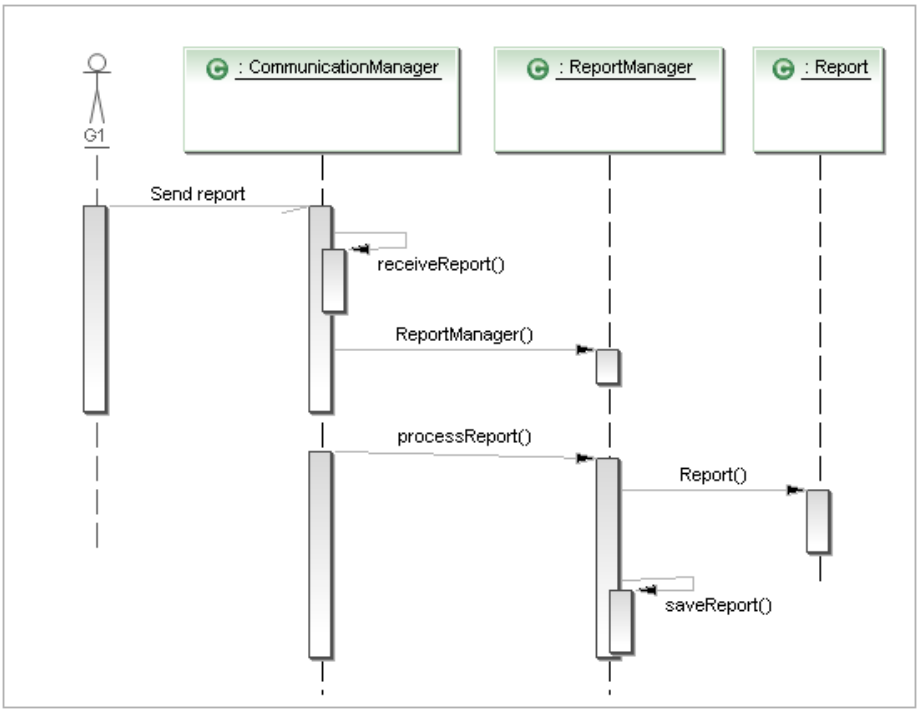


Diagram 6.7: Ontvangen van een melding

In diagram 6.7 is te zien hoe de melding die wordt verstuurd vanaf de G1 wordt ontvangen door de server bij de *ConnectionManager*. Deze delegeert de afhandeling naar de *ReportManager*. Deze klasse zorgt ervoor dat de melding in de database opgeslagen wordt.

6.3.3 Versturen reward

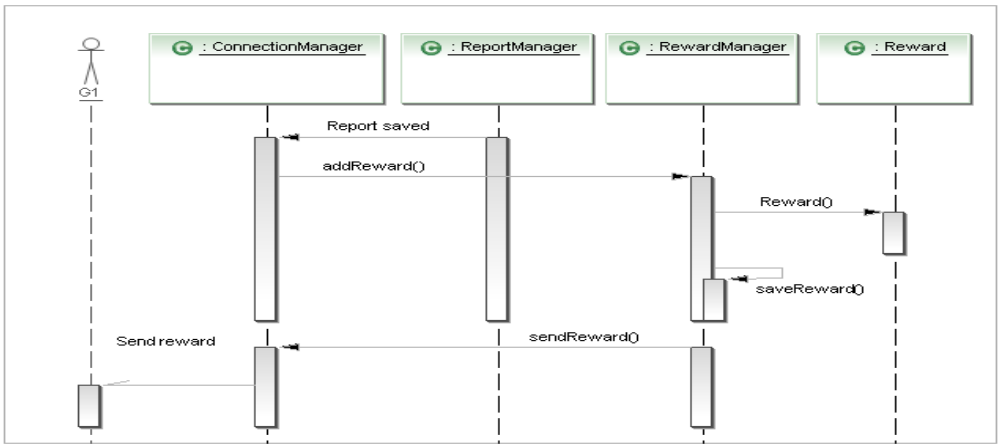


Diagram 6.8: Versturen feedback / reward

In diagram 6.8 is te zien hoe het proces voor het versturen van een reward verloopt. Als de melding is opgeslagen dient er een reward te worden verstuurd. Hiervoor wordt de *RewardManager* geactiveerd. Deze zal een reward aanmaken en opslaan in de database. Vervolgens geeft hij de opdracht aan de *ConnectionManager* om de reward te versturen naar de G1.

6.3.4 Ontvangen reward

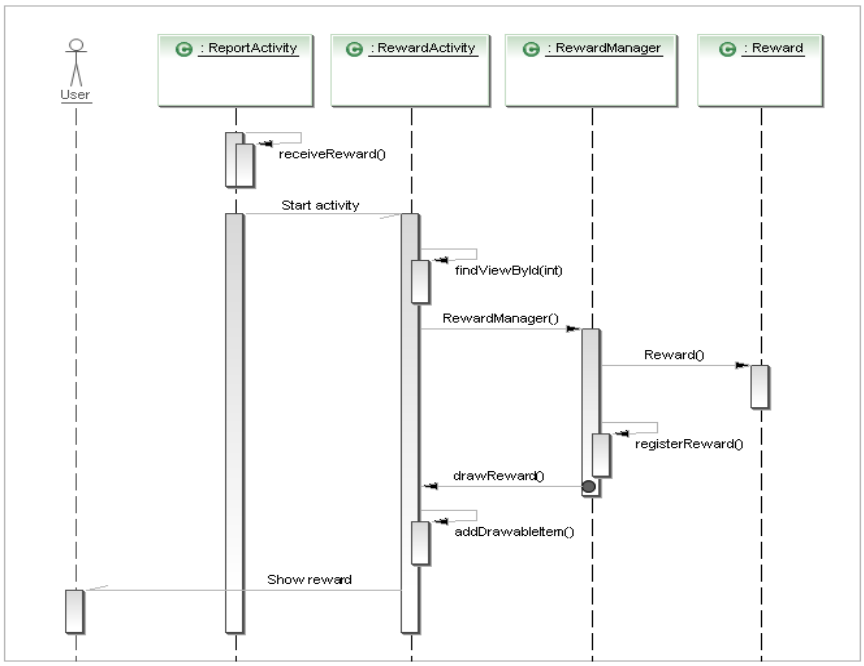


Diagram 6.9: Ontvangen reward

In diagram 6.9 is het proces te zien voor het ontvangen van een reward. De *ReportActivity* ontvangt de reward en geeft dit door aan de *RewardActivity* welke ervoor zorgt dat de reward in het beheer van de *RewardManager* komt. Deze zal de reward in de database opslaan. Als dit is gebeurd, tekent de *RewardActivity* deze op het scherm.

7 Resultaten ontwikkeling

7.1 Functionaliteiten Android applicatie

In tabel 6.1 is een overzicht te zien van alle functionaliteiten voor het systeem. Deze zijn ingedeeld volgens de MoSCoW methode. Het gemaakte prototype bevat niet alle functionaliteiten die in tabel 6.1 genoemd worden. De functionaliteiten met de prioriteiten *could have* en *would have* zijn in verband met de beperkte tijd eruit gelaten. Volgens de definities van deze prioriteiten zijn dit functionaliteiten die makkelijk weggelaten kunnen worden binnen de huidige ontwikkeling, maar wel waardevol zijn voor het systeem.

Must have functionaliteiten zijn essentieel voor het systeem en deze zijn dan ook allemaal geïmplementeerd. Hierdoor is het ontworpen prototype zeer geschikt om de werking en het doel van het systeem aan te tonen.
Door tijdgebrek moesten de *should have* functionaliteiten ook geschrapt worden uit de ontwikkeling. Wel wordt hierbij de opmerking geplaatst dat de combinatie van de volgende twee functionaliteiten ‘ophalen problemen in de buurt (Android applicatie)’ en ‘Status probleem weergeven’ een hoge waarde heeft voor het systeem. Door deze functionaliteiten toe te passen kunnen gebruikers zien waar welke problemen al zijn gemeld. Dit voorkomt dat de gebruiker een melding doet van een probleem dat al gemeld is.

7.2 Communicatie

In paragraaf 5.5.4 is er tot de conclusie gekomen dat XMPP als protocol geprefereerd wordt boven HTTP. Echter, tijdens het ontwikkelproces van de server applicatie is gebleken dat het versturen en ontvangen van bestanden meer tijd vereist dan de planning toelaat. Ondanks de kennis van XMPP binnen Logica blijkt het versturen van bestanden over dit protocol nog lastig. Ook blijkt het versturen van bestanden met behulp van de Smack library (open source XMPP Java client library) nog niet geheel stabiel. Om toch de gemaakte foto’s en data te kunnen versturen is er overgestapt naar HTTP. Hierop draait een PHP (**P**HP: **H**ypertext **P**reprocessor) applicatie die de taken van de server kant uitvoert. Door het tijdig overstappen is er voor de deadline toch een server applicatie ontwikkeld die aan de *must have* prioriteiten voldoet.

Er is enige ondersteuning voor XMPP in de Android applicatie en ook een deel van de XMPP server applicatie ontwikkeld. De Android applicatie kan een melding versturen (met uitzondering van de foto). De server deel wat ontwikkeld is kan een melding ontvangen en een reward terugsturen. De Android applicatie ontvangt deze response echter niet. In de volgende paragrafen wordt daarom ingegaan op het gemaakt deel.

Voor de server kant wordt gebruik gemaakt van Openfire (zie paragraaf 6.2.1). Android bevat geen standaard implementatie van XMPP. Om de ondersteuning voor XMPP toe te voegen is er in het ARCA framework (zie paragraaf 6.2.2) de Smack library gebruikt. De library wordt gebruikt om XMPP te kunnen integreren in applicaties. Echter is de Smack library niet volledig compatibel met Android. Hierop is een variant gemaakt genaamd ‘aSmack’. Op deze library zijn patches toegepast om het compatibel te maken met Android.

Tussen de server (Openfire) en de Android applicatie worden verschillende XML berichten (stanzas) verstuurd. Hiervoor worden IQ[23] (**I**nfo/**Q**uery) stanzas gebruikt. Dit is een request-response patroon. Waarbij de request verschillende types kan zijn. Dit zijn:

- **get:** stanza voor het aanvragen van informatie
- **set:** stanza voor het geven van data
- **result:** stanza response voor een goed verlopen *get* of *set*
- **error:** stanza response voor een verkeerd verlopen *get* of *set*

Daarnaast bevat elke IQ stanza een *from* en *to* attribuut waarin respectievelijk de verzender en ontvanger in staan. Het *id* attribuut is een automatisch gegenereerde waarde van letters en cijfers.

Om inzicht te krijgen hoe het systeem werkt zullen de zelf gedefinieerde stanzas beschreven worden.

In codeblok 7.1 is de stanza te zien die wordt verstuurd van de Android applicatie naar de XMPP server. Deze stanza wordt gebruikt om een melding te registreren in de database van de server. In de stanza wordt de longitude (lengtegraad), latitude (breedtegraad), categorie en de opmerking meegestuurd. De response van de server is een standaard stanza van het type *result* (zie codeblok 7.2) op succes of *error* als er iets is fout gegaan.

```
<iq type='set' from='sandervdwal' to='127.0.0.1' id='purpledisco61c276f7'>
  <query xmlns='amgc:iq:report'>
    <report>
      <longitude>4.9213</longitude>
      <latitude>54.5</latitude>
      <category>bushok</category>
      <comment>Ingegooide ruit</comment>
    </report>
  </query>
</iq>
```

Codeblok 7.1: Report stanza

In codeblok 7.2 is de stanza te zien die wordt verstuurd door de XMPP server naar de Android applicatie als de melding met succes is ontvangen en geregistreerd. De stanza bevat het type reward die is toegekend aan de melding.

```
<iq type='result' from='127.0.0.1' to='sandervdwal@127.0.0.1' id='purpledisco61c276f7'>
  <query xmlns='amgc:iq:report'>
    <reward>
      <type>buddy</type>
    </reward>
  </query>
</iq>
```

Codeblok 7.2: Report result stanza

8 Conclusie en aanbevelingen

8.1 Conclusie

De vraag die centraal stond in dit afstudeerproject luidde als volgt:

“Op welke wijze kan de eindgebruiker door middel van een Android applicatie die gebruik maakt van augmented reality, verleidt worden om problemen of schade aan eigendommen te melden bij de onderhoudsdienst?”

Tijdens dit afstudeerproject zijn er, om de hoofdvraag te beantwoorden, vier deelvragen geformuleerd en beantwoord. Aan de hand van deze antwoorden kon een prototype van de Android applicatie gemaakt worden om te demonstreren dat de theoretische beredenering ook in de praktijk is uit te voeren.

Deze vier antwoorden tezamen genomen bieden een antwoord op de centrale vraag:

Door een Android applicatie te maken welke snel is, intuïtief werkt en is uitgerust met dynamische content maakt het interessant om de applicatie te blijven gebruiken. Door het toevoegen van rewards is het mogelijk om punten te verzamelen en dit maakt het, naast het gevoel dat men iets kan doen, verleidelijk voor de eindgebruiker. Daarnaast is het weergeven van een bericht waarin de gebruiker wordt bedankt, de interactie naar de gebruiker toe die dit waarderen. Het toepassen van augmented reality gebeurt door middel van de rewards te tekenen over de reële beelden. GPS wordt gebruikt om de positie van het probleem te bepalen, hierdoor wordt de eindgebruiker niet meer lastig gevallen over de plaats waar het probleem zich bevindt.

8.2 Aanbevelingen

Dit afstudeerproject was vooral gericht op het client gedeelte van het hele systeem. Dit betekent dat het overgrote deel van de tijd is besteed aan het ontwikkelen van de Android applicatie. Hierbij zijn niet alle aspecten voor de communicatie aan bod gekomen en is de server kant niet veel meer dan een backend voor het ontvangen van de meldingen en het terugsturen van een reward om te demonstreren dat er met een server gecommuniceerd kan worden. De volgende aanbevelingen kunnen gedaan worden:

Ontwikkelen server applicatie
Om de onderhoudsdiensten optimaal te kunnen laten profiteren van de mogelijkheid om problemen te melden, dient er een degelijke server applicatie te worden ontwikkeld. Met de server applicatie is het dan mogelijk om meldingen te beheren een aan te passen naar een andere status. Hiervoor is een nieuw onderzoek nodig om te bepalen wat de nodige functionaliteiten zijn en technische specificaties.

Beloning voor de rewards
Om het geheel nog aantrekkelijker te maken en breder publiek aan te spreken is het mogelijk om de verzamelde rewards in te ruilen voor echte goederen. Zo zou men bijvoorbeeld verschillende rewards kunnen krijgen. Als eenzelfde eindgebruiker tien keer een volle prullenbak voor een café meldt, dat de eindgebruiker dan een gratis kopje koffie kan krijgen in ruil voor de rewards. Hoe dit mogelijk gemaakt kan worden en eruit moet gaan zien heeft een nieuw onderzoek nodig om het beter uit te kunnen werken.

Doorontwikkelen Android applicatie
Naast de server applicatie zijn er ook nog veel mogelijkheden met de Android applicatie. In tabel 6.1 (MoSCoW prioriteiten) staan een aantal functionaliteiten die nog niet in de huidige Android applicatie zitten. Voor een nieuwe versie is het aan te bevelen om ook deze functionaliteiten alsnog erin te verwerken.

9 Evaluatie

Aan het begin van de afstudeeropdracht werd een plan van aanpak gemaakt. Ondanks alle plannen van aanpak die ik op school heb gemaakt was het toch lastiger dan ik had gedacht. Om het zo op papier te zetten dat het voor iedereen duidelijk is waar het over gaat en wat uiteindelijk het doel is, was lastig. Hier is dan ook meer tijd ingegaan dan ik er voor gepland had in mijn afstudeervoorstel.

Nadat de grote lijnen uit het afstudeervoorstel gedetailleerder waren uitgewerkt in het plan van aanpak begon ik aan het onderzoek. Ik had nog nooit eerder een dergelijk onderzoek gedaan en gedocumenteerd. Gelukkig waren er wel voorbeelden en een aantal richtlijnen om me hierbij te helpen. Achteraf heb ik teveel tijd besteed aan het onderzoek. Tijdens het onderzoek had ik wel een aantal kleine demo's gemaakt, waardoor ik al wat praktijk ervaring opdeed met Android. Alleen het doel van de demo's, kunnen werken met de sensoren, was naderhand niet meer echt nodig. Dit komt door het ARCA framework dat ik heb gebruikt. Deze handelt de sensoren af en ik had er bijna niets mee te maken, maar ik had hierdoor wel alvast met Android gewerkt voordat ik begon aan het ontwikkelen van de Android applicatie. De opgedane ervaring heeft wel geholpen in de realisatiefase, wat een lichte tijdwinst gaf.

Ondanks de vier componenten waaruit Android bestaat, was het maken van een functioneel- en technisch ontwerp niet veel anders. De tijd die was ingepland voor deze twee documenten kwam redelijk overeen. Alleen door de eerdere opgelopen vertraging, was er niet genoeg tijd om de final versie van de documenten op te leveren voordat de ontwikkelfase begon. De gemaakt diagrammen waren grotendeels wel goed genoeg om aan de hand hiervan een applicatie te maken. De beschrijvingen lieten echter te wensen over, dus was het voor externen niet altijd erg duidelijk wat bepaalde acties moesten doen.

Het ontwikkel traject verliep gestaag. Van Android had ik al enige kennis toen ik hier aan begon, maar met de server kant had ik geen ervaring. Gelukkig was het opzetten van de XMPP communicatie niet zo lastig als ik eerst gedacht, bij Android zat XMPP functionaliteit ingebouwd in het ARCA framework. Aan de server kant was het mogelijk om voor Openfire (XMPP server) een plugin te ontwikkelen met de nodige functionaliteit om de Android applicatie te testen. Deze plugin moest in Java worden geprogrammeerd. Ik had hierdoor maar te maken met één soort code syntax (Android gebruikt Java syntax). Voor het ontwikkelen van de Openfire plugin was genoeg documentatie beschikbaar om de functionaliteiten toe te passen die ik nodig had. Helaas is de ondersteuning voor het versturen van bestanden minimaal. Mede daardoor was de tijd te kort om dit toe te kunnen passen. Daarom is er overgestapt naar het eerste alternatief; HTTP. Door middel van PHP kon er redelijk snel een vervangende applicatie gemaakt worden. Ook bij de Android applicatie was de aanpassing goed te doen.

Tijdens de hele periode werd ik begeleid door Herbert Leenstra. Onder andere via Herbert ben ik in contact gekomen met een aantal collega's. Één van hen is Tako Sondorp. Hij heeft binnen Logica ervaring met Android en ik kon bij hem dus terecht met Android specifieke vragen. Daarnaast had ik een inhoudelijk begeleider. Jos Duker was Java engineer in Nieuwegein tijdens mijn afstudeerperiode. Wij hadden wekelijks een vergadering om te bekijken hoe het project verliep. Bij hem kon ik terecht met vragen over het ontwerp en ontwikkeling van de Android- en server applicatie.

Over het algemeen ben ik zeer tevreden met mijn periode bij Logica. Er is veel kennis en hulp aanwezig en de ontwikkelde proof of concept werkt naar behoren.

10 Appendix

10.1 Verklarende woordenlijst

Term	Toelichting
Activity	Een applicatiecomponent van Android. Een activity presenteert een zichtbare user interface waarmee de gebruiker acties kan ondernemen.
ADT	**A**ndroid **D**evelopment **T**ools; plugin voor Eclipse welke extensies toevoegt voor het ontwikkelen van Android applicaties.
AMGC	**A**ugmented **M**aintenance **G**arbage **C**ontrol
Android	Open source platform voor mobiele telefoons.
ARCA	**A**ugmented **R**eality **C**ontext **A**wareness
aSmack	Smack library met toegepaste patches, zodat de library compatible is met Android.
Augmented Reality	Computergemaakte beeld toevoegen aan reële beelden.
Breedtegraad	Verticale geografische locatie in graden.
Broadcast receiver	Een applicatiecomponent van Android dat broadcast announcements ontvangt en hierop reageert.
DSDM	**D**ynamic **S**ystems **D**evelopment **M**ethod is een methodische aanpak voor het ontwikkelen van software.
Eclipse	Open source framework voor software ontwikkelomgevingen.
Eindgebruiker	Gebruiker van de mobiele Android applicatie.
EUT	**E**nergy **U**tilities & **T**elecom; divisie binnen Logica.
G1	Google Phone 1; eerste mobiele telefoon die op Android draait.
GPS	**G**lobal **P**ositioning **S**ystem; plaatsbepalingsysteem welke gebruik gemaakt van satellieten.
GUI	**G**raphical **U**ser **I**nterface
HSQLDB	**H**yper **S**tructured **Q**uery **L**anguage **D**ata**B**ase
HTML	**H**yper**T**ext **M**arkup **L**anguage
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol; protocol voor client-server communicatie.
IDE	Integrated Development Environment; ontwikkelomgeving welke programmeurs ondersteund bij het ontwikkelen van applicaties.
IQ	**I**nfo/**Q**uery; type stanza om een request te versturen en een response te ontvangen.
Iteratief	Herhalend doorlopen van een proces.
JCDecaux	Bedrijf dat gemeenten stadsmeubilair aanbiedt en zorgt voor plaatsing, onderhoud en reparatie hieraan. In ruil daarvoor krijgen ze het recht om reclameruimte te exploiteren.
Lengtegraad	Horizontale geografische locatie in graden.

MHz	**Mega**-**Hertz**; eenheid van frequentie
MoSCoW	MoSCoW is een acroniem voor de manier waarop prioriteiten aan de vereisten worden toegekend. De letters staan voor **M**ust have, **S**hould have, **C**ould have en **W**ould like to have.
Namespace	Een scope om logisch te groeperen.
Onderhoudsdienst	Bedrijf of onderdeel van een bedrijf dat onderhoud pleegt aan eigendommen of eigendommen van het bedrijf dat hen inhuurt.
PHP	**P**HP: **H**ypertext **P**reprocessor; scripttaal om dynamische websites te ontwikkelen.
Service	Een applicatiecomponent van Android dat voor een onbepaalde periode op de achtergrond van een applicatie draait.
Smack	Open source XMPP client library geprogrammeerd in Java.
SQL	**S**tructured **Q**uery **L**anguage
SQLite	Embedded database engine welke gebruik maakt van SQL.
SSID	**S**ervice **S**et **Id**entifier; naam voor een WLAN.
Stanza	XML bericht dat wordt verstuurd tussen de client en een XMPP server.
Timeboxing	Tijd tussen de begin- en einddatum van een periode waarvan aan het eind ‘iets’ wordt afgeleverd. Dit ‘iets’ kan bijvoorbeeld zijn een analysemodel, deel van een frontend of functionaliteit.
UI	**U**ser **I**nterface
WiFi-netwerk	Draadloze netwerk dat voldoet aan de IEEE 802.11 standaarden.
WLAN	**W**ireless **L**ocal **A**rea **N**etwork
WT	**W**orking **T**omorrow; een programma dat opgestart is binnen Logica om plaats te bieden aan studenten die op verschillende locaties binnen het programma aan hun afstudeeropdracht werken.
XML	E**x**tensible **M**arkup **L**anguage; standaard voor het gestructureerd versturen en opslaan van data.
XMPP	E**x**tensible **M**essaging and **P**resence **P**rotocol; open XML techniek voor real-time communicatie.

Tabel 10.10 1: Verklarende woordenlijst

10.2 Literatuurlijst

[1] Rubin, A. (2007). *Where's my Gphone?*. Geraadpleegd op 22-02-2010 via http://googleblog.blogspot.com/2007/11/wheres-my-gphone.html

[2] Toet, D. (2009). *Android-telefoon ligt in Nederlandse schappen*. Geraadpleegd op 02-03-2010 via http://www.computable.nl/artikel/ict_topics/telecom/2852260/1276977/androidtelefoon-ligt-in-nederlandse-schappen.html

[3] http://layar.com/ - Geraadpleegd op 22-02-2010

[4] http://www.jcdecaux.nl/stadsmeubilair_onderhoud.html - Geraadpleegd op 03-03-2010

[5] http://www.jcdecaux.nl/ - Geraadpleegd op 03-03-2010

[6] Stapleton, J. (2005[2]). *DSDM – De methode in de praktijk*. z.pl.: Pearson Education Benelux.

[7] http://developer.android.com/guide/index.html - Geraadpleegd op 23-02-2010

[8] http://www.eclipse.org/ - Geraadpleegd op 23-02-2010

[9] Yinger, C.H. (2002). *Operation and Application of the Global Positioning System*. Geraadpleegd op 11-03-2010 via http://www.aero.org/publications/crosslink/summer2002/02.html

[10] (2003). *TTFF Comparisons*. Geraadpleegd op 10-03-2010 via http://www.pocketgpsworld.com/ttffcomparisons.php

[11] Biuk-Aghai, R.P. (2007). *GSM-Based Provider-Independent Positioning Method*. Geraadpleegd op 15-03-2010 via http://www.sftw.umac.mo/~robertb/publications/LocationAsia2007/LOC_A07_29.pdf

[12] Khare, D. (2010), *A secret to success for your mobile* application. Geraadpleegd op 23-03-2010 via http://www.nextwala.com/nextwala/2010/01/the-secret-to-success-for-your-mobile-application.html

[13] Capobianco, F. (2009), *Why mobile apps fail*. Geraadpleegd op 23-03-2010 via http://www.funambol.com/blog/capo/2009/06/why-mobile-apps-fail.html

[14] Wizdo, L. (2007). *End User Experience – The Performance Management Hat Trick*. Geraadpleegd op 23-03-2010 via http://knoa.com/main/docs/CIO Post_The Performance Management Hat Trick.pdf

[15] Wilska, E. *Non-Fatal Errors: Creating Usable, Efective Error Messages*. Geraadpleegd op 24-03-2010 via http://www.writersua.com/articles/message/index.html

[16] (2009). *5 Design Tips For A Winning Mobile Application*. Geraadpleegd op 26-03-2010 via http://fivemobile.com/tips-tricks/top-10-tips/

[17] Hamilton, R. (2008). *Fast is better than slow*. Geraadpleegd op 26-03-2010 via http://googlemobile.blogspot.com/2008/03/fast-is-better-than-slow.html

[18] http://xmpp.org/ - Geraadpleegd op 29-03-2010

[19] Wagener, J., Spjuth, O., Willighagen, E.L., Wikberg, J.E.S. (2009). *XMPP for cloud computing in bioinformatics supporting discovery and invocation of asynchronous web services*. Geraadpleegd op 29-03-2010 via http://www.biomedcentral.com/content/pdf/1471-2105-10-279.pdf

[20] (2009). *HTTP Headers for Dummies*. Geraadpleegd op 29-03-2010 via http://www.neurosoftware.ro/programming-blog/facebook-web-design/web-resources/http-headers-for-dummies/

[21] http://www.igniterealtime.org/projects/openfire/index.jsp - Geraadpleegd op 29-03-2010

[22] Uijtdewilligen, F. (2010). *A framework for context-awareness applications using augmented reality: A train station navigation proof-of-concept on Google Android*

[23] http://xmpp.org/rfcs/rfc3920.html - Geraadpleegd op 30-03-2010

10.3 Lijsten

Figurenlijst

Tabellenlijst

Diagrammenlijst

Codeblokkenlijst

[1] http://googleblog.blogspot.com/2007/11/wheres-my-gphone.html
[2] http://www.computable.nl/artikel/ict_topics/telecom/2852260/1276977/androidtelefoon-ligt-in-nederlandse-schappen.html
[3] http://layar.com/
[4] http://www.jcdecaux.nl/stadsmeubilair_onderhoud.html
[5] http://www.jcdecaux.nl/
[6] http://www.dsdm.org/
[7] http://developer.android.com/guide/index.html - geraadpleegd op 23-02-2010
[8] http://www.eclipse.org/ - geraadpleegd op 23-02-2010
[9] http://www.aero.org/publications/crosslink/summer2002/02.html - geraadpleegd op 11-03-2010
[10] http://www.pocketgpsworld.com/ttffcomparisons.php - geraadpleegd op 10-03-2010
[11] http://www.sftw.umac.mo/~robertb/publications/LocationAsia2007/LOC_A07_29.pdf - geraadpleegd op 15-03-2010
[12] http://www.nextwala.com/nextwala/2010/01/the-secret-to-success-for-your-mobile-application.html
[13] http://www.funambol.com/blog/capo/2009/06/why-mobile-apps-fail.html
[14] http://knoa.com/main/docs/CIO Post_The Performance Management Hat Trick.pdf - geraadpleegd op 23-03-2010
[15] http://www.writersua.com/articles/message/index.html
[16] http://fivemobile.com/tips-tricks/top-10-tips/
[17] http://googlemobile.blogspot.com/2008/03/fast-is-better-than-slow.html
[18] http://xmpp.org/ - geraadpleegd op 29-03-2010
[19] Bron: http://www.biomedcentral.com/content/figures/1471-2105-10-279-1-l.jpg
[20] Bron: http://www.neurosoftware.ro/programming-blog/facebook-web-design/web-resources/http-headers-for-dummies/
[21] Openfire: http://www.igniterealtime.org/projects/openfire/index.jsp
[22] ARCA; *Uijtdewilligen, F. (2010). "A framework for context-awareness applications using augmented reality: A train station navigation proof-of-concept on Google Android"*
[23] http://xmpp.org/rfcs/rfc3920.html - geraadpleegd op 18-05-2010