Wireless & Cable (INTEC)

Academiejaar 2010-2011

Departement Toegepaste Ingenieurswetenschappen
Valentin Vaerwyckweg 1, 9000 Gent

Ontwikkeling van een mobiel conversatiehulpprogramma op Google Android om de communicatie te verbeteren van dove en slechthorende kinderen

Masterproef voorgedragen tot het behalen van het diploma van
MASTER IN DE INDUSTRIËLE WETENSCHAPPEN: INFORMATICA

Laurens VAN ACKER

Promotoren:	*Intern:*	*dr. Veerle ONGENAE*
	Extern:	*prof. dr. ir. Wout JOSEPH*
		prof. dr. ir. Luc MARTENS
	Begeleiders:	*ir. Adrián JUAN*
		ir. Toon DE PESSEMIER
		ing. Kris VANHECKE

Inhoudsopgave

Inhoudsopgave **v**

Voorwoord **vi**

Engelstalige abstract **viii**

Nederlandstalig abstract **ix**

Inleiding **1**

1 Wireless & Cable (WiCa) **3**

2 Situering **5**
2.1 Probleemstelling 5
2.2 Doelstelling 6
2.3 Praktijk 6
2.4 Gedetailleerde omschrijving van de opdracht 6
2.5 Problemen die opgelost moesten worden 8
2.6 Doelgroep 9
2.7 Beperkingen 10
2.8 Een psychologisch perspectief op doofheid 10
2.8.1 Gesprekken op het werkveld 12
2.8.1.1 Koninklijk Instituut voor Doven en Spraakgestoorden (KIDS) 13
2.8.1.2 Het GR@SP project 13
2.8.1.3 MPI Sint-Lievenspoort 14
2.8.1.4 BuSO Sint-Gregorius 14
2.8.2 Problemen in het dagelijkse leven 14
2.8.3 Onderlinge communicatie 15
2.8.4 De context van een tekst 15
2.8.5 Invloed van gebarentaal op de geschreven taal 15
2.8.6 Workflow van de voorgestelde applicatie 16

2.8.7 Technische hulpmiddelen 16
2.8.8 Liplezen 17
2.8.9 Gebarentalen 17
2.8.10 Gebarensystemen 17
2.8.11 Woordenschat van dove personen 18
2.8.12 Bedreigde verwerving van gesproken taal en het taalbegrip 18
2.8.13 Leren lezen 19
2.9 Bestaande projecten 19
2.9.1 I-CITY 19
2.9.2 Draadloze videoconversaties 19
2.9.3 Oralys op pocket PC 20
2.9.4 Sprint en Kuzweil 3000 21

3 Gebruikte technologiën 22
3.1 Google Android 22
3.1.1 Android componenten 22
3.1.1.1 Activities 22
3.1.1.2 Intents 25
3.1.1.3 Content providers 27
3.1.1.4 Services 28
3.1.2 Veiligheid en rechten 28
3.1.2.1 Eigen toegangsrechten declareren 29
3.2 Het gebruik van de savedInstanceState bundel 30
3.3 Handige Android-bibliotheken die gebruikt zijn in de applicatie 31
3.3.0.2 De android.net.Uri.Builder klasse 31
3.3.1 De java.util.zip.ZipFile klasse 32
3.4 Optical character recognition (OCR) software voor Android . 32
3.4.1 De theorie 33
3.4.1.1 Training 34
3.4.1.2 Taaltechnologie 34
3.4.2 ABBYY 34
3.4.3 Google Goggles 34
3.4.3.1 Verbinden met de Google Goggles Service . . 35
3.4.4 IQ Engines 37
3.4.5 Mezzofanti 37
3.4.6 Gebruikte technologie 37
3.4.7 De OCR API van WiseTrend 38
3.4.7.1 Het gebruik van de API 39
3.5 Spraakherkenning voor Android 42
3.6 Bronnen 43

3.6.1 Google Image Search API . . . 43
3.6.2 De API Flickr . . . 44
3.6.3 De data API van YouTube . . . 46
3.6.4 Vimeo Advanced API . . . 47
3.7 Ophalen en bewaren van afbeeldingen . . . 49
3.8 Onderzoek woordenboek API's . . . 49
3.8.1 Google Woordenboek . . . 50
3.8.2 WordNet . . . 50
3.8.3 Van Dale . . . 51
3.8.4 LookWAYup . . . 52
3.8.5 OpenTaal . . . 52
3.8.5.1 OpenThesaurus . . . 55
3.8.6 Wiktionary . . . 55
3.8.7 Part-of-speech . . . 56
3.8.7.1 Frog . . . 56
3.8.7.2 Frog installatie . . . 57
3.9 JSON . . . 59
3.10 OAuth . . . 59
3.10.1 Inleiding . . . 60
3.10.2 De verschillende soorten sleutels . . . 60
3.10.3 Het 3-legged OAuth proces . . . 61
3.10.4 Het 2-legged OAuth proces . . . 62

4 Implementatie 63
4.1 Structuur . . . 63
4.1.1 Alternatieve resource-bestanden . . . 66
4.2 Eigen klassen . . . 67
4.2.1 De package 'controller' . . . 67
4.2.2 De package 'model' . . . 68
4.2.3 Het package 'filter' . . . 70
4.2.4 De package 'hints' . . . 72
4.2.5 De package 'sources' . . . 72
4.3 Externe klassen . . . 72
4.3.1 De OAuth-signpost bibliotheek . . . 72
4.4 Databankstructuur . . . 73
4.4.1 De OpenTaal-databank . . . 74
4.4.2 De statistiekendatabank . . . 74
4.5 Voorbeelden van use cases . . . 74
4.5.1 Invoer via het toetsenbord . . . 74
4.5.2 Webhulp . . . 75
4.5.3 Tekstherkenning . . . 75

4.5.4 Spraakherkenning . . . 76
4.5.5 Hulp voor sms- of e-mailberichten . . . 77
4.5.6 Zinsdeelselectie . . . 77
4.5.7 Filteren en multimediabestanden ophalen . . . 77
4.5.8 Multimediabestanden bekijken . . . 78
4.5.9 Statistieken bekijken . . . 78
4.6 Opbouw van de schermen . . . 79
4.7 Plugin structuur van bronnen, filters en hints . . . 79
4.8 Combineren van filters . . . 81
4.9 Het gebruik van de applicatie . . . 81
4.9.1 Gebruikersinstellingen . . . 81
4.9.2 Gebruikersinteractie doormiddel van vibratie . . . 83
4.9.3 Extra hints in tekstvorm . . . 83
4.9.4 Volgorde van de resulaten . . . 84
4.9.5 Landschapsmodus . . . 85
4.9.6 Pluginfunctionaliteit in Android . . . 85
4.9.7 Het loggen van activiteiten en resultaten in het programma . . . 86

5 User experience **88**
5.1 Soorten testen . . . 88
5.1.1 Sketchbook . . . 88
5.1.2 Wizard of Oz . . . 88
5.2 Praktijktest . . . 88
5.2.1 De toestellen . . . 89
5.2.2 De presentatie . . . 89
5.2.3 Voorbereidende test in Leuven . . . 89
5.2.4 Opmeten van de resultaten . . . 90
5.2.5 De testpersonen . . . 90
5.2.5.1 Communicatie met de kinderen . . . 91
5.2.5.2 Gsm gebruik . . . 91
5.2.5.3 Bruikbaarheid van de communicatiehulp volgens de testpersonen . . . 91
5.2.6 De begeleiders . . . 92
5.2.7 De testen . . . 92
5.2.7.1 Test 1 . . . 92
5.2.7.2 Test 2 . . . 92
5.2.7.3 Test 3 . . . 92
5.2.7.4 Test 4 . . . 93
5.2.7.5 Test 5 . . . 93

Besluit **94**

Lijst met afkortingen **96**

Lijst van figuren **98**

Lijst van codefragmenten **99**

A Contactpersonen **100**

B Poster **102**

C Broncode OpenTaal naar SQL **104**

Literatuurlijst **106**

Voorwoord

Graag wil ik mijn interne promotor, Veerle Ongenae van de Hogeschool Gent, bedanken voor de opvolging en de bruikbare feedback. Ook bedank ik graag mijn externe begeleiders, Adrián Juan, Toon De Pessemier, Kris Vanhecke en promotoren Wout Joseph en Luc Martens, voor hun tijd en energie.

Bedankt ook aan mijn tweede lezer, mevrouw Marleen Denert en ook mijn oprechte dank aan Emilie Van Dijck en de mensen waarmee ik kon samenwerken in de voorzieningen voor dove en slechthorende kinderen en jongeren. Met een bijzondere dank aan het BuSO Sint-Gregorius instituut waar de gebruikerstest mocht doorgaan. Ook Ruud Baars van het OpenTaal-project en Kathleen Pollefliet van de HoGent hebben mij uitgebreid geholpen met de opbouw van mijn masterproef.

Bedankt aan mijn vader voor zijn zoektocht naar spelfouten. Bedankt ook aan mijn moeder, mijn vriend Branko Vermote en zijn ouders voor het nalezen en de actieve steun.

Abstract

Children and young people who are deaf or hard of hearing don't master the spoken and written language well. Moreover their friends and relatives do not always speak sign language. In this work a communication aid on the Google Android mobile operating system is developed, that tries to reduce these barriers by converting text to relevant images and movies. These are found through online sources like Flickr, YouTube, Vimeo and Google Image Search.

The texts can be entered into the device through speech recognition and text recognition or by selecting pages from the Internet or e-mail and sms messages. By means of various methods the context of a sentence is determined. Finally movies and images are searched to explain that context.

Keywords

Android, mobile voice recognition, mobile text recognition, context of a sentence, context clarification, deaf kids, dyslexic children

Abstract

Kinderen en jongeren die doof of slechthorend zijn, beheersen de gesproken en geschreven taal niet zo goed. Bovendien beheersen hun vrienden en familieleden niet altijd gebarentaal. In dit werk wordt een communicatiehulp ontwikkeld op het besturingssysteem voor mobiele apparatuur van Google, Android, dat deze hindernissen moet reduceren door tekst om te zetten naar relevante plaatjes en filmbestanden. Deze worden gevonden op onlinebronnen zoals Flickr, YouTube, Google Image Search en Vimeo.

De teksten kunnen in het toestel ingegeven worden door middel van spraakherkenning en tekstherkenning of geselecteerd worden uit internetpagina's of e-mail- en sms-berichten. Via verschillende methoden wordt geprobeerd de context uit een zin te halen zodat het juiste verduidelijkende materiaal gevonden kan worden.

Trefwoorden

Android, mobiele spraakherkenning, mobiele tekstherkenning, context van een zin, context verduidelijking, dove en slechthorende kinderen, dyslectische kinderen

Inleiding

De verkoop van smartphones blijft fenomenaal stijgen. Steeds meer geïnteresseerden ontdekken de brede waaier aan functionaliteit van een dergelijk toestel. De dalende prijs ervan zorgt ervoor dat steeds meer mensen dit zich aanschaffen. De toestellen worden steeds krachtiger en de mobiele bandbreedte wordt goedkoper en veel gemakkelijker beschikbaar, waardoor complexere applicaties voor de toestellen ontworpen kunnen worden. Zo kunnen de smartphones in de toekomst ook meer en meer een hulp betekenen voor mensen met een taalachterstand.

Communicatie van dove of slechthorende kinderen met hun nabije omgeving loopt vaak niet optimaal. Ze vangen de gesproken communicatie onvoldoende op door hun auditieve beperking, ze beschikken over een beperkte taal en woordenschat om wat ze horen te begrijpen. Het complexe taalaanbod van de omgeving wordt vaak niet begrepen. Daarnaast zien we dat gebarentaal vaak toegankelijker is, maar hier stoten we op het probleem dat de meeste ouders van dove en slechthorende kinderen zelf niet doof zijn en de gebarentaal dus niet als moedertaal gebruiken.

Tot nu toe zijn er nog geen volwaardige communicatiehulpmiddelen beschikbaar die deze problemen kunnen reduceren. Bepaalde bronnen [13] geven nochtans aan dat doven bereid zijn om nieuwe technologie te gebruiken om de communicatie te verbeteren met horende personen. Verschillende auteurs [15] [17] [19] [21] verwijzen ook naar het belang van hulptechnologie om de communicatieproblemen te reduceren met dove personen.

Dove personen vormen een belangrijk segment van de populatie: de VUB heeft een universitaire studie uitgevoerd waaruit bleek dat er in het jaar 2003, 1.054.366 Vlamingen waren met gehoorproblemen. Op dat moment waren er 5.736.367 Vlamingen. Ongeveer 1 op 1000 personen wordt hier in Vlaanderen doof geboren [50].

In dit werk wordt een communicatiehulp ontwikkeld op het besturingssys-

teem voor mobiele apparatuur van Google, Android, dat deze hindernissen moet reduceren door tekst om te zetten naar relevante plaatjes en filmbestanden. Deze worden gevonden op onlinebronnen zoals Flickr, YouTube, Google Image Search en Vimeo.

De teksten kunnen in het toestel ingegeven worden door middel van spraakherkenning en tekstherkenning of geselecteerd worden uit internetpagina's of e-mail- en sms-berichten. Via verschillende methoden wordt geprobeerd de context uit een zin te halen zodat het juiste verduidelijkende materiaal gevonden kan worden.

Hopelijk kan de applicatie die ontworpen werd in dit eindwerk, bijdragen tot de ontwikkeling van toekomstige, vergelijkbare projecten.

Mijn voorkeur ging uit naar dit onderwerp omdat dit onderzoek naar communicatiehulpmiddelen voor dove en slechthorende personen een zeer praktisch onderzoek is met een belangrijke maatschappelijke en sociale relevantie.

Hoofdstuk 1

Wireless & Cable (WiCa)

De onderzoeksgroep Wireless & Cable van het Interdisciplinair Instituut voor Breedband Technologie (IBBT) staat onder leiding van prof. Luc Martens en voert onderzoek uit op verschillende domeinen in verschillende niveaus van het OSI-model: onderzoeken op o.a. de fysische laag van draadloze communicatie, nieuwe interactieve media-applicaties en onderzoek naar technologiën voor *hybride coax*-netwerken.

Ook aanbevelingssystemen worden in het IBBT onderzocht. Het steeds groeiende aanbod van video's op internet en tv vereist namelijk nieuwe algoritmen om de consumenten beelden te laten ontdekken die ze interessant vinden. Voor elke gebruiker kan een profiel worden aangemaakt dat zijn interesses weerspiegelt. Bij de impliciete feedback wordt onderzocht waarnaar de gebruiker kijkt, bij de expliciete worden beoordelingen gegeven.

Mobiele applicaties met draadloze netwerkfunctionaliteit worden steeds belangrijker. Steeds meer applicaties worden uitgebracht. Waarom een applicatie meer succes heeft dan een andere, hangt niet enkel af van de technische eigenschappen, maar ook steeds meer van de Quality of Experience (QoE), het gebruiksgemak waarmee de applicatie te gebruiken is.

Klimaatveranderingen hebben ervoor gezorgd dat er meer aandacht gaat naar energieverbruik. Het WiCa voert onderzoek naar de reeds aanzienlijke en snel groeiende ecologische consequenties van de informaticatechnologie.

Een Wireless Body Area Network (WBAN) verbindt onafhankelijke *nodes* (bijvoorbeeld sensoren en aandrijvingen) die zich bevinden in de kledij en op of onder de huid van een persoon. Hierdoor is een breed palet aan veelbelovende nieuwe toepassingen mogelijk: automatische opvolging van de gezondheidstoestand, multimediatoepassingen of opmeting van sportresultaten.

WiCa onderzoekt ook de vereisten van de toekomstige draadloze bandbreedte. Wat zijn de gevolgen van de straling op de menselijke gezondheid? Welke materialen houden de stralingen tegen? Hoe kunnen de vermogens van de draadloze zenders dynamisch worden gemaakt om stroom te besparen?

Op dit moment zijn in de onderzoeksgroep achttien mensen in dienst, waarvan de meeste de titel *Research Engineer* dragen. Er werden 292 publicaties uitgebracht.

Hoofdstuk 2

Situering

2.1 Probleemstelling

Kinderen en jongeren die doof of slechthorend zijn beheersen de gesproken en geschreven taal niet zo goed. Sommige dove of slechthorende kinderen of jongeren communiceren in gebarentaal, maar die taal is niet altijd zo goed gekend door familie en kennissen. Hierdoor is het moeilijk om te communiceren tussen dove kinderen en hun vrienden en familie. Ook de communicatie tussen doven en vreemden loopt niet van een leien dakje, maar deze vorm van communicatie wordt niet behandeld[1].

Ook het begrijpen van teksten in folders, informatieborden of boeken kan voor hen moeilijk zijn.

De voorgestelde applicatie heeft tot doel deze hindernissen reduceren door spraak en tekstherkenning in te zetten om relevante plaatjes en filmpjes te vinden. Deze multimediabestanden worden gezocht op online bronnen zoals Flickr, YouTube, Google, Vimeo, .. en kunnen helpen om te illustreren wat de gebruiker aan iemand anders probeert over te brengen. Verschillende technologieën zijn hiervoor ingezet, waaronder Google Android (Java) en publieke APIs.

Daarnaast is ook een ondersteuning gerealiseerd om incorrect taalgebruik te verbeteren. Zinnen kunnen ingegeven worden waarna de mogelijkheid bestaat om een woord aan te klikken. Dat woord wordt dan uitgebeeld in afbeeldingen en filmpjes. Zo kan door de gebruiker geverifieerd worden of de juiste betekenis wel gebruikt is.

[1]Door het aantal mogelijke situaties waar de applicatie in gebruikt kan worden beperkt te houden, kunnen deze meer geoptimaliseerd worden.

2.2 Doelstelling

Het hoofddoel van deze masterproef was om een applicatie te ontwikkelen voor Android, het besturingssysteem voor mobiele apparatuur van Google. Dit programma is in staat om de communicatieproblemen van dove en slechthorende kinderen te reduceren.

De gebruiker kan op verschillende manieren gegevens in de applicatie invoeren: via directe invoer, het maken van een selectie (uit een sms, e-mailbericht of uit een webpagina) of door middel van tekst- of spraakherkenning. Daarna zal een filter de kernwoorden uit de zin trachten te halen door onder andere gebruik te maken van een woordenboek en intelligente software die woordsoorten en andere kenmerken aanduidt. Nadien wordt een navigeerbare verzameling multimediabestanden weergegeven die de context van de zin proberen te verduidelijken.

Het doel is om de gebruiker relevante video's en afbeeldingen te laten selecteren die de conversatie kunnen verduidelijken.

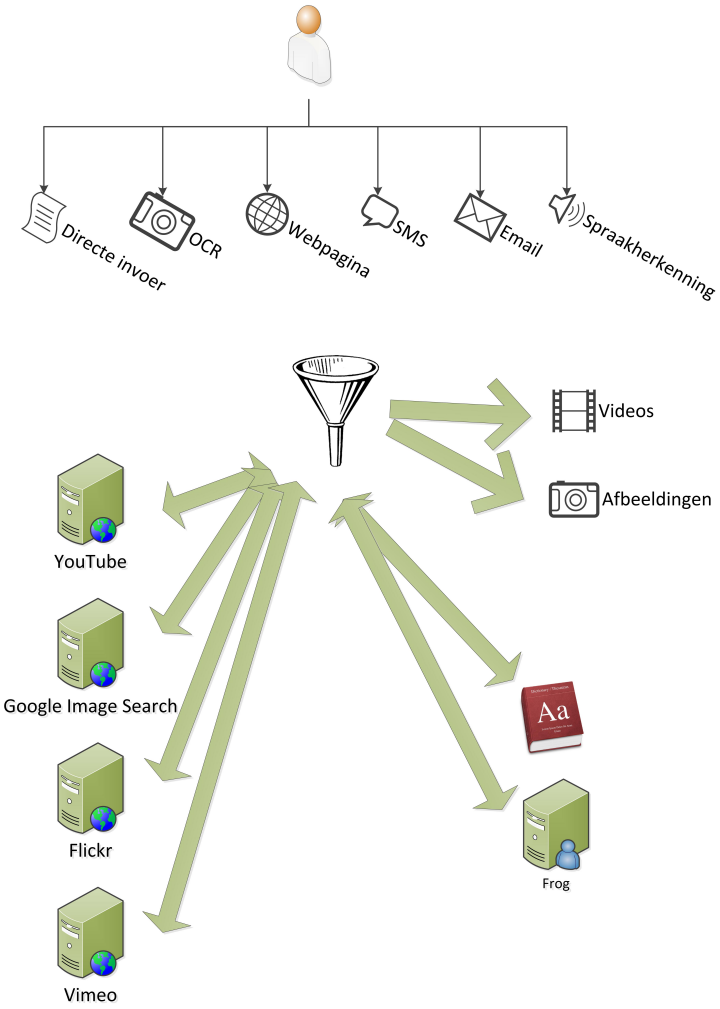
2.3 Praktijk

De *use cases* voor de voorgestelde applicatie zijn opgesteld in samenspraak met begeleiders van voorzieningen voor dove en slechthorende kinderen en jongeren. Samen met Emilie Van Dijck zijn in een later stadium gebruikersevaluaties afgenomen (zie hoofdstuk 5). Zo is de applicatie geëvalueerd door de doelgroep. Hopelijk kunnen deze resultaten als inspiratie helpen bij toekomstige vergelijkbare projecten.

Emilie Van Dijck, student aan de KU Leuven, met een vooropleiding als logopedist-audioloog, moet een conceptueel model evalueren op vlak van usability. Ze werkt dus aan een analoog onderwerp, maar dan aan de niet-technische kant. Zij heeft de bevragingen opgesteld en aan de hand daarvan zijn de gebruikersevaluaties afgenomen.

2.4 Gedetailleerde omschrijving van de opdracht

Eerst is er een literatuurstudie uitgevoerd naar de beschikbare bronnen van multimediabestanden op het internet (hoe deze gebruikt kunnen worden) en de beschikbare bibliotheken voor het Google Android platform. Ook is



Figuur 2.1: Abstracte voorstelling van de communicatiehulp.

er bestudeerd hoe men relevante woorden uit een tekst herkent en wat de noden zijn van dove en slechthorende kinderen. Daarna is de applicatie opgebouwd in verschillende fasen. Elke fase bevatte wat meer functionaliteit in vergelijking met de vorige fase.

In een eerste fase werd een applicatie ontwikkeld waar een zin kon ingegeven worden via het toetsenbord en dan doorgestuurd werd voor analyse.

In een tweede fase werd dan de navigeerbare verzameling ontwikkeld die de resultaten moest weergeven.

In een derde fase werd het mogelijk een zin te selecteren uit een internetapplicatie. Die zin werd dan doorgestuurd voor analyse.

In de volgende stadia werden dan andere inputbronnen toegevoegd zoals spraakherkenning, tekstherkenning en input via sms en e-mail.

De vereisten en use cases zijn opgesteld in samenspraak met Emilie Van Dijck en experten gerelateerd aan de doelgroep.

2.5 Problemen die opgelost moesten worden

Het was bij de ontwikkeling essentieel dat onderzocht werd hoe de geschreven en gesproken tekst zo efficiënt mogelijk werd verwerkt zonder de batterij van de mobiele telefoon te zwaar te belasten. Het moet mogelijk zijn de applicatie de hele dag door te gebruiken zonder dat de gebruiker zijn toestel moet opladen. Om de applicatie vlot te laten draaien, stellen we voorop dat een 3G-verbinding en een 1Ghz-processor de minimale vereisten zijn van het toestel. De vooropgestelde applicatie kan Android versie 2.1 als minimale versie eisen, aangezien een groot deel van de huidige toestellen versie 2.1 of hoger draaien. Spraakherkenning zit in Android sinds versie nummer 1.5.

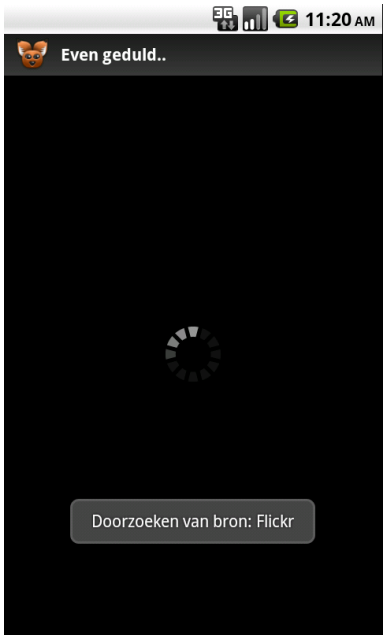
Een studie is uitgevoerd naar wat de kernwoorden in een tekst inhouden, welke woorden de communicatie verduidelijken en welke van die woorden meestal onduidelijk zijn voor de doelgroep. Vermoedelijk heeft het slechte tekstbegrip te maken met de kleine-, voor ons schijnbaar onbelangrijke woorden, die de betekenis van een zin kunnen kleuren, zoals voegwoorden en voorzetsels. Deze worden moeilijk begrepen, waardoor de zin vaak fout geïnterpreteerd wordt.

Ook bij het weergeven van de multimediabestanden moest aandacht besteed worden aan de beperkingen van de hardware van het mobiele toestel. Er moest namelijk rekening gehouden worden met de beperkte batterijduur, de grootte van het scherm, de beschikbare mobiele bandbreedte en de hoeveelheid dataoverdracht. De mobiele applicatie is afhankelijk van informatie uit verschillende externe bronnen. Het is belangrijk dat de webservice verzoeken efficiënt uitvoert, waardoor het aantal aanvragen beperkt blijft.

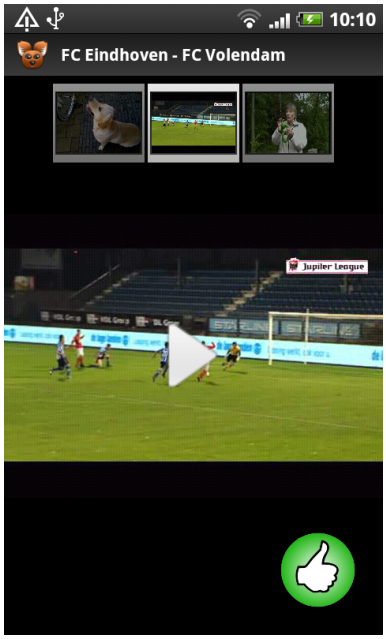
Android is nog relatief jong en er zijn nog niet zoveel (gratis) ontwikkelde bibliotheken om goed aan spraak- en tekstherkenning te doen. Onderzoek heeft plaatsgevonden naar hoe deze functionaliteit toch in de applicatie kon geplaatst worden met een realistisch insteek aan inspanningen.

Er moest een gezonde balans gevonden worden tussen functionaliteit die bruikbaar is voor de kinderen en functionaliteit die bruikbaar zou kunnen zijn, maar niet voldoende uitgewerkt kon worden door technische beperkingen.

De applicatie moest zo zelfstandig mogelijk kunnen werken, gebruikersvriendelijk zijn en het vragen van input aan de gebruiker tot een minimum beperken.



Figuur 2.2: Verwerking van de input.



Figuur 2.3: Navigeerbare verzameling resultaten.

2.6 Doelgroep

De applicatie werd ontwikkeld voor dove jongeren (die vooral communiceren met gebarentaal) bij wie het niveau van Nederlandse taalverwerving minder hoog is: jongeren die over een beperkte woordenschat beschikken en die moeite hebben met het begrijpen van Nederlandse zinsstructuren.

Als referentie zijn dove jongeren tussen 12 en 18 jaar genomen, die staan voor begrijpend lezen ongeveer even ver als leerlingen van het regulier derde leerjaar.

2.7 Beperkingen

De spraakherkenning werkt nog niet optimaal. Door de beperkte rekencapaciteit van de toestellen moet de herkenning op een andere machine gebeuren. Nederlandstalige spraakherkenning is nog niet zo lang in ontwikkeling als de Engelstalige en de mogelijkheden om dit op een mobiel toestel te doen zijn nog zeer beperkt. De herkenning werkt ook enkel als er langzaam een mooie Nederlandstalige zin ingesproken wordt en de omgeving van het toestel vrij is van ruis.

In een ideale applicatie zou de dove persoon boodschappen kunnen inbrengen met behulp van een gebarentaalherkenningssysteem. Praktisch is dat voorlopig nog te complex, want de gebarentalen worden meestal bijzonder vlug uitgebeeld en kleine nuance verschillen zijn soms zeer belangrijk. Daarnaast zijn gebarentalen meestal verbonden aan een kleine regio en verschillende personen met een verschillend postuur zouden de gebaren moeten kunnen uitbeelden zodat deze herkend worden.

2.8 Een psychologisch perspectief op doofheid

In deze sectie worden enkele aspecten behandeld die belangrijk kunnen zijn bij de ontwikkeling van de communicatiehulp en het onderzoek dat ernaar gebeurd is in het kader van deze masterproef.

Er werden verschillende pogingen ondernomen om het onderscheid tussen doofheid en slechthorendheid objectief vast te leggen.

- **Audiologisch standpunt**
  Er zijn auteurs die een dove persoon kenmerken als iemand waarbij een gemiddeld gehoorverlies van 90 decibel gemeten wordt, in het beste oor, voor de belangrijkste spraakfrequenties[2]. Daar is nog een definitie bijgekomen. Van Uden deelt de kinderen op in twee categorien: zie-kinderen en hoor-kinderen[3].

[2]In een geluidscabine kunnen audiometrische tests afgenomen worden die uitgetekend worden in een audiogram. Hiermee kan vervolgens ook de graad van de stoornis bepaald worden. Later is men bij de categorisatie ook rekening gaan houden met de mate waarmee iemand met gehoorverlies, geholpen door hoorapparaten, nog spraakklanken kan waarnemen.

[3]'Hoor-kinderen zijn gehoorgestoorde kinderen die de omgangstaal hoofdzakelijk verstaan door het horen. Het liplezen is echter wel een grote noodzakelijke steun. Zie-kinderen

- **Medisch-biologisch standpunt**
  Een tweede onderscheid kan gemaakt worden op basis van een medisch-biologisch standpunt, namelijk een benadering volgens de plaats van het letsel en de oorzaak van het gehoorverlies. Doofheid kan dan gezien worden als een quasi totale opheffing van de geluidswaarneming.

- **Standpunt van linguïsten**
  Linguïsten besteden voornamelijk aandacht aan de mate waarop doofheid impact heeft op het taalverwerkingsproces. Daardoor maken zij het onderscheid volgens het tijdstip waarop het gehoorverlies is opgetreden. Men spreekt van preliminaire doofheid als deze optrad voor de leeftijd van 2 jaar. Anders spreekt men van postlinguale doofheid.

- **Het sociaal-cultureel model**
  Vanuit het sociaal-cultureel model stelt men dat er geen handicap is in het individu, maar wel in de relatie met de omgeving. Doofheid wordt gedefinieerd als een verschil, dat communicatieproblemen met zich meebrengt.

In de opvoeding en het onderwijs aan dove kinderen staan twee verschillende opvoedingsvisies scherp tegenover elkaar:

- Een eerste benadering beschouwt de dove mensen als een variant binnen de horende wereld. Men wil ze stimuleren, zodat ze optimaal kunnen integreren in de horende wereld. Het kind moet zo vlug mogelijk in een omgeving vertoeven waarin enkel gesproken taal gebruikt wordt als communicatiemiddel. Voorstanders van deze methode staan eerder weigerachtig tegenover het pedagogisch gebruik van gebaren.

- In een bilinguale en biculturele benadering beschouwt men het vroegdove kind niet per definitie als taalgestoord. Prioriteit wordt gegeven aan de communicatie tussen de moeder en het jonge kind. Men hecht belang aan de waardering binnen de dovengemeenschap. Dove volwassenen worden ingezet als partners in de opvoeding. Hierin is het wel belangrijk dat de horende ouders de gebarentaal kennen.

Momenteel groeien beide visies sterk naar elkaar toe, waardoor mengvormen ontstaan.

Door middel van taal kunnen mensen contacten leggen, een behoefte of een wens uitdrukken, protesteren, problemen oplossen en eigen en andermans

zijn kinderen voor wie het liplezen een hoofdzaak is en het horen (met een hoorapparaat) een steun' - Van Uden 1990

gedrag benvloeden. Als de communicatie op jonge leeftijd misloopt, kunnen bij kleuters en peuters al snel sociaal-emotionele moeilijkheden optreden. Enkele kenmerken van de dovenpopulatie [5]: het gemiddelde performale IQ bedraagt 115 en ongeveer 75 procent eindigt zijn onderwijsloopbaan in het buitengewoon onderwijs. Jongere generaties behalen echter gemiddeld meer hogere diploma's door het stijgend aantal jongeren dat geïntegreerd wordt in het 'gewoon' onderwijs. De meeste dove jongeren bekomen primair informatie via de televisie en ongeveer 79 procent leest geen boeken. Niettegenstaande 75 procent thuis of op school oraal opgevoed werden, gebruikt toch 84 procent hoofdzakelijk gebaren.

2.8.1 Gesprekken op het werkveld

Om de applicatie zoveel mogelijk te kunnen afstemmen op de noden van de doelgroep zijn voorafgaand aan de ontwikkeling enkele gesprekken gebeurd met mensen die verschillende functies bekleden in verschillende instituten. De vergaderingen werden telkens opgebouwd uit dezelfde verzameling vragen:

- Wat zijn de noden van dove kinderen en hoe uiten ze deze?
- Hoe communiceren dove kinderen met elkaar en met horende mensen?
- Hoe kan de communicatie verbeterd worden?
- Wat is de werking van de instelling?
- Zijn er zaken die de kinderen nog niet zelfstandig kunnen?
- Hoe kan een mobiele applicatie plezant zijn en wat mag er zeker niet in ontbreken?
- Zijn de kinderen actief met nieuwe technologie bezig?
- Met welke soort woorden hebben de kinderen de meeste moeilijkheden?
- Hoe zou een mobiele applicatie voornaamwoorden en lijmwoorden (samenstellingen) kunnen verduidelijken?

Hierna komt een overzicht van de instellingen waar contacten mee gelegd zijn, samen met enkele specifieke eigenschappen of bemerkingen van de instelling. De informatie die bekomen is uit de gesprekken is verwerkt op verschillende plaatsen in de scriptie. Er waren contacten met mensen met verschillende functies binnen hun werk.

In de scriptie wordt verwezen naar deze secties, wat betekent dat die informatie verkregen is na gesprekken met contactpersonen uit die instellingen. De gegevens van de contactpersonen zijn terug te vinden in bijlage A.

2.8.1.1 Koninklijk Instituut voor Doven en Spraakgestoorden (KIDS)

KIDS is een verzameling van een aantal instellingen die onder andere bezig zijn met het integreren van de dove kinderen in de samenleving. Gebarentaal was er geruime tijd taboe, opdat de kinderen zoveel mogelijk de gesproken taal zouden leren [2].

De slechthorenden hebben een eigen cultuur en gebruiken eigen uitgaansgelegenheden. Het KIDS wil slechthorenden niet als een selectieve culturele groep en niet als mensen met een handicap beschouwen.

In het KIDS is er gesproken met Stefan Coenen, verantwoordelijke voor onder andere externe hulpmiddelen op school en thuis bij de kinderen.

2.8.1.2 Het GR@SP project

Het GR@SP project is een nieuw project rond nieuwe media voor doven en slechthorenden.

Quality of Experience (QoE), een van de twee termen die centraal staan in het project, handelt over de ervaring die gebruikers ondervinden met een bepaalde toepassing, terwijl Quality of Service (QoS) de kwaliteit van de dienst die de toepassing aanstuurt bespreekt.

Het zijn beide belangrijke elementen in het moderne innovatieproces, met name voor onderzoek naar diensten die aan eindgebruikers aangeboden worden. Om deze kwaliteitsnormen te verhogen is het belangrijk om gebruikers meer te betrekken bij het ontwikkelingsproces van een applicatie.

De termen worden helaas nog vaak door elkaar gebruikt dus heeft het project tot doel de kloof tussen beide termen te dichten.

Het GR@SP project wil een soort model en methodologie ontwikkelen voor QoS en QoE en een visie over wat deze juist inhouden bij mobiele applicaties. Hierbij zal rekening gehouden worden met de verwachtingen van de gebruiker.

2.8.1.3 MPI Sint-Lievenspoort

Op het MPI Sint-Lievenpoort is er een interessante vergadering geweest met een audiologe-logopediste, een orthopedagoge en een stagiair die zelf doof is. Hier is veel documentatie verkregen over de taalontwikkeling, alfabetisatie, psychologie en sociaal-emotionele ontwikkeling van dove en slechorende kinderen. Er werd gepraat over correcte terminologie met betrekking tot de doelgroep.

2.8.1.4 BuSO Sint-Gregorius

Jongeren van groepen OV1 en OV2 werden hier als geschikte doelgroep bevonden voor de applicatie, maar GON-jongeren minder. OV1 zijn jongeren die voorbereid worden op het functioneren binnen een instelling. OV2-jongeren zullen waarschijnlijk tewerkgesteld worden in een beschutte werkplaats. GON staat in voor het begeleiden van jongeren binnen normale onderwijsvormen.

- **OV1** jongeren hebben geen goede zinsbouw door hun gebrekkige grammaticakennis. Een verklarend beeldwoordenboek zou bij deze doelgroep wel handig zijn door hun beperkte woordenschat.
- De **OV2** groep heeft een betere zinsbouw.
- **OV3** kan soms nog moeilijk onderscheid maken tussen bepaalde moeilijkere woorden, zoals bijvoorbeeld het verschil tussen terrorist en toerist.

In deze instelling mocht de applicatie door de leerlingen uitgetest worden zodat hierover bevindingen konden opgeschreven worden (zie hoofdstuk 5 op pagina 88).

2.8.2 Problemen in het dagelijkse leven

Het is voor dove personen moeilijk om een expert te raadplegen zonder een horende mee te nemen. Denk hierbij aan het raadplegen van een arts, een advocaat of een notaris. Ze kunnen zich niet goed verstaanbaar maken of begrijpen niet wat de expert aan hen wil uitleggen. Dit komt niet enkel doordat ze het gesprek niet kunnen horen, maar vooral omdat hun woordenschat beperkt is en ze de betekenis van moeilijke woorden niet kennen (zie referentie 2.8.1.1).

Een ander probleem vormt het gebruik van het openbaar vervoer. In een treinstation worden last-minute wijzigingen vaak enkel afgeroepen en dus niet

steeds weergegeven op de informatieborden. Ook al hebben sommige dove of slechthorende treingebruikers een hoorapparaat of cochleair inplantaat (CI), toch blijft het moeilijk om deze boodschap te begrijpen, omdat er bij het afroepen een hoop echo gegenereerd wordt en er veel achtergrondlawaai is. Ze kunnen namelijk best helder geluid waarnemen met hun implantaten (zie referentie 2.8.1.1 en 2.8.1.3).

Een laatste probleem dat ik behandelde situeert zich bij cultuur en ontspanning. Dit beleven zij op een andere manier. Veel musea bieden namelijk extra uitleg aan via een audio guide of een gids. Ook op de beeldbuis kan je maar voor een aantal programma's ondertitels activeren (zie referentie 2.8.1.1).

2.8.3 Onderlinge communicatie

Sommigen communiceren onderling met gebarentaal, terwijl andere gewoon spreken. Dat hangt af van de mogelijkheden en van waar hun voorkeur naar uit gaat (zie referentie 2.8.1.1).

2.8.4 De context van een tekst

Het is niet eenvoudig om te achterhalen welke woorden verduidelijkt moeten worden, zodat de context van een zin duidelijk wordt. De gebruiker kan dit meestal moeilijk zelf aangeven, doordat een gebruiker een deel van een zin niet begrijpt omdat de verwijswoorden niet begrepen worden. Het uitbeelden van verwijswoorden is echter zeer moeilijk.

Het kan ook zijn dat de gebruiker een zin niet begrijpt, maar dat het onbegrip veroorzaakt wordt door een zin die eerder in de tekst voorkwam (zie referentie 2.8.1.2).

Figuurlijk taalgebruik is ook moeilijk te begrijpen voor dove jongeren, maar dit soort informatie is voor een mobiele applicatie vermoedelijk niet eenvoudig te verklaren (zie referentie 2.8.1.4).

De zelfstandige naamwoorden zijn natuurlijk het eenvoudigst te verduidelijken met multimediamateriaal.

2.8.5 Invloed van gebarentaal op de geschreven taal

De jongeren ondervinden soms problemen bij het terugvinden van de infinitief achter een (meestal in de verleden tijd) vervoegd werkwoord. Als wij zeggen 'liep', is dat in gebarentaal eerder iets in de vorm van 'vroeger lopen'. Zo

wordt in gebarentaal het woord 'welke kleur' bijvoorbeeld vervangen door 'kleur wat' (zie referentie 2.8.1.3). De term 'welke' is namelijk moeilijk uit te leggen door middel van het weergeven van multimediamateriaal, maar 'kleur wat' legt direct de bedoeling van de zin uit.

2.8.6 Workflow van de voorgestelde applicatie

Om de gebruikers zo snel mogelijk door de applicatie te laten navigeren, kan best zoveel mogelijk gebruik gemaakt worden van korte Nederlandstalige zinnen en een goed gebruik van pictogrammen (zie referentie 2.8.1.1, 2.8.1.3).

Het kan een goed idee zijn om zinnen of zinsdelen die moeilijk in afbeeldingen om te zetten zijn te vervangen door een Nederlandstalige zin met een vergelijkbare structuur als bij gebarentaal. Toch is het belangrijk om ook hiervoor correcte Nederlandstalige zinnen te gebruiken om de jongeren geen verkeerd Nederlands aan te leren.

De kinderen in Gentbrugge zijn het naar verluidt al gewoon om afbeeldingen van woorden die ze niet begrijpen op te zoeken in Google Image Search.

2.8.7 Technische hulpmiddelen

De meeste dove kinderen kunnen tegenwoordig beperkt horen met behulp van een cochleair implantaat. Een CI wordt ingepland als de kinderen een aantal maanden oud zijn.

Het plaatsen van een CI in één oor wordt terugbetaald, maar de ouders opteren er meestal voor om ook te investeren in een CI voor het andere oor (onder een bepaalde leeftijd wordt dat ook terugbetaald) (zie referentie [2]). Met behulp van zo'n CI kunnen de kinderen een deel van de klanken horen die wij horen.

Het implantaat kan overigens ook niet bij iedereen ingeplant worden. De auditieve zenuw moet wel nog geleiden voordat het kan werken.

De gehoorprothesen schieten wel tekort op drie vlakken: ze kunnen geen gevoeligheid of richting detecteren (als ze horen met een oor) en zijn minder selectief als het menselijk gehoor. Dit is moeilijk in een lawaaierige omgeving. Groepsgesprekken vormen daarom meestal een probleem, aangezien doven moeilijk horen wie er aan het spreken is of wat er gezegd wordt. Daarom gebruiken de leerkrachten een microfoon. De hoorapparaten kunnen dan

door middel van inductie die geluiden opnemen en de dove ontvangt dan een helderder geluid.

Er zijn ook allerhande apparaten die auditieve signalen omzetten naar visuele signalen zoals wekkers, deurbellen, melders voor een telefoongesprek, etc.

2.8.8 Liplezen

Het aantal doven dat goedhorende mensen probeert te begrijpen door middel van liplezen daalt (zie referentie 2.8.1.1).

2.8.9 Gebarentalen

Gebarentaal is een natuurlijke taal, ontstaan binnen de dovengemeenschap. De verwerking ervan gebeurt in dezelfde hersenzones als van gesproken talen. Gebarentaal is niet universeel en is een taal op zich, onafhankelijk van gesproken talen. Zowel in Vlaanderen als in Wallonië is er sprake van een gebarentaal (BeGT) met dialectvarianten. Deze onderscheidt zich van de Nederlandse Gebarentaal.

Gebarentaal is als het ware de moedertaal voor dove kinderen, de invulling van het lexicon komt slechts later. De jongeren met het beste taalgevoel hebben voornamelijk dat gevoel ontwikkeld door veel boeken te lezen (zie referentie 2.8.1.4). Ook de onderlinge communicatie tussen de dove kinderen gebeurt meestal (niet altijd) met behulp van gebarentaal.

Gebarentaal wordt soms aan doven met een implantaat aangeleerd, omdat er een defect aan het implantaat kan optreden.

2.8.10 Gebarensystemen

Bij gebarensystemen zijn er niet enkel afspraken over welke gebaren aan de woorden van de gesproken taal gekoppeld zullen worden, maar ook over welke gebruiksregels.

Die regels hebben voornamelijk te maken met de visuele voorstelling van syntactische of morfologische kenmerken van de gesproken taal. Zo wordt een verkleinwoord door een J-gebaar gevisualiseerd. Dit J gebaar volgt op het gebaar dat als verkleinwoord voorgesteld moet worden. Hoe meer van deze regels toegevoegd worden, hoe meer deze op de gesproken taal gaat lijken. Hierdoor wordt dit gebarensysteem logger en minder soepel in gebruik.

Gesproken talen worden dus ondersteund door gebaren. Het spreken van de gesproken taal gebeurt dan volgens de grammaticale regels van de gesproken taal en niet volgens die van de gebarentaal. Deze vorm wordt ook wel "Simultane Communicatie" genoemd.

Voor dove mensen kunnen deze systemen zeer verwarrend lijken door de verschillende zinsconstructies. Daarom zijn deze meestal niet zo populair en wordt voor onderlinge communicatie meestal de voorkeur gegeven aan de natuurlijke gebarentalen [**?**]

2.8.11 Woordenschat van dove personen

Uit onderzoek is gebleken dat een dove persoon een woordenschatbereik heeft van zo'n 5800 woorden. Er zijn woordenboeken uitgebracht waar deze woorden in vermeld staan, samen met een afbeelding die het woord moet verduidelijken.

- **Fevlado** heeft zo'n boek uitgewerkt, maar de copyrights hierop maken het moeilijk om de afbeeldingen te gebruiken.
- Het woordenboek dat ontwikkeld is aan **de Rijksuniversiteit Gent** heeft als nadeel dat er veel te veel varianten in staan waar een keuze uit gemaakt moet worden.
- Er is ook een commercieel bedrijf die zo'n woordenboek als boek heeft laten publiceren.

2.8.12 Bedreigde verwerving van gesproken taal en het taalbegrip

Kinderen met een gehoorstoornis kunnen moeilijker toegang krijgen tot gesproken taal, aangezien de moedertaalontwikkeling zich afspeelt binnen de gewone communicatie. Peuters hebben normaal een drievoudige opdracht binnen de communicatie: ze moeten de taal verwerven, leren begrijpen en vervolgens ook gebruiken.

Bij slechthorende kinderen kunnen deze problemen vroegtijdig opgevangen worden met een hoorapparaat. Slechts een klein percentage van de dove kinderen heeft een of twee ouders of grootouders waarmee hij kan communiceren in gebarentaal. Deze kinderen maken daardoor een grotere kans om toch over een moedertaal te beschikken en dus ook een volwaardige 'eerste' taal te beheersen. Ze beschikken met andere woorden over een volwaardig "taalbegrip".

2.8.13 Leren lezen

Op een gewone school leren kinderen meestal lezen vanaf zes jaar. De horende kinderen hebben dan genoeg kennis over de gesproken taal en kunnen daarover reflecteren. Bij prelinguaal dove kinderen verloopt dit leesproces veel moeizamer door hun gebrekkige kennis van de gesproken taal. In het onderwijs aan dove en slechthorende kinderen wordt lezen gezien als cruciale ingangspoort om tot de verwerving van het Nederlands te komen. Vandaar dat men ook poogt het leesproces reeds in de kleuterleeftijd in gang te zetten.

2.9 Bestaande projecten

2.9.1 I-CITY

I-CITY is een mobiele applicatie ontwikkeld voor de stad Hasselt waar ook een deeltje voorzien was voor dove mensen. Je kan bijvoorbeeld een losliggende steen doorgeven aan de gemeente of interessante winkelpromoties ontdekken in de winkelstraat.

Voor de ontwikkeling ervan is samengewerkt met KIDS (zie referentie 2.8.1.1).

2.9.2 Draadloze videoconversaties

Op de computer wordt bij de doelgroep voornamelijk een programma gebruikt wat te vergelijken is met het bekende Skype VoIP programma: ooVoo. ooVoo ziet er grafisch erg mooi afgewerkt uit en heeft voornamelijk dove personen als klant. De mobiele versie draait enkel op Windows Mobile en niet op Android.

Dove volwassen hebben reeds experimenten uitgevoerd met gebarentaal via een gsm. Technische beperkingen zorgden er echter voor dat dit vooralsnog geen succes blijkt te zijn.

De technische beperkingen bij de huidige videoconversaties zijn de snelheid en het aantal frames per seconde die doorgestuurd kunnen worden. Gebarentaal hangt erg af van nuanceverschillen en als de kwaliteit niet voldoende hoog is zijn die moeilijk om te zien (zie referentie 2.8.1.2). Ook kan men bij het gebruik van een mobiel toestel slechts één hand gebruiken aangezien men het toestel vasthoudt met de andere hand (zie referentie 2.8.1.3).

Natuurlijk vormen ook de beschikbaarheid en de kostprijs van de bandbreedte een probleem.

2.9.3 Oralys op pocket PC

De Mobile Communicator software voor pocket PC van Oralys is ontwikkeld om directe communicatie gemakkelijk te maken, maar enkel van de dove persoon naar de horende persoon toe. Om een gesproken boodschap te communiceren in het Engels, Frans of Spaans [6] moet de dove persoon pictogrammen aanduiden op het aanraakscherm dat de objecten of situaties representeert. De volgorde waarin dit kan gebeuren, hangt samen met de structuur van de gebarentaal. In dit geval hangt deze samen met de gebarentaal van Quebec. Momenteel bevat de applicatie reeds 3800 pictogrammen.

Als de zin geconstrueerd is, kan de dove persoon een film bekijken waarin een man de zin in gebarentaal toont. Zo kan hij verifiëren of dit inderdaad de boodschap is die hij wil overbrengen. Indien hij tevreden is, dan wordt de zin door het toestel luidop uitgesproken zodat de horende persoon deze kan horen.

Figuur 2.4: Een PocketPC met de Mobile Communicator van Oralys Inc. [6]

Er is onderzoek gedaan [48] naar de invloed op de sociale omgang met behulp van een *assistive technology* (AT) gericht op de de communicatie tussen een horende en een dove persoon. De voorwaarde was dat die dove persoon niet kon lezen, schrijven of liplezen. Vijftien proefpersonen die aan de voorwaarden voldeden, kregen een PocketPC met software van Oralys [6] op geïnstalleerd. Het onderzoek liep gedurende drie maanden en de invloed van de AT werd gemeten met behulp van een aantal standaard testmethoden (LIFE-H, FACS, QUEST).

Bij aanvang van het onderzoek werd een eerste versie van de software gebruikt die nog niet commercieel verkocht werd (in april 2011 wordt ze ondertussen wel al verkocht).

Na verloop van tijd gebruikten steeds minder mensen de AT, maar de sociale en functionele communicatie met horende personen ging er een klein beetje op vooruit. De meeste proefpersonen vonden de nieuwe technologie niet bevredigend, maar ook niet slecht.

2.9.4 Sprint en Kuzweil 3000

Deze programma's, ontwikkeld voor jongeren die kampen met dyslexie, werken een beetje gelijkaardig als de te ontwikkelen applicatie. Er kunnen zinnen ingegeven worden en nadien kan men woorden aanklikken om de uitleg van een woord op te vragen. Die uitleg kan aangevuld worden met afbeeldingen. Woorden die foutief gespeld zijn worden ook volautomatisch verbeterd.

Op het BuSO Sint-Gregorius instituut in Gentbrugge wordt Sprint al gebruikt tijdens de lessen.

Hoofdstuk 3

Gebruikte technologiën

3.1 Google Android

Google Android is een opensourceplatform voor mobiele telefoons. Dit platform is gebaseerd op de Linux-kernel en het Java-programmeerplatform. Android is op dit moment het meest verkochte besturingssysteem op mobiele telefoons. Applicaties kunnen op het platform gemakkelijk verspreid worden via de Android Market. Om een applicatie te ontwikkelen op Google Android kan je JAVA gebruiken samen met een uitgebreid aanbod aan bibliotheken die het Java besturingssysteem ter beschikking stelt.

Er zijn op dit moment al erg veel Android toestellen op de markt en er komen er nog veel meer aan. Verschillende fabrikanten bieden toestellen met Android aan [11] zoals HTC, Samsung, Motorola, LG, Sony Ericsson, Google en T-Mobile. De prijzen voor Android toestellen variëren in april 2011 tussen 105,05 en 838 euro.

3.1.1 Android componenten

In Google Android zijn er vier types van componenten [3]. Deze worden in deze sectie kort besproken.

3.1.1.1 Activities

Een activity is te vergelijken met een venster of een dialoogvenster in een andere applicatie. De Activity klasse creëert een venster waarin nadien grafische componenten geplaatst kunnen worden. Deze vensters kunnen ook in andere vensters geplaatst worden of boven andere vensters. Ze kunnen het

volledige scherm innemen of een vormgeving verkrijgen met behulp van een thema.

Activities worden in het besturingssysteem bijgehouden in een stapel. Wanneer een activity gestart wordt, dan wordt deze boven aan de stapel toegevoegd en als actieve activity ingesteld. De vorige actieve activity zal onder deze actieve blijven draaien en slechts naar de voorgrond komen als de nieuwe activity gestopt wordt.

Een activity heeft vier staten:

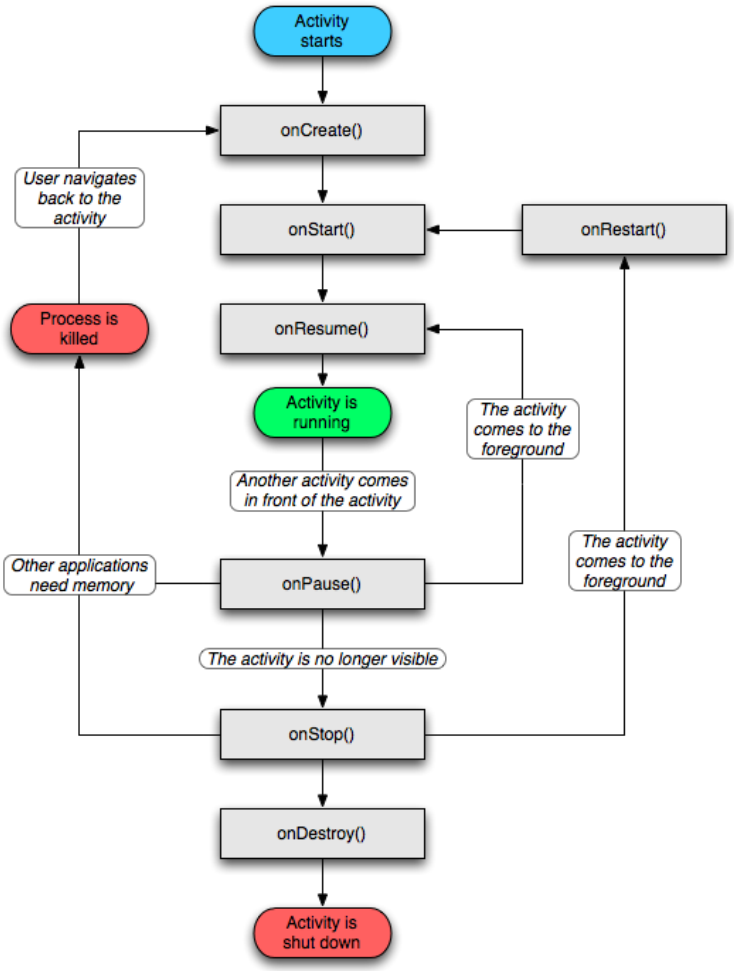
- Als de activity zichtbaar is, dan is deze **running**
- Als de activity zijn focus verloren is, bijvoorbeeld doordat een onvolledige of transparante activity zich voor de huidige bevindt, dan is deze **paused**[1].
- Als de activity onzichtbaar is, maar zich nog steeds in het geheugen bevindt, dan is deze **killed**.
- Wordt de activity uit het geheugen verwijderd dan is ze **shut down**.

Elke activity die zich niet in de **running** staat bevindt, kan gestopt worden door het besturingssysteem als deze meer geheugen nodig heeft. Dit kan op een mooie manier gebeuren door de activity hiervan op de hoogte te stellen en zichzelf te laten afsluiten, maar in het slechtste geval kan deze ook gewoon direct gestopt worden.

De volledige levenscyclus van een activity is weergegeven in figuur 3.1 op pagina 24 en alle methoden die gebruikt kunnen worden bij het ontwikkelen van een activity zijn te zien in codefragment 3.1. Deze methoden kan je zien als een soort event listeners. Als de activity een overgang maakt van een staat naar een andere dan worden deze methoden eerst aangeroepen. De staten waar een activity zich in kan bevinden worden in de figuur in kleur weergegeven. De rechthoekige blokken zijn callback methoden die je kan implementeren.

```
public class Nieuw extends Activity {
    protected void onCreate(Bundle savedInstanceState);
    protected void onStart();
    protected void onRestart();
```

[1]Deze staat is niet op de figuur weergegeven



Figuur 3.1: De belangrijkste statenpaden van een activity, de rechthoekige blokken zijn callback methoden, die je kunt implementeren. De gekleurde blokken zijn de staten waar een activity zich in kan bevinden [3].

```
    protected void onResume();
    protected void onPause();
    protected void onStop();
    protected void onDestroy();
}
```

Codefragment 3.1: Alle mogelijke hooks voor een activity.

De *onCreate()* methode wordt meestal gebruikt om informatie op te halen, eigenschappen in te stellen van visuele elementen, referenties op te halen naar

visuele elementen, de xml-layout in te stellen, de titel in te stellen, ... Deze methode wordt zo goed als altijd geïmplementeerd en bevat de meeste code.

De Bundle die deze optie meekrijgt is bij de eerste aanroep steeds leeg. Omdat een activity zeer regelmatig opnieuw gecreëerd wordt (bijvoorbeeld na een schermrotatie), kan de programmeur hier waarden in bewaren die gebruikt kunnen worden bij de volgende creatie. Doordat bepaalde waarden gecached kunnen worden in deze Bundle, wordt tijd bespaard bij het opnieuw aanmaken van de Activity omdat deze waarden niet opnieuw berekend of opgehaald moeten worden. Je kan de Bundle vullen door de *onSaveInstanceState(Bundle savedInstanceState)* methode te implementeren. Die methode wordt aangeroepen net voor de activity gesloten wordt. De waarden die je wil bewaren voor de volgende aanroep stop je dan in de Bundle.

De *onPause()* methode wordt voornamelijk gebruikt om wijzigingen op te slaan of interactie met de gebruiker stop te zetten.

De *onRestart()* methode kan opnieuw elementen activeren die interactie nodig hebben, of muziek laten spelen.

3.1.1.2 Intents

Dit zijn een soort events. Intents kunnen applicaties triggeren bij bepaalde gebeurtenissen en eventueel andere gebeurtenissen oproepen.

Een intent is een abstracte beschrijving van iets wat uitgevoerd moet worden en kan bijvoorbeeld gebruikt worden om een activity te starten, om de intent naar verschillende luisteraars te sturen, of om te communiceren met achtergrond services.

Een intent wordt gekenmerkt door twee primaire attributen:

- **action**: de actie die uitgevoerd moet worden, zoals bijvoorbeeld *ACTION_VIEW*, *ACTION_EDIT* of *ACTION_MAIN*.
- **data** een Uri die de data voorstelt waar de operatie op uitgevoerd moet worden.

Enkele voorbeelden van intents:

action - data in Uri formatie
Wat gebeurt er als deze intent uitgevoerd wordt?

ACTION_VIEW - content://contacts/people/1
Toon informatie over het contact met identificatie nummer 1.

ACTION_DIAL - content://contacts/people/1
Toon het bellerscherm met het hoofdtelefoonnummer van contact 1 ingevuld.

ACTION_VIEW - tel:123
Toon het bellerscherm met telefoonnummer 123 reeds ingevuld.

ACTION_EDIT - content://contacts/people/1
Toon het bellerscherm met het hoofdtelefoonnummer van contact 1 ingevuld.

ACTION_VIEW - content://contacts/people/
Toon de contactenlijst.

Naast deze primaire attributen zijn er ook een aantal secondaire, optionele attributen die men kan instellen in een intent:

- **category**: geeft extra informatie over de uit te voeren actie.
- **type**: definieert het MIME type van de data. Als deze optie niet opgegeven wordt, dan wordt het type afgeleid van de data zelf.
- **extras**: dit is een bundel met extra informatie, die gebruikt kan worden om extra informatie mee te geven. Starten we bijvoorbeeld een intent om een e-mail te versturen, dan kunnen extra headers voor de e-mail als extra's meegegeven worden.

Hoe worden intents in de applicatie gebruikt?

In codefragment 3.2 wordt gedemonstreerd hoe een intent in de applicatie gebruikt wordt om een nieuwe activity te starten en deze wat extra informatie mee te geven.

Zoals je kan zien wordt een Collection eerst omgezet naar een ArrayList, voor deze in de extra bundel geplaatst wordt. Een ArrayList is in tegenstelling tot een Collection namelijk wel serializable, wat nodig is om iets in een bundel te kunnen stoppen.

```
Collection<Result> listData;
Intent output = new Intent(this, Output.class);
output.putExtra("results", (ArrayList<Result>) listData);
startActivity(output);
```

Codefragment 3.2: Het gebruik van een intent om een nieuwe activity op te starten.

In codefragment 3.3 wordt een fragment van het AndroidManifest.xml-bestand getoond. Twee activities worden er gedefinieerd samen met hun bijhorende class bestand. De eerste activity bevat een extra eigenschap *android:name* die de naam aangeeft die in het programmamenu moet verschijnen. De *intent-filter* die er onder gedefinieerd staat duidt aan dat deze activity in het programmaoverzicht mag verschijnen van het besturingssysteem.

De tweede activity heeft een intent-filter die aangeeft dat die activity mag verschijnen in een lijst. Die lijst verschijnt elke keer als een programma een intent opstart met de action SEND en als data het MIME type *text/plain*. Waarvoor dit in de applicatie gebruikt kan worden staat beschreven in hoofdstuk 4.9.6.

```
<activity android:label="@string/app_name" android:name=".
    controller.Welcome">
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name=                                "
        android.intent.category.LAUNCHER" />
  </intent-filter>
</activity>

<activity android:windowSoftInputMode="stateAlwaysHidden"
    android:name=".controller.Analysis">
  <intent-filter>
    <action android:name="android.intent.action.SEND" />
    <data android:mimeType="text/plain" />
    <category android:name="android.intent.category.DEFAULT"
        />
  </intent-filter>
</activity>
```

Codefragment 3.3: Het gebruik van intents in het AndroidManifest.xml-bestand.

3.1.1.3 Content providers

Met een content provider bouw je een model rond je data. Alle applicaties of activities kunnen deze content providers vervolgens aanspreken om data op te halen. Zo bepaal je zelf hoe data opgehaald wordt. Als je data wil uitwisselen tussen verschillende programma's, dan kan dit enkel maar gebeuren met behulp van content providers.

In de ontwikkelde applicatie zijn geen content providers gebruikt.

3.1.1.4 Services

Services zijn gemaakt om meerdere programma's op hetzelfde moment te laten draaien, onafhankelijk van elke andere activiteit. Zo kan je er bijvoorbeeld voor zorgen dat muziek blijft spelen of dat periodiek een RSS feed opgehaald wordt.

In de ontwikkelde applicatie zijn geen services gebruikt.

3.1.2 Veiligheid en rechten

Elk programma dat op Android draait krijgt een eigen gebruikersnaam en groepsidentificatienummer. Ook systeemcomponenten worden op deze manier door verschillende virtuele identiteiten opgestart. Dit zorgt er voor dat programma's van elkaar en van het systeem geïsoleerd worden.

Een 'permission' systeem zorgt er voor dat programma's enkel die operaties kunnen uitvoeren waar vooraf toestemming voor verleend is. Deze toestemmingen worden gegeven via het *AndroidManifest.xml*-bestand van het programma. Het rechten deel van dit bestand bij de vooropgestelde applicatie ziet er bijvoorbeeld zo uit als in codefragment 3.4.

```
<uses-permission android:name="android.permission.
    ACCESS_NETWORK_STATE"></uses-permission>
<uses-permission android:name="android.permission.WAKE_LOCK"
    ></uses-permission>
<uses-permission android:name="android.permission.
    ACCESS_WIFI_STATE"></uses-permission>
<uses-permission android:name="android.permission.READ_SMS"
    ></uses-permission>
<uses-permission android:name="android.permission.
    READ_CONTACTS"></uses-permission>
<uses-permission android:name="com.fsck.k9.permission.
    READ_MESSAGES"></uses-permission>
<uses-permission android:name="android.permission.VIBRATE"
    ></uses-permission>
<uses-permission android:name="android.permission.INTERNET"
    ></uses-permission>
<uses-permission android:name="android.permission.CAMERA"></
    uses-permission>
<uses-permission android:name="android.permission.
    WRITE_EXTERNAL_STORAGE"></uses-permission>
<uses-feature android:name="android.hardware.camera.
    autofocus"/>
```

Codefragment 3.4: Alle permissions voor de communicatiehulp.

Bijvoorbeeld de lijn met **VIBRATE** zorgt er voor dat de applicatie het toestel kan laten vibreren als knoppen aangeduid worden. De applicatie K9 heeft een eigen recht gecreëerd en met de lijn waarin **READ_MESSAGES** staat kan de aanvraag gedaan worden om van dit recht gebruik te maken. In dit geval betekent dit dat de communicatiehulp mails kan ophalen uit de e-mailboxen die in K9 ingesteld staan.

De *uses-feature* lijn met **autofocus** was nodig om van de autofocus functie van de camera gebruikt te mogen maken. Deze functie wordt gebruikt bij het nemen van de foto bij de tekstherkenning.

Het idee achter deze permissions is dat geen enkele applicatie standaard toestemming heeft om operaties uit te voeren die eventueel andere applicaties, het besturingssysteem of de gebruiker zouden kunnen beïnvloeden. Dit houdt ook het lezen van private gegevens in zoals contacten, e-mails, lokale bestanden, het verkrijgen van netwerktoegang, etc.

Deze rechten moeten door de gebruiker eerst expliciet goedgekeurd worden bij de installatie en kunnen niet automatisch toegekend worden.

Bestanden die geschreven worden door een gebruikersprogramma en die niet expliciet de eigenschappen meegekregen hebben om voor iedereen lees- of schrijfbaar te zijn hebben standaard de gebruikersidentificatie van de gebruiker en zijn dus ontoegankelijk voor andere programma's.

3.1.2.1 Eigen toegangsrechten declareren

Het is ook mogelijk om een applicatie zelf nieuwe rechten te laten declareren:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/
    android"
     package="com.me.app.myapp" >
     <permission android:name="com.me.app.myapp.permission.
         DEADLY_ACTIVITY"
          android:label="@string/permlab_deadlyActivity"
          android:description="@string/permdesc_deadlyActivity
              "
          android:permissionGroup="android.permission-group.
              COST_MONEY"
          android:protectionLevel="dangerous" />
     ...
</manifest>
```

Codefragment 3.5: De applicatie zelf rechten laten declareren.

De verplichte **protectionLevel** eigenschap vertelt het systeem op welke manier de gebruiker geïnformeerd moet worden als een applicatie die rechten wil opnemen.

De **permissionGroup** eigenschap is optioneel en wordt enkel gebruikt door het besturingssysteem om de rechten visueel te groeperen. De voorkeur gaat er naar uit om een standaard groep te gebruiken die je kan terugvinden in *android.Manifest.permission_group*, maar het is ook perfect mogelijk om er zelf een te definiëren.

De **label** en **description** eigenschappen moeten beide ingevuld worden en bevatten de teksten die getoond worden aan de gebruiker wanneer ze de rechten van een applicatie bekijken. Het label moet een korte beschrijving zijn van de functionaliteit die het recht beschrijft. De beschrijving mag langer zijn en dieper over het recht ingaan.

3.2 Het gebruik van de savedInstanceState bundel

Bij het wisselen van modus wordt de huidige activity afgebroken en een nieuwe opgezet. Zonder extra aanpassingen zorgt dit er voor dat berekeningen opnieuw gebeuren en informatie opnieuw opgehaald wordt. Om dit te vermijden kan bij het sluiten van een activity informatie opgeslaan worden *savedInstanceState*-bundel.

Om informatie op te slaan in de bundel moet de activity de volgende methode overschrijven:

```
@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
  super.onSaveInstanceState(savedInstanceState);
};
```

Informatie kan in de bundel gestopt worden door er methoden op aan te roepen zoals *putInt()* en *putSerializable()*. De eerste methode stopt een integer in de bundel en de tweede kan een serialiseerbare klasse in de bundel stoppen. Elk element wordt samen met een sleutel opgeslaan, die sleutel wordt dan gebruikt om de waarden terug op te halen. Dat ophalen gebeurt in de *onCreate()* functie van de activity In codefragmenten 3.6 en 3.7 wordt getoond hoe dit kan.

```
@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
  savedInstanceState.putSerializable("results", results);
  savedInstanceState.putInt("currentPosition",
     currentPosition);
  super.onSaveInstanceState(savedInstanceState);
}
```

Codefragment 3.6: Het vullen van een savedInstanceState bundel.

```
@Override
public void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  // Terughalen van de vorige staat
  if (savedInstanceState != null) {
    results = (ArrayList<Result>) savedInstanceState.
       getSerializable("results");
    currentPosition = savedInstanceState
       .getInt("currentPosition");
  }
}
```

Codefragment 3.7: Het uitlezen van een savedInstanceState bundel.

3.3 Handige Android-bibliotheken die gebruikt zijn in de applicatie

3.3.0.2 De android.net.Uri.Builder klasse

In Android is een handige android.net.Uri.Builder klasse meegeleverd. Deze bouwt een internet adres op uit verschillende delen: de server, de mappen en pagina's en de parameters. Je kan bij het aanmaken van de builder een adres meegeven:

```
urlBuilder = Uri.parse(getBasicURL()).buildUpon();
```

Vervolgens kan je op een object georiënteerde manier eenvoudig extra parameters toevoegen.

```
urlBuilder.appendQueryParameter("method", "flickr.photos.
   search")
          .appendQueryParameter("api_key", "17
             e16818b422b2bcfa122b85461c9d22")
          .appendQueryParameter("format", "json")
```

```
        .appendQueryParameter("sort", "relevance")
        .appendQueryParameter("media", "photos");
```

Codefragment 3.8: Het aanvullen van een internetadres met extra parameters.

Deze parameters vormen samen de inputaanvraag van een API aanvraag. Die aanvraag wordt met de REST methode uitgevoerd.

3.3.1 De java.util.zip.ZipFile klasse

De informatie over de woordsoorten is opgeslagen in een SQLite-databank 3.8.5. Om toegang te krijgen tot de database is een klasse geschreven die afgeleid is van de android.database.sqlite.SQLiteOpenHelper klasse, die reeds functionaliteit bevat om een dergelijke databank te gebruiken.

Om de database niet op het toestel te moeten aanmaken wordt deze met de applicatie meegeleverd. Een applicatie is op Android een ZIP-container (met extentie .apk) met daarin alle benodigde Java-klassen, afbeeldingen en andere bestanden. De database wordt daar dus ook in meegeleverd. Omdat deze niet rechtstreeks gelezen kan worden vanuit de applicatie, dient deze eerst naar de telefoon gekopieerd te worden. De databank is echter 25Mb groot en dit overschrijdt de voorschriften van Android. Volgens die voorschriften mogen er namelijk zo geen grote resource-bestanden gelezen worden. Hierdoor kon deze dus niet rechtstreeks uit de *assets* map van het applicatie ZIP-bestand naar de telefoon gekopieerd worden.

Via een omweg wordt het *.apk*-bestand van de mobiele applicatie (wat eigenlijk gewoon een ZIP-container is) gelezen. Hier wordt de database vervolgens in opgezocht en wordt een stream geopend naar dat bestand waarna het naar het interne geheugen van de telefoon gekopieerd kan worden. Dit gebeurt slechts eenmalig bij elke eerste opstart van het programma.

Uiteindelijk hebben we een SQLite-database op het telefoongeheugen die snel uitgelezen kan worden met behulp van SQL-queries.

3.4 Optical character recognition (OCR) software voor Android

Optische tekstherkenning is een transformatie waarbij de tekens in een afbeelding automatisch herkend worden en omgezet worden naar leesbare tekst. Zodoende moet de tekst niet meer manueel ingevoerd worden.

Deze technologie is in de voorgestelde applicatie ingebouwd. Zo is het mogelijk om een foto te nemen van een deel van een folder of een informatiebord, waarna de tekst opgehaald kan worden.

Er is onderzoek gebeurd naar verschillende mogelijkheden om tekstherkenning in de applicatie in te bouwen. Er zijn bedrijven die interfaces aanbieden om software te gebruiken die de bedrijven zelf hosten of er zijn bibliotheken die al dan niet tegen betaling met de applicatie meegeleverd kunnen worden. In de volgende secties worden enkele mogelijkheden besproken.

3.4.1 De theorie

In een eerste fase bekijkt de software de indeling van de foto. Er wordt bepaald waar de tekst staat, waar afbeeldingen staan en wat eigenlijk als vlekken kan aangeduid worden. Ook kolommen worden gedetecteerd. Als de tekst scheef staat wordt deze eerst recht gezet.

De software deelt de tekst vervolgens op in steeds kleinere onderdelen: eerst in zinnen, dan in losse woorden en ten slotte in letters. Die letters worden dan vergeleken met letters die de software kent. De software heeft een databank met de vormen van letters in verschillende soorten lettertypen.

Dat vergelijken kan gebeuren door een matrix op te bouwen met witte en zwarte vakjes. De vakjes worden dan vergeleken met de matrix van vakjes van de tekens in de databank. Hoe meer vakjes overeen komen hoe groter de kans dat we dezelfde letter hebben.

Een minder voorkomende techniek is Intelligent Character Recognition (ICR). De software gaat dan op zoek naar enkele eigenschappen in een teken zoals open en gesloten vormen, diagonale lijnen en kruisende lijnen. Wanneer een minder voorkomend lettertype gebruikt wordt werkt deze methode beter dan het vergelijken van matrixen.

Vaak maakt de software ook gebruik van neurale netwerken. Tekst wordt dan herhaaldelijk aan de software aangeboden. Het resultaat daarvan wordt dan vergeleken met de gewenste output en de fouten worden opgezocht. Die fouten worden dan gebruikt om het netwerk te trainen en sommige waarden te optimaliseren. Na elke ronde zou de software zo beter moeten worden.

Veel letters lijken op elkaar, de kans is groot dat foute detecties gebeuren. Twee technieken proberen dit op te lossen: training en taaltechnologie.

3.4.1.1 Training

Als teksten met een onbekend lettertype ingegeven worden ligt de kwaliteit van de herkenning bijzonder laag. Daarom wordt de software dan beter getraind op de nieuwe letters.

De training vindt plaats vóór de tekstherkenning.

3.4.1.2 Taaltechnologie

Verbeteringen kunnen ook gebeuren na de tekstherkenning. Dit wordt post-processing genoemd. De software probeert na de herkenning uit te maken in welke taal de tekst geschreven is. Een woordenboek van die taal wordt dan ingelezen. Woorden die niet in het woordenboek terug te vinden zijn worden vergeleken met woorden die er sterk op lijken. Eventueel wordt het woord dan vervangen.

Eventueel kan de software ook zelflerend zijn. Worden bijvoorbeeld twee letters regelmatig verwisseld, dan kan deze eerst opzoeken of hij met één van de twee letters een bestaand woord kan maken. Herkenningen kunnen eventueel ook beter gebeuren als de software samengestelde woorden kan opsplitsen.

Zeer geavanceerde software kan het juiste woord ook afleiden van de context.

3.4.2 ABBYY

ABBYY, een bekende leverancier van scansoftware voor de desktop, heeft een Mobile OCR Engine uitgebracht voor Android. De tekstherkenning heeft een ondersteuning voor 58 talen, waaronder het Nederlands. Je kan gebruik maken van C, C++ of C# en er is een gratis testversie. Die gratis testversie is volledig functioneel, maar werkt maar voor een gelimiteerde periode [8]. Aangezien een licentie voor het gebruik van deze bibliotheek niet goedkoop is, hebben we besloten geen gebruik te maken van de software van ABBYY.

3.4.3 Google Goggles

In de zoekmachine van Google kan je een zoekopdracht starten door in te typen wat je zocht of door in te spreken wat je wil zoeken. Met Google Goggles kan je ook naar informatie over een object zoeken door een foto te nemen van een object. Of met behulp van deze software kan je een visitekaartje inscannen en er de tekst van herkennen. Volgens een artikel [22] zou Google

zijn Goggles-applicatie willen ombouwen naar een API. Maar voorlopig is die nog niet vrijgegeven [14].

In een tussentijdse versie van het programma was het mogelijk om toch de servers van Google te gebruiken om de tekstherkenning door te voeren.
Met behulp van het analysesoftware [20] en op internet gevonden informatie [18] is de bytestream bestudeerd die het Goggle-programma uitstuurt. De herkenning gebeurt namelijk op externe servers en niet op het mobiele toestel zelf. Zo is een programmacode opgebouwd die zich voordeed als het Google Goggles programma.

Het spreekt voor zich dat deze methode enkel maar gebruikt kan worden in een testomgeving en niet als de communicatiehulp in grote hoeveelheden gedistribueerd zou worden. Er is van het bedrijf ook geen toestemming gekregen om deze methode te mogen gebruiken.

3.4.3.1 Verbinden met de Google Goggles Service

Om een aanvraag te richten naar de Goggles servers moet eerst een authentificatie gebeuren. Hiervoor moet de programmacode een **CSSID** aanmaken. Deze identificatie is een willekeurig gegenereerde string van 16 hexadecimale waarden lang en blijft geldig voor een beperkt aantal aanvragen. Een voorbeeld **CSSID** is bijvoorbeeld 43CBE97C0C31FE5A.

Als het **CSSID** aangemaakt was, moest een POST-aanvraag verzonden worden naar:

```
http://www.google.com/goggles/container_proto?
   cssid=[aangemaakt CSSID]
```

Deze aanvraag moest de volgende headers bevatten:

Content-Type	application/x-protobuffer
Pragma	no-cache

En de volgende body (als byte verzameling):

```
22, 00, 62, 3C, 0A, 13, 22, 02, 65, 6E, BA, D3, F0, 3B, 0A,
08, 01, 10, 01, 28, 01, 30, 00, 38, 01, 12, 1D, 0A, 09, 69,
50, 68, 6F, 6E, 65, 20, 4F, 53, 12, 03, 34, 2E, 31, 1A, 00,
22, 09, 69, 50, 68, 6F, 6E, 65, 33, 47, 53, 1A, 02, 08, 02,
22, 02, 08, 01
```

Als deze aanvraag succesvol verlopen is, dan verkrijg je een antwoord met '*200 OK*'. De CSSID kan nu gebruikt worden toekomstige aanvragen.

Voor de aanvraag doorgestuurd kan worden moeten eerst enkele berekeningen gebeuren. Eerst worden drie waarden berekend (x is de grootte van de afbeelding):

```
a = x + 401
b = x + 14
c = x + 10
```

Deze drie waarden moeten omgezet worden in varints [51]. Deze omzetting kan gebeuren via onderstaand algoritme.

1. Het nummer moet naar een binaire voorstelling omgezet worden en in groepen van 7 bits geplaatst worden.

   ```
   0000001 1011010 1010001
   ```

2. De groepen moeten daarna omgekeerd geplaatst worden.

   ```
   1010001 1011010 0000001
   ```

3. Zet vervolgens elke meest beduidende bit op 1, behalve bij de meest rechtse byte, daar zet je hem op 0.

   ```
   11010001 11011010 00000001
   ```

Opnieuw moet nu een POST aanvraag gebeuren, waar opnieuw de twee aangepaste headers gezet worden. De aanvraag gebeurt naar het volgende adres:

```
http://www.google.com/goggles/container_proto?
   cssid=[aangemaakt CSSID]
```

En de volgende body:

```
0A a 0A b 0A c 0A x [afbeelding als byte verzameling]
```

De waarden a tot c worden hierin vervangen door de berekende varints, x door een varint voorstelling van de afbeeldingsgrootte en *[afbeelding als byte verzameling]* door de afbeelding.

Het antwoord dat je terugkrijgt, is opgemaakt in het ProtoBuffer-formaat. Hierin ontdek je verschillende internetadressen, maar ook de letterlijke tekst is terug te vinden [18].

Op 20 april 2011 heeft Google helaas het protocol aangepast waardoor deze methode niet meer werkte. Daarom is in de uiteindelijke versie van het programma de API van WiseTrend gebruikt om de tekstherkenning uit te voeren.

3.4.4 IQ Engines

IQ Engines [16] heeft ook een open Developer API voor mobiele applicatie ontwikkelaars. Ze hebben een engine die afbeeldingen zou moeten herkennen. Het werkt zeer eenvoudig: je stuurt een afbeelding via HTTP post en je krijgt een JSON/XML terug met het resultaat.

De prijsstructuur is heel transparant: de eerste 100 afbeeldingen per dag zijn gratis en erna betaal je 7 cent per afbeelding. Maar ze geven geen OCR terug, maar een label wat de foto beschrijft. Je stuurt bijvoorbeeld een foto van een fles ketchup en je verkijgt als resultaat: "Fles ketchup van het merk Heinz".

Voor het project konden we helaas geen gebruik maken van deze engine omdat deze niet aan tekstherkenning doet. De engine zal enkel afbeeldingen herkennen en vergelijken met andere afbeeldingen in zijn database. Het label dat bij die andere afbeelding staat wordt dan teruggegeven.

3.4.5 Mezzofanti

Mezzofanti is een opensourceprogramma voor Android die aan tekenherkenning doet. Die teksten die het programma gevonden heeft, kunnen vervolgens eenvoudig vertaald, opgezocht of op internet gezocht worden. De OCR-engine is gebaseerd op Tesseract versie 2.03. Deze engine is geschreven in C++ en de interface is gewone Java code. JNI wordt gebruikt als interface tussen deze twee programmadelen.

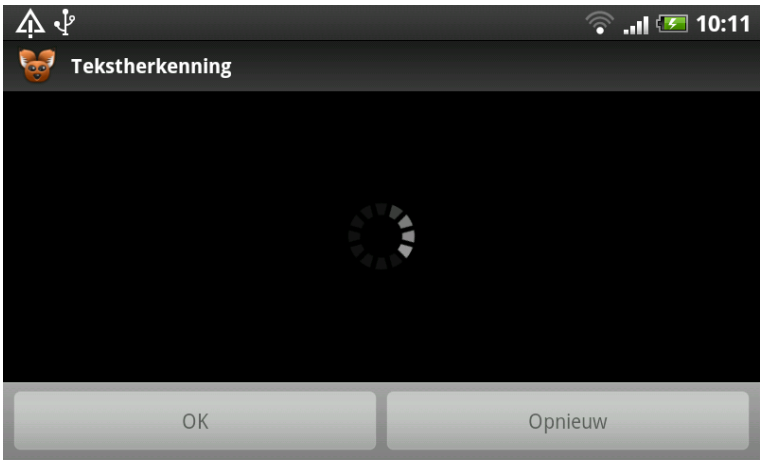
3.4.6 Gebruikte technologie

In eerste instantie was het de bedoeling om de broncode van het Mezzofanti-project 3.4.5 als startpunt te nemen. Deze code is bestudeerd en uitgetest, maar we zijn tot de conclusie gekomen dat deze code als startpunt nemen te

veel tijd zou innemen. De resultaten die dit gaf waren helaas ook niet van een uitmuntende kwaliteit.

Omdat dit onderdeel niet de hoofdmoot in de masterproef is, is er voor gekozen om de tekstherkenning op een andere manier uit te werken. In een ander project met een smallere focus en meer ontwikkelingstijd zou dit project evenwel als startpunt gekozen kunnen worden en de kwaliteit ervan verbeterd kunnen worden.

3.4.7 De OCR API van WiseTrend



Figuur 3.2: De applicatie is aan het wachten tot Wisetrend de afbeelding verwerkt heeft.

Eerdere pogingen om een goede tekstherkenningsbibliotheek te vinden liepen spaak. Daarom is er uiteindelijk voor gekozen de OCR API van Wisetrend [58] te gebruiken. Nadeel van deze bibliotheek is dat de verwerking van afbeeldingen niet gratis gebeurt. Maar er kon begonnen worden met enkele gratis verwerkingen en na contact met het bedrijf zijn nog extra credits verkregen.

De WiseTrend API laat toe om afbeeldingen in allerhande formaten op te sturen (pdf, tif, png, jpg, bmp, pcx of dcx afbeeldingen). Resultaten kunnen aangeboden worden in txt, rdf, rtf, Word, Excel, xml of csv-formaat. De herkenning kan ook gebeuren in verschillende talen.

3.4.7.1 Het gebruik van de API

Om de API te kunnen gebruiken moet een account aangemaakt worden[2], zo verkrijgt met een API sleutel.

Het proces verloopt in drie stappen:

1. Een nieuwe job wordt aangemaakt.
2. De status wordt gemonitord door ofwel manueel periodiek de status op te vragen of door automatisch op de hoogte gebracht te worden.
3. De resultaten kunnen opgehaald worden.

In de communicatiehulp is hier nog een stap voor geplaatst. Als een job aangemaakt wordt, dan moet een internetlocatie naar de afbeelding opgegeven worden. Daarom wordt de afbeelding eerst online geplaatst bij Webservius. Webservius is het bedrijf wat instaat voor de API-infrastructuur van onder andere WiseTrend.

Codefragment 3.9 toont hoe een verbinding opgezet wordt, een afbeelding doorgestuurd wordt en de locatie uitgelezen wordt.

```
URL url = new URL(postaddress);
HttpURLConnection conn = (HttpURLConnection) url.
    openConnection();
conn.setRequestProperty("Content-Type", "image/jpeg");
conn.setDoOutput(true);
OutputStream wr = conn.getOutputStream();
wr.write(image);
wr.flush();
BufferedReader rd = new BufferedReader(new InputStreamReader
    (conn.getInputStream()));
String result = rd.readLine();
```

Codefragment 3.9: De internetlocatie van een afbeelding wordt verkregen na het uploaden van een afbeelding.

Nu de afbeelding online staat kan een job aangemaakt worden door een xml-bestand via HTTP POST te versturen. Zo'n xml ziet er zo uit:

```
<Job>
 <InputURL>[locatie van de afbeelding]</InputURL>
```

[2]Een account kan men aanmaken op:
http://www.webservius.com/cons/subscribe.aspx?p=wisetrend&s=wiseocr

```
<InputType>[formaat van de afbeelding (PDF/TIF/PNG/JPG/enz
    .)]</InputType> <!-- Optioneel -->
<NotifyURL>[job status meldingsadres]</NotifyURL>
    <!-- Optioneel -->
<CleanupSettings>[instellingen om de afbeelding te
    verhelderen]</CleanupSettings> <!-- Optioneel -->
<OCRSettings>[OCR instellingen]</OCRSettings>
    <!-- Optioneel -->
<OutputSettings>[Instellingen voor de resultaten]</
    OutputSettings> <!-- Optioneel -->
</Job>
```

Bij *NotifyURL* kan een internetadres ingevuld worden wat geopend moet worden als de taak klaar is. Dan is het niet meer nodig om manueel de status van een taak te blijven opvragen tot deze wijzigt. Bij de communicatiehulp is dit niet gebruikt.

Een aantal instellingen kunnen bij *CleanupSettings* ingesteld worden:

- Door **Deskew** (true) op *true* in te stellen wordt de afbeelding automatisch gedraaid voor ze verwerkt wordt.
- De **RemoveGarbage** (true) functie zorgt er voor dat kleine puntjes eerst gewist worden.
- **RemoveTexture** (true) kan ingeschakeld worden als eerst ruis verwijderd moet worden.
- Je kan ook de richting specificeren waarnaar het document moet gedraaid worden met de optie: **RotationType** (Automatic). De mogelijkheden zijn: *NoRotation*, *Automatic*, *Clockwise*, *Counterclockwise* of *Upsidedown*.

Het optimaliseren van de tekstherkenning kan door het aanpassen van *OCR-Settings*:

- **PrintType** (Print) bepaalt wat voor soort tekst doorgestuurd wordt: *Print*, *Typewriter*, *DotMatrix*, *OCR_A*, *OCR_B* of *MICR_E13B*. wij gebruiken de combinatie van *Print*, *OCR_A* en *OCR_B*.
- De taal kan je met de **OCRLanguage** (English) optie meegeven. Wij geven *Dutch* mee[3].

[3]Dit staat foutief in de documentatie van de API. Dit werd duidelijk na een contact met het bedrijf.

- Eventueel kan men gebruik maken van de **SpeedOCR** (false) optie. De herkenning gebeurt dan tot 2,5 maal sneller, maar kan tot 2 maal zoveel fouten leiden.
- **AnalysisMode** (MixedDocument) geeft het type documenten weer.
  - De modus'**MixedDocument**' is handig als je zowel naar afbeeldingen als naar tekst wil laten zoeken in een document.
  - De modus '**TextIndexing**' wordt gebruikt om op zoek te gaan naar tekst op het document en in de afbeeldingen op het document. De logische volgorde van het document blijft bewaard.
  - De modus '**TextAggressive**' werkt tamelijk aggresief en gaat alle tekst detecteren die het kan vinden. Handig voor lage kwaliteitsdocumenten met veel kleine text delen en veel ruis.
  - De modus '**BarcodesOnly**' gaat enkel barcodes ontcijferen.

  Het verschil zit er in hoe je wil dat afbeeldingen op documenten behandeld worden. In de *TextAggressive* modus zal een verkeersbord in de achtergrond bijvoorbeeld als potentiële tekst aanzien worden. In *MixedDocument* modus zal datzelfde bord als een afbeelding gezien worden en in *TextIndexing* zal het bord nog steeds een afbeelding zijn in het resultaat, maar alle tekst die er in herkend is zal nog steeds opzoekbaar zijn.
- De optie **LookForBarcodes** (true) spreekt voor zich.

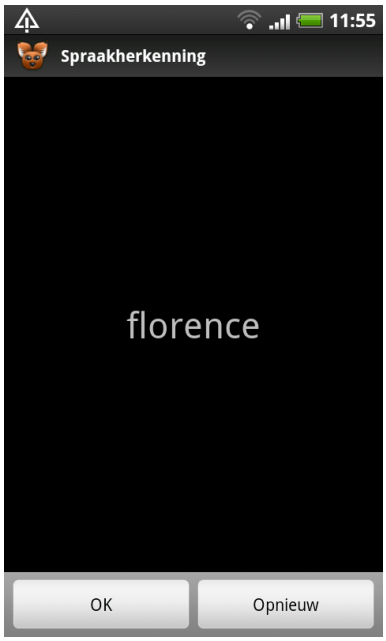
Bij de *OutputSettings* kan met de **ExportFormat** (Text;PDF) optie ingesteld worden welk resultaat we willen terugkrijgen. De communicatiehulp neemt genoegen met een Text bestand.

Als de job aangemaakt is krijg je een xml-resultaat terug dat er zo uitziet:

```
<JobStatus>
 <JobURL>[internetadres waar de status van de job op
    gevonden kan worden]</JobURL>
 <Status>Submitted</Status>
</JobStatus>
```

Op de *jobURL* kan de status van de job opgezocht worden. Is de job gelukt, dan kunnen op dat adres ook de locaties van de resultaten gevonden worden.

Figuur 3.3: Spraakherkenning in de applicatie.



Figuur 3.4: Resultaat van de spraakherkenning.

3.5 Spraakherkenning voor Android

Recent [24] heeft Google Nederlandstalige spraakbesturing aan Android toegevoegd.

De spraakherkenning werkt online via een server. Er wordt een soort hash over je audiobestand getrokken en die wordt dan naar de servers van Google gestuurd. Dit heeft het voordeel dat de tekstherkenning steeds beter zal kunnen werken, maar er is natuurlijk een internetverbinding voor nodig. Voorlopig werkt dit nog het beste voor korte zoekwoorden en minder voor een gedicteerde lange tekst of langere zin. Ook werken langere woorden beter dan korte woorden, op voorwaarde dat je ze duidelijk inspreekt.

In de applicatie is rond deze bibliotheek een ingavescherm opgebouwd met spraakherkenning.

3.6 Bronnen

De afbeeldingen en filmpjes die in de applicatie teruggegeven worden zijn afkomstig van verschillende bronnen. In de applicatie is een structuur opgezet waardoor gemakkelijk bronnen toegevoegd kunnen worden. Er zijn reeds vier verschillende bronnen uitgewerkt. De API's daarvan staan in volgende hoofdstukken uitgelegd.

3.6.1 Google Image Search API

De Google Image Search API is zeer eenvoudig op te roepen door middel van een HTTP-request. Het resultaat is een JSON-object met een selectie van de resultaten. Dit object heeft een aantal interessante eigenschappen:

- **cursor**: een optioneel attribuut dat toegevoegd wordt als een zoekaanvraag succesvol verlopen is en waarmee extra informatie opgevraagd kan worden. Het object bevat de pagina's met resultaten, een ruwe schatting van het aantal resultaten, de huidige pagina-index en een internetadres om meer resultaten op te halen. Er kunnen in totaal 64 afbeeldingen opgehaald worden, waarvan er 8 per pagina komen.
- **results**: bevat de resultaten:

height	de hoogte van de afbeelding
tbHeight	de hoogte van de thumbnail
tbUrl	adres van de thumbnail
tbWidth	breedte van de thumbnail
titleNoFormatting	titel van de afbeelding zonder HTML-codes
unescapedUrl	internetadres van de bronafbeelding
width	breedte van de afbeelding

Bij de aanvraag kunnen ook een aantal opties ingesteld worden die nuttig zijn in het hulpprogramma:

- **asfiletype**: wordt gebruikt om restricties op te leggen aan het bestandsformaat.
- **as_rights**: hiermee kunnen restricties gelegd worden op de rechten van de ontvangen afbeeldingen.
- **as_sitesearch**: wordt gebruikt om beperkingen op te leggen tot een bepaald sitedomein.

- **imgsz**: beperken van de afbeeldingen tot een bepaalde grootte.
- **imgtype**: hiermee beperk je de afbeeldingen tot een bepaald type, zoals gezichten, foto's, clipart of tekeningen.
- **rsz**: aantal terug te geven argumenten.
- **safe**: bepaalt de filter mode die gebruikt wordt.
- **start**: index van de eerste afbeelding die moet teruggegeven worden. Er kunnen namelijk slechts 8 afbeeldingen per keer teruggegeven worden. Wil je dus 24 afbeeldingen dan moeten er dus 3 aanvragen gebeuren waar je deze waarde respectievelijk instelt op 0, 8 en 16.

3.6.2 De API Flickr

De API van Flickr bevat een ruim aanbod aan technologieën die gebruikt kunnen worden om zoekopdrachten uit te voeren op hun database. Er is voor gekozen om de afbeeldingen op te halen via REST en een JSON-antwoord op te vragen en te verwerken.

Een typische REST aanvraag bij Flickr:

```
http://api.flickr.com/services/rest/?
    method=flickr.test.echo&name=value
```

Flickr stelt zo goed als zijn volledige database ter beschikking. Maar het interessantste object om aanvragen aan te doen is natuurlijk het *photos*-object. Het *Flickr.photos*-object bevat een zoekmethode ("search") die gebruikt kan worden om foto's op te halen. Er is slechts één verplicht veld: *api_key*, maar ook andere optionele velden zijn interessant om te gebruiken in de applicatie:

- **tags**: een lijst met tags, gescheiden door een komma. Vul je deze parameter in, dan verkrijg je enkel foto's die een van deze tags bevat. Ook kan je een minteken voor een tag zetten om er voor te zorgen dat je enkel foto's terugkrijgt die deze tag zeker niet bevatten.
- **tag_mode**: (*een van de tags*) met deze instelling kan je kiezen of een foto alle tags uit het *tags* veld moet bevatten, of slechts een van de tags.
- **text**: foto's die deze tekst bevatten in hun titel, beschrijving of tags worden teruggegeven. Ook hier kan het minteken gebruikt worden om resultaten terug te geven die een woord *niet* bevatten.

- **license**: licenties die de resultaten moeten hebben.
- **sort**: (*date-posted-desc*) in welke volgorde moeten foto's teruggegeven worden? De mogelijkheden zijn: date-posted-asc, date-posted-desc, date-taken-asc, date-taken-desc, interestingness-desc, interestingness-asc en relevance. De standaard instelling *date-posted-desc*, wordt beter vervangen door *relevance*.
- **media**: (*all*) drie modes kunnen hier opgegeven worden: *all*, *photos*, *videos*.
- **per_page**: (*100*) aantal foto's om terug te geven per pagina.
- **page**: (*0*) pagina die opgevraagd moet worden.

Elke foto in het resultaat heeft een aantal interessante eigenschappen:

- **id**: identificatie van de foto.
- **secret**: token nodig voor het ophalen van de foto.
- **server**: de server waar de foto staat.
- **title**: titel van de foto.

Met behulp van deze informatie kunnen we de URL opbouwen waar de foto te vinden is:

```
http://farm{farm-id}.static.flickr.com/{server}/
   {id}_{secret}_[mstzb].jpg
```

Uit experimenten is gebleken dat je *farm-id* voor elke foto door *1* kan vervangen, deze informatie krijg je namelijk niet terug in het resultaat. *server*, *id* en *secret* vervang je door de respectievelijke waarden die je terug gekregen hebt en *[mstzb]* vervang je door een van de letters in de array. De gekozen letter bepaalt het formaat van foto dat je wil ophalen:

- **s**: 75x75 pixels
- **t**: langste zijde is maximum 100 pixels lang
- **m**: langste zijde is maximum 240 pixels lang

- **z**: langste zijde is maximum 640 pixels lang
- **b**: langste zijde is maximum 1024 pixels lang

Een foto kan bijvoorbeeld te vinden zijn op dit adres:

```
http://farm1.static.flickr.com/2/3_secret_m.jpg
```

3.6.3 De data API van YouTube

Omdat deze API zo uitgebreid is worden hier enkel de opties besproken die bij de communicatiehulp gebruikt zijn.

Het standaard internet adres om aanvragen aan te richten is dit:

```
http://gdata.youtube.com/feeds/projection/videos?v=2
```

De **v** parameter staat voor de versie. Dit adres is voor de communicatiehulp uitgebreid met een aantal extra parameters:

- De **alt** parameter (atom) kan het outputformaat aanpassen. Mogelijke waarden zijn: *atom*, *rss*, *json*, *json-in-script* of *jsonc*. De communicatiehulp werkt met json.
- Met de **format** parameter kan aangegeven worden welk formaat van video het resultaat moet bevatten. Door hier 5 in te stellen worden enkel video's teruggegeven die met de *embedded player* kunnen weergegeven worden. De video's worden namelijk weergegeven door over het volledige scherm een webpagina weer te geven met de embedded player in.
- Door de **orderby** (relevance) parameter mee te geven kan de volgorde van de resultaten gewijzigd worden. Mogelijke waarden: *relevance*, *published*, *viewCount*, *rating*. Wij gebruiker *relevance*.
- **max-results** wordt nog ingevuld met het maximumaantal resultaten, wat afhangt van de instellingen van de gebruiker.
- De zoektermen die we in de resultaten willen terugzien worden tenslotte in de **q**-parameter gezet.

Een *feed*-object wordt teruggestuurd met daarin een eigenschap *entry* die een lijst bevat met alle filmpjes die aan de zoekopdracht voldoen. Uit elk resultaat halen we een aantal interessante eigenschappen:

- De *$t* eigenschap van de *title*-eigenschap bevat de titel van het filmpje.
- In het media$group attribuut kunnen we op zoek naar de media$thumbnail array. In die lijst nemen we dan de url eigenschap van het element waar *yt$name* de waarde *hqdefault* heeft. Dat is dan het adres van de thumbnail voor het filmpje.
- Door *$t* van de *id* eigenschap uit te lezen, en de eerste 27 tekens weg te gooien, bekomen we het id van het filmpje.

Het filmpje kunnen we in een ingebouwde browser weergeven door in de volgende HTML code *id* te vervangen door de identificatie van het filmpje:

```
<html>
 <head>
  <style type="text/css">
   body,html{
    margin:0; overflow:hidden
   }
  </style>
 </head>
 <body>
 <embed
  src="http://www.youtube.com/v/{id}?version=3&feature=
     player_embedded"
  type="application/x-shockwave-flash"
  allowfullscreen="true"
  width="100%"
  height="100%">
 </body>
</html>
```

3.6.4 Vimeo Advanced API

Elke Vimeo API call is geauthenticeerd met OAuth, voor meer informatie daarover verwijzen we naar hoofdstuk 3.10. Enkel de zoekmethode die in de communicatiehulp gebruikt is, wordt besproken.

Het standaard internetadres wat gebruikt kan worden om geavanceerde aanvragen naar toe te sturen is het volgende:

```
http://vimeo.com/api/rest/v2/
```

De methode die gebruikt is heet *vimeo.videos.search*. Deze methode heeft ook een aantal extra parameters die we gebruiken. In de applicatie is het basisadres dus aangevuld met de volgende parameters:

- **format**: (xml) deze algemene parameter bepaalt in welk formaat de output mag teruggestuurd worden. Hier is *json* gebruikt, maar ook andere mogelijkheden konden gekozen worden: *php* of *jsonp*.
- **sort**: mogelijke waarde voor deze optie zijn: *relevant*, *newest*, *oldest*, *most_played*, *most_commented*, or *most_liked*.
- **method**: deze parameter geeft de methode op die we willen aanroepen. In dit geval is de vimeo.videos.search methode gebruikt.
- **full_response**: voor de communicatiehulp was het nodig deze waarde op *waar* te zetten. Anders stonden er niet genoeg details in het antwoord.
- **per_page**: bepaalt hoeveel video's maximaal mogen gezocht worden.
- **query**: in deze parameter kunnen de zoekwoorden gespecificeerd worden.

Er komt een *video* object terug waar de *videos* array in uitgelezen wordt. Elk object in die array heeft een aantal interessante eigenschappen:

- In de *urls* eigenschap zit een array *url*. In eerst instantie wordt in de applicatie onderzocht of daar een url bijzit met *type* gelijk aan *mobile*. Is dit niet het geval, dan kan dat filmpje niet mobiel weergegeven worden en is het dus niet bruikbaar.
- In de *title*-eigenschappen vinden we de titel terug.
- De identificatie is gemakkelijk terug te vinden in de *id*-eigenschap.
- Voor de kleine voorweergaveafbeelding gaan we in de eigenschap *thumbnails* op zoek naar een array genaamd *thumbnail*. De applicatie loopt dan alle thumbnails af tot er een gevonden kan worden waar *width* ingesteld staat op *640*.

3.7 Ophalen en bewaren van afbeeldingen

In eerste instantie werd de stream van de afbeelding rechtstreeks ingeladen en in een Android Drawable gestopt. Bij het heen en weer navigeren in de resultaten dienden de afbeeldingen bijgevolg wel telkens opnieuw binnengehaald te worden. Dit gaf een zichtbare vertraging tot gevolg en onnodig dataverkeer.

Uiteindelijk is er voor gekozen om de afbeeldingen eerst te downloaden en op te slaan in de cache die de applicatie krijgt van Android op het telefoongeheugen. Bij een volgende aanvraag wordt dan gekeken of de afbeeldingen zich nog in de cache bevindt en vervolgens daaruit opgehaald. Er vindt dus een translatie plaats van een inputstream (van het internet) naar een bitmap (die op de schijf opgeslaan wordt). Die gegevens worden dan vervolgens in een Drawable ingeladen. Als thumbnail wordt ook gewoon de hoge kwaliteitsversie teruggegeven zodat slechts één versie van elke afbeelding bewaard moet worden. De identificatie van elke afbeelding waarmee hun cachelocatie kan teruggevonden kunnen worden is een hash die gegeneerd wordt op basis van hun URL. Dus als een afbeelding op eenzelfde locatie door meerdere bronnen teruggegeven wordt, dan zal de afbeelding slechts eenmalig opgehaald worden.

De cachelocatie kan door Android leeg gemaakt worden als het telefoongeheugen vol loopt. Maar het is beter als de applicatie zelf het geheugen dat het gebruikt beperkt. Daarom werd de code zo geschreven dat de afbeeldingen automatisch verdwijnen als de virtuele machine gestopt wordt. Het kan zijn dat als deze niet op een mooie manier afgesloten werd, de afbeeldingen niet gewist werden. Daarom is het ook nodig dat periodiek de cache geleegd wordt. Hiervoor zal ook een optie in het instellingen menu voorzien worden.

3.8 Onderzoek woordenboek API's

Als een zin in de applicatie ingevoerd wordt, moet hierover voldoende informatie worden verkregen zodat de belangrijkste woorden er uit kunnen gehaald worden. De belangrijkste woorden zijn die woorden die de context van de zin kunnen aanduiden. Andere woorden worden er uit gefilterd, zodat slechts een selectie van de zin naar de zoekbronnen doorgestuurd wordt.

De mogelijkheden hiervoor lopen uiteen. Er kan gebruik gemaakt worden van woordenboeken, waar woordsoortinformatie opgezocht kan worden, part-of-speech taggers, die via een slimme analyse van de zin een betere woord-

soortinformatie teruggeven, frequentietabellen, waarin te vinden is hoe vaak bepaalde woorden in een taal voorkomen en een thesaurus.

Een thesaurus bevat voor een verzameling woorden de aanverwanten. Elk woord dat er in opgenomen is kan een lijst bevatten van synoniemen, hyperoniemen (woorden die een ruimer begrip beschrijven), hyponiemen (woorden met een engere betekenis), antoniemen (woorden met een tegengestelde betekenis) of begrippen die er aan verwant zijn, een andere nuance uitdrukken of een overlappende betekenis hebben [29].

In de uiteindelijke applicatie wordt gebruik gemaakt van een woordenboek en een part-of-speech tagger. Frequentietabellen en een thesaurus worden nog niet gebruikt, maar deze zouden bij toekomstige ontwikkelingen aan de applicatie wel uitgewerkt kunnen worden.

3.8.1 Google Woordenboek

Zoekgigant Google heeft naast zijn bekende zoekmachine ook een woordenboekzoekmachine. Sommige webmasters ontdekten reeds ongedocumenteerd de mogelijkheid om deze engine op een REST achtige manier aan te spreken:

```
http://www.google.com/dictionary/json?
    callback=dict_api.callbacks.id100&q=auto&sl=en&tl=en&
    restrict=pr%2Cde&client=te
```

Helaas hebben sommige ontwikkelaars, die een applicatie rond deze API opgezet hadden, van Google de vraag gekregen [27] om deze niet meer te gebruiken. Google's contracten bepalen namelijk dat de informatie niet aan derde partijen mag aangeboden worden.

Omdat deze methode niet rechtstreeks gedocumenteerd is en niet legitiem te gebruiken is, wordt deze niet gebruikt in de applicatie.

3.8.2 WordNet

WordNet is een grote lexicale databank met Engelstalige woorden. Deze is ontwikkeld aan de universiteit van Princeton en bevat woorden gegroepeerd in verzamelingen van synoniemen, korte definities en de semantische relaties tussen de verzamelingen.

Deze database is opgezet met een tweeledig doel: enerzijds het aanbieden van een woordenboek en anderzijds het ter beschikking stellen van een thesaurus

met ondersteuning voor automatische analyse van tekst en kunstmatige intelligentietoepassingen. De database is beschikbaar gesteld onder een BSD licentie, die licentie houdt in dat de programmatuur vrij gedownload en gebruikt kan worden. De gegevensbank is ook online te raadplegen.

Dit project was de inspiratie voor het EuroWordNet project. Dit project wou dezelfde informatie aanbieden, maar dan voor enkele Europese talen: Nederlands, Italiaans Spaans, Duits, Frans, Tsjechisch en het Estlands. Het project liep van 1996 tot 1999 en is ondertussen afgelopen. Via dit project is te vinden dat uit dit project een organisatie gevloeid is: *The Global WordNet Association* die een kosteloze, publieke en niet-commercile organisatie is. Deze organisatie heeft tot doel het aanbieden van een platform voor discussie, delen en het verbinden van *Worknets* voor verschillende talen.

Deze organisatie geeft ook informatie over een Nederlandstalige wordnet, ontwikkeld door de Universiteit van Amsterdam in samenspraak met Van Dale B.V.. Helaas is het gebruik ervan beperkt en niet gratis. Voor een Nederlandstalige Lexicon, bestaande uit 64 duizend woorden, moeten prijzen betaald worden tussen de 12 800 en 102 400 euro.

Op de Android markt is reeds een WordNet applicatie te vinden om Engelstalige woorden in op te zoeken.

Omdat we geen Nederlandstalige WordNet konden vinden die gratis gebruikt mocht worden liep dit spoor dood. Het Engelstalige WordNet zou gebruikt kunnen worden voor een Engelstalige communicatiehulp.

Voor de Nederlandstalige communicatiehulp had de WordNet database gebruikt kunnen worden in combinatie met een vertaalwoordenboek. Een woord zou eerst naar het Engels vertaald kunnen worden, opgezocht worden en de resultaten zouden terug naar het Nederlands vertaald kunnen worden. Door de dubbele vertaling zou de kwaliteit hiervan heel erg laag liggen en is deze methode dus niet uitgewerkt.

3.8.3 Van Dale

Voor tien euro is het mogelijk het Van Dale woordenboek op de Android telefoon te downloaden. Na een contact met de makers van die software was te horen dat de API en de broncode vertrouwelijke gegevens zijn die niet publiek verdeeld worden aan klanten of derde partijen. Ook deze bron was dus niet bruikbaar.

3.8.4 LookWAYup

LookWAYup [26] heeft een online databank opgezet, gebaseerd op WordNet en andere bronnen. Het geeft vertalingen tussen de volgende talen: Engels, Frans, Spaans, Duits, Nederlands en Portugees. De tool maakt gebruik van morfologie, gerelateerde woorden, spellingcorrecties en afgeleide woorden.

Helaas zijn de definities ook enkel maar te vinden voor Engelstalige woorden en dus niet bruikbaar voor de Nederlandstalige communicatiehulp.

3.8.5 OpenTaal

Dit project [25] maakt Nederlandstalige hulpbestanden aan die vrij te gebruiken zijn in opensourceprojecten. Ze werken aan spellingscontrole, woordafbreking, synoniemenlijsten en grammaticacontrole. Het doel van het project is om de Nederlandse taal zo volledig mogelijk te ondersteunen. De database wordt onder andere gebruikt in de programmatuur van OpenOffice.org.

Het project werkt met een internetrobot, genaamd *Harvester*. Deze zoekt waar woorden op het internet gebruikt worden, de alinea's die een woord bevatten worden dan opgehaald. Ze noemen dit ook wel *taal oogsten*.

Op de site van het project [25] heb ik trouwens ook een nieuwsitem mogen plaatsen over deze masterproef.

Meehelpen aan het project kan door een ZIP-bestand te downloaden met de Java bestanden van de Harverster en enkele hulpbestanden. Je start het project vervolgens in een terminal venster:

```
java -jar Harvester.jar -window LauBook Laurens
```

Daarna wordt een woord verkregen van de OpenTaal-server waar dan vervolgens op internet naar gezocht wordt.

```
Machine name set to: LauBook, user to: Laurens
searchText:ochtendwandelingetje
.......
.........................
(11:28) hits:4020, blocks:828, exact hits:774;-)#+#
searchText:inkoopprijswijzigingen
..................
(11:30) hits:21, blocks:21, exact hits:16;-)#+#
searchText:conserveringsprocessen
```

. . . . . .

Codefragment 3.10: Voorbeeld output van het Harvester programma.

Het oogsten gebeurt zoveel mogelijk op websites van een aantal overheidssites en kranten. Deze sites zouden namelijk een goed oog moeten hebben voor correct taalgebruik.
De belasting en de grootte ervan is bijzonder laag.

Het programma wordt gebruikt om goede voorbeelden te vinden van de wat meer zeldzame woorden. Voor een bepaald zeldzaam woord worden zinnen gezocht waarin het woord gebruikt wordt en die worden dan aan de voorbeeldenverzameling toegevoegd. Zo kan vervolgens beter worden ingeschat of het correct is of niet en wat voor soort woord dat is.

Het OpenTaal-project geeft enkel *dictionaries*. Deze zien er bijvoorbeeld als volgt uit:

```
confiscabel/E
punnikte
daklekkage
duimendik
hoorntje/S
aimabel/E
obligatiehandelaar
aartsdomme
drijf/AFCKUZVB
walsmaat
```

Fragment 3.11: Fragment van een *dictionarie* van het OpenTaal-project.

De letters rechts van de woorden na de schuine streep duiden op afleidingen die van de woorden gemaakt kunnen worden. Ze bepalen bijvoorbeeld hoe het verkleinwoord of de meervoudsvorm gemaakt wordt.

Naast deze woordenlijst met *.dic* als bestandsextentie, is er ook een *.aff* bestand meegeleverd. Dit bevat de structuur van de gebruikte afkortingen in de woordenlijst. Zoals ook te vinden in de - door OpenOffice.org component - Hunspell [30].

Bijvoorbeeld bij *drijf* in bovenstaande voorbeeld is zo te vinden dat het meervoud *drijven* is en dat er een heleboel suffixen aan het woord kunnen gehangen worden: *aandrijf, afdrijf, indrijf, ..*

In deze bestanden is helemaal geen informatie terug te vinden over de woordsoorten en daarom waren deze dus helaas ook onbruikbaar bij het ontwikkelen van de filter.

Na contact met Ruud Baars[4] via de OpenTaal-mailinglist [31] is een databank verkregen met woordsoortinformatie. Deze databank bevat woordsoortinformatie voor zo'n 130 000 woorden. De databank is nog niet als download aangeboden op de website van het project omdat de kwaliteit nog niet optimaal bevonden is. De databank mocht gebruikt worden als aan de licentievoorwaarden voldaan werd. Deze hielden in om een bronvermelding naar het project te plaatsen en mee te helpen waar mogelijk. De bronvermelding is aanwezig in het informatiescherm van de applicatie en zowel de woordenzoeker als de zinnenoogster van het project hebben af en toe gedraaid.

In de uiteindelijke applicatie wordt gebruik gemaakt van deze databank om woordsoortinformatie op te zoeken. De databank wordt met de applicatie meegeleverd en wordt geïnstalleerd op het mobiele toestel de eerste keer dat het programma opgestart wordt. Er kan eenvoudig ingesteld worden welke woordsoorten belangrijk zijn voor de context van een zin. Deze selecteert nu bijvoorbeeld de zelfstandige naamwoorden en al dan niet vervoegde werkwoorden.

Dat de databank met de applicatie meegeleverd wordt zorgt er wel voor dat de applicatie wel heel erg groot wordt, maar opzoekingen gebeuren dan een stuk sneller dan als een externe bron geraadpleegd moet worden.

Eventueel zou in een commerciële versie van de applicatie er voor gekozen kunnen worden om de databank achteraf te laten downloaden van een server. Zo kan steeds een geüpdatet versie van de databank opgestuurd worden en kunnen databanken in verschillende talen aangeboden worden.

Deze manier van filteren is helaas niet optimaal omdat sommige woorden op verschillende manieren kunnen gebruikt worden. Zo zijn bijvoorbeeld *wil* en *bal* zowel een zelfstandig naamwoord als een vervoegd werkwoord. Dat probleem is op te lossen door Frog (zie hoofdstuk 3.8.7.1) als filter te gebruiken.

Er is nagedacht over een verbeterd gebruik van de databank. Zo kan bijvoorbeeld een zwarte lijst gebruikt worden die woorden zoals willen, kunnen, hebben automatisch gaat uitfilteren, maar er zijn enkele voorbeelden te be-

[4]E-mail: baarsrj@xs4all.nl

denken waarin deze woorden toch belangrijk zijn voor het resultaat. Zoals in de volgende zin: *"Ik heb een eigen wil"*.

Een volgend idee is om enkel zelfstandige naamwoorden over te houden met een lidwoord voor, maar ook hier zijn tegenvoorbeelden voor te vinden. Zo kan men wel correct *"bal"* vinden in *"Ik wil graag met de bal spelen"*, maar *"vulpennen"* heeft geen lidwoord voor het woord in *"Thuis heb ik veel vulpennen"*.

Een ander idee was om enkel het laatste werkwoord in een reeks werkwoorden over te houden, maar ook dit is geen ideale oplossing. Al deze problemen zijn veel beter op te lossen door het goed gebruik van de output van het Frog commando.

De databank die origineel van de auteurs verkregen is, was een plat tekstbestand met een entry op elke lijn. Dit bestand is omgezet naar een SQLite databank zodat snellere opzoekingen mogelijk zijn aan de hand van SQL queries.

De programmacode die gemaakt is voor deze omzetting is aanwezig in bijlage C op pagina 104.

3.8.5.1 OpenThesaurus

OpenThesaurus is een databank die het OpenTaal-project opgezet heeft met als doel een synoniemenlijst aan te bieden. Deze bevat momenteel ongeveer 130 duizend woorden en 140 duizend verzamelingen van synoniemen. De huidige versie van de databank kan gedownload worden.

In een verdere ontwikkeling van de voorgestelde applicatie zou deze databank gebruikt kunnen worden voor het aanbieden van synoniemen.

3.8.6 Wiktionary

Dit project [32] heeft tot doel een meertalig, vrij woordenboek aan te maken waar woordbetekenissen, vertalingen, woordherkomsten en uitspraken in beschreven worden. De site draait op de Wikimedia programmatuur die ook voor het bekende Wikipedia project gebruikt wordt. De informatie is vrij beschikbaar onder de GNU licentie en er is een API beschikbaar.

Over de API is bijzonder weinig informatie te vinden [33]. De woordsoorten kan men opzoeken door een URL op te stellen:

```
http://nl.wiktionary.org/w/api.php?format=json&action=query&
    titles=ik|ga|met|de|bal|spelen&export=true
```

In het JSON antwoord dat je verkrijgt, kan je vervolgens verschillende *page* objecten ontdekken. Elke *page* stelt een woord voor, vervolgens kan programmacode geschreven worden die in die *pages* op zoek gaat naar termen zoals bijvoorbeeld:

nlpronom-pers	Persoonlijk voornaamwoord
nlnoun	Zelfstandig naamwoord

In de huidige applicatie is deze methode niet uitgewerkt aangezien de databank van het OpenTaal-project 3.8.5 reeds woordsoortinformatie teruggeeft.

De databank van het OpenTaal-project kan meegeleverd worden en is dus een stuk sneller.

3.8.7 Part-of-speech

Enkel de woordsoorten gebruiken om te bepalen welke woorden naar de API's gestuurd moeten worden is onvoldoende. Neem je bijvoorbeeld de zin *"Ik wil met de bal spelen"*. Programmeer je dat enkel de infinitieven en zelfstandige naamwoorden over gehouden worden, dan ga je *wil, bal* en *spelen* overhouden, terwijl het beter was geweest als enkel *bal* en *spelen* waren overgebleven. *Wil* is namelijk naast vervoegd werkwoord ook een zelfstandig naamwoord.

Een Part-of-speech is een softwaremethode die die kenmerken gaat toekennen op basis van de informatie van de volledige zin. De tagger gaat de juiste keuze van woordsoort gaan maken op basis van de volledige zinsconstructie. Zo komt *wil* van het werkwoorden willen, maar is *een wil* ook een zelfstandig naamwoord.

Er bestaan twee soorten taggers. Een trigramtagger wordt getraind met reeds aangeduide zinnen en gaat de kansen berekenen dat een woord tot een bepaalde categorie kan behoren. Een regelgebaseerde tagger daarentegen gaat werken op basis van lingustische regels. Hybride vormen zijn er ook.

3.8.7.1 Frog

Aan de Universiteit van Tilburg zijn een aantal software pakketten uitgewerkt in verband met taalanalyse. Een ervan is Frog, een Part-of-speech tagger van de eerste soort (een trigramtagger). Frog gaat de woorden van een zin

opsplitsen in verschillende delen en geeft een tab-gelimiteerde output met voor elk woord de basisvorm, lemma, omschrijving van de vorm, een tagnummer en de relatie met het hoofdwoord.

Frog wordt vrijgegeven onder de gratis GNU General Public licentie. Maar Frog heeft een heleboel vereisten en kan dus moeilijk op het mobiele toestel zelf gedraaid worden.

Daarom is er voor gekozen dit op een server te draaien en vanop het toestel aanvragen door te sturen naar die server.

3.8.7.2 Frog installatie

Voor de installatie van Frog is Debian Linux gebruikt. Een handige feature van Debian is namelijk het Debian package management system. Deze collectie van tools automatiseert het proces van het installeren, upgraden en configureren van programma's.

Als men vertrekt van een systeem met daarop een up to date versie van het Debian besturingsysteem, dan kunnen de volgende stappen gevolgd worden om tot een draaiende versie van de Frog server te komen. Deze is onbeveiligd, dus extra toegangscontrole op een hogere laag kan nodig zijn.

1. Eerst kunnen reeds een aantal packages via het debian package management systeem geïnstalleerd worden. Om de packages van de ILK onderzoeksgroep gemakkelijk te installeren kan de bronnenlijst van het package management systeem aangevuld worden:

```
echo deb http://apt.ticc.uvt.nl lenny main contrib
   non-free > /etc/apt/sources.list.d/ticc.list
wget -O - http://apt.ticc.uvt.nl/public.key
apt-key add -
apt-get update
```

2. Hierna kunnen een aantal packages geïnstalleerd worden die nodig zijn om Frog te kunnen draaien:

```
apt-get install pkg-config timbl timblserver timbl python-
   django libxml2 gconf2 libgconf2-dev libxml2 python
   build-essential mbt libicu-dev python python-dev
   libxml2 libgnome2-dev gnome-devel
```

3. Ucto is ook nodig en moet manueel geïnstalleerd worden. Dit kan bijvoorbeeld zo:

```
wget http://ilk.uvt.nl/downloads/pub/software/
   ucto-latest.tar.gz
tar xzvf ucto-latest.tar.gz
cd ucto-0.4.4/
./configure
make && make install
```

4. Dan kan Frog gedownload, uitgepakt, ingesteld en geïnstalleerd worden. De prefix optie duidt de locatie aan waar de uiteindelijke binaire bestanden geplaatst worden:

```
wget http://ilk.uvt.nl/downloads/pub/software/
   frog-0.11.1.tar.gz
tar zxvf frog-0.11.1.tar.gz
cd frog-0.11.1
./configure --prefix=/frog
make && make install
```

5. Frog kan dan opgestart worden met een tekstbestand als input. Maar veel handiger is de server functionaliteit van Frog. Een server start je zo bijvoorbeeld op poort 123:

```
cd /frog/bin
(./Frog -S 123) &
```

6. Via een socket kan dan verbinding gemaakt worden naar de Frog service. Dit wordt gedemonstreerd in codefragmenten 3.12 en 3.13.

```
# telnet 91.198.203.3 443
Trying 91.198.203.3...
Connected to 91.198.203.3.
Escape character is '^]'.
Ik wil graag met de bal spelen.
1 Ik  ik  [ik]  VNW(pers,pron,nomin,vol,1,ev) 2 su
2 wil willen  [wil] WW(pv,tgw,ev) 0 ROOT
3 graag graag [graag] BW()  7 mod
4 met met [met] VZ(init)  7 pc
5 de  de  [de]  LID(bep,stan,rest)  6 det
6 bal bal [bal] N(soort,ev,basis,zijd,stan) 4 obj1
7 spelen  spelen  [speel][en] WW(inf,vrij,zonder) 2 vc
8 . . [.] LET() 7 punct

READY
```

Codefragment 3.12: Het onderzoeken van het gedrag van de Frog server software met behulp van telnet.

```
Socket socket = new Socket(InetAddress
    .getAllByName("91.198.203.3")[0], 443);
BufferedReader in = new BufferedReader(new
    InputStreamReader(socket.getInputStream()));
PrintWriter out = new PrintWriter
    (socket.getOutputStream(), true);
String line;
while ((line = in.readLine()) != null && !line.equals(""))
  Helper.getHelper().log("Frog␣resultaat:" + line);
```

Codefragment 3.13: Frog aanspreken vanuit Java.

3.9 JSON

De JavaScript Object Notation wordt gebruikt voor het uitwisselen van datastructuren, meestal in webapplicaties. Het is eenvoudig voor mensen om het te lezen en te schrijven en voor machines om het te ontleden en te genereren. Het is gebaseerd op JavaScript en dankt daar ook zijn succes aan: omdat de gegevens voorkomen in de vorm van JavaScript-expressies kunnen deze in deze taal zeer eenvoudig gebruikt worden.

Er zijn in JSON twee structuren terug te vinden:

1. Een combinatie van naam/waarde-paren. Deze worden in verschillende programmeertalen dan gelezen als *objecten*, *records*, *structs*, *woordenboeken*, *hash tabellen*, *lijsten* en *associatieve arrays*.

2. Een gesorteerde lijst van waarden. Deze worden geïnterpreteerd als *array*, *vector*, *lijst* of *sequence*, afhankelijk van de gebruikte programmeertaal.

Deze waarden kunnen onderscheiden worden als verschillende types: *objecten*, *strings*, *nummers*, *arrays*, *true*, *false* of *null*.

3.10 OAuth

OAuth is een open protocol dat toelaat om aan veilige API authorizatie te doen op een eenvoudige en gestandaardiseerde manier voor desktop en webprogramma's.

3.10.1 Inleiding

Veel luxe auto's worden tegenwoordig geleverd met een speciale sleutel. Deze kan je aan de piccolo bezorgen die je auto kan parkeren. De sleutel is speciaal omdat deze slechts toelaat om enkele kilometers ver te rijden. Sommige van die speciale sleutels zorgen er voor dat de koffer niet open kan, het adresboek van je telefoon niet gelezen kan worden, .. Het is een fantastisch idee: je kan iemand beperkte toegang verlenen tot je auto en zelf je gewone sleutel gebruiken om alles te openen.

Nieuwe websites worden ontwikkeld die functionaliteiten van andere sites gebruiken. Een drukkerij die jouw online foto's afdrukt in een boek, een sociale netwerksite die jouw adresboek gebruikt om nieuwe vrienden op te zoeken, een API die gebruikt kan worden om je eigen offline versie van een site te ontwikkelen, het zijn allemaal voorbeelden van toepassingen die jouw toestemming nodig hebben om die informatie te kunnen opvragen bij de andere site. Bezorg je die jouw eigen wachtwoord dan hebben die sites volledige toegang en kunnen ze dus doen wat ze willen.

OAuth beperkt de toegang. Het zorgt er voor dat bepaalde informatie (geleverd door de **service provider**) gelezen kan worden door een andere site (de **consumer**, niet te verwaren met jezelf: de **user**).

Bij de ontwikkeling van de standaard is gekeken naar andere vergelijkbare protocollen zoals Google AuthSub, AOL OpenAuth, Yahoo BBAuth, Upcoming API, Flickr API en Amazon Web Services API. De beste methoden zijn er uit overgenomen.

3.10.2 De verschillende soorten sleutels

De consumer key en consumer secret

De *consumer key* is een unieke identificatie voor de *consumer*. De *consumer* gebruikt de *consumer* secret om de aanvragen te ondertekenen.

Het request token en het token secret

Het *request token* wordt gegenereerd door de service provider en is een tijdelijke en slechts eenmalig gebruikte identificatie die gebruikt wordt om de *user* te laten inloggen op de site van de *service provider*. Het *token secret* wordt gebruikt om de aanvraag te ondertekenen.

Het access token en token secret

Het *access token* is een identificatie die gebruikt wordt door de *consumer* om informatie bij de *service provider* op te halen. De *service provider* kan de sleutel op elk moment als ongeldig bestempelen als deze vervallen is of omdat de gebruiker deze ingetrokken heeft. Het token secret wordt opnieuw gebruikt om de aanvragen te ondertekenen.

3.10.3 Het 3-legged OAuth proces

Hier wordt het zogenaamde 3-legged OAuth-proces beschreven. Dit is de meest gebruikte methode van OAuth en ook de meest gedocumenteerde.

1. Een ontwikkelaar van de consumer vraagt een *consumer key* and *consumer secret* aan bij de service provider.

2. Een user (gebruiker) zit op een website of programma van de consumer en voert een actie uit. Voor die actie heeft de *consumer* informatie nodig die de service provider moet leveren. De *consumer* vraagt een *request token* aan de service provider.

3. Wanneer de *consumer* de *request token* ontvangt, stuurt deze de *user* door naar de *OAuth User Authorization URL* van de *service provider* waar de *user* zijn identiteit kan bewijzen door in te loggen. Met deze aanvraag wordt ook een internetadres meegestuurd waar de *user* naar toe kan doorverwezen worden nadat deze succesvol bij de *service provider* is ingelogd. Op geen enkel moment zijn de gebruikersnaam en wachtwoord combinaties op de *consumer* website ingegeven. Deze zijn enkel op de site van de *service provider* gebruikt.

4. Wanneer de *user* ingelogd is op de site van de *service provider*, dan markeert deze *service provider* de *request token* als *user-authorized*. De user wordt naar de pagina doorverwezen die met de *request* is meegestuurd. De *request token* wordt meegestuurd.

5. De *geauthorizeerde request token* wordt hierna door de *consumer* gebruikt om een *access token* aan te vragen.

6. Het is deze code die vervolgens gebruikt kan worden door de *consumer* om de nodige informatie op te vragen aan de *service provider*. Deze code kan meermaals gebruikt worden.

   Wanneer een aanvraag gebeurt naar een beveiligde webpagina, dan voegt de *consumer* een *authorization* hoofding toe (of query parameters) met

daarin de *consumer key*, het *access token*, de methode die gebruikt is voor het ondertekenen, een tijdsaanduiding, een *nonce* en optioneel ook de versie van het OAuth-protocol dat gebruikt is.

Een nonce is een getal dat een protocol slechts één keer zal gebruiken. Als het OAuth-protocol een nonce eenmaal heeft gebruikt, zal het deze verder nooit meer gebruiken.

3.10.4 Het 2-legged OAuth proces

Meestal verwijst de OAUth term naar 3-*legged* OAuth, wat dan ook als de standaard methode wordt aanzien, maar er bestaat ook 2-*legged* OAuth. Dit is de methode die ook gebruikt is in de ontwikkelde applicatie. 2-*legged* OAuth wordt gebruikt om in authenticatie te voorzien bij de API aanvragen die bij Vimeo gebeuren.

Wat de 2-*legged* OAuth minder heeft dan de 3-*legged* OAuth noemt men een *dance*. Een *dance* is de workflow waar de gebruiker van de *consumer* site naar de site van de service provider gestuurd wordt, daar moet inloggen en dan teruggestuurd wordt. De 2-*legged* OAuth variant wordt ook wel *phone home* genoemd omdat deze methode ook wel vaak gebruikt wordt in gadgets die een verbinding maken naar een achterliggende server. De identificatie van de gebruiker wordt soms weggelaten omdat deze gewoon niet nodig is.

Bij deze variant is dus geen access token nodig. Er wordt een *request token* met een *token secret* aangevraagd[5] aan de *service provider* en het is juist die sleutel die in de *authorization* hoofding meegegeven wordt.

[5]Met behulp van de *consumer key* en *consumer secret*

Hoofdstuk 4

Implementatie

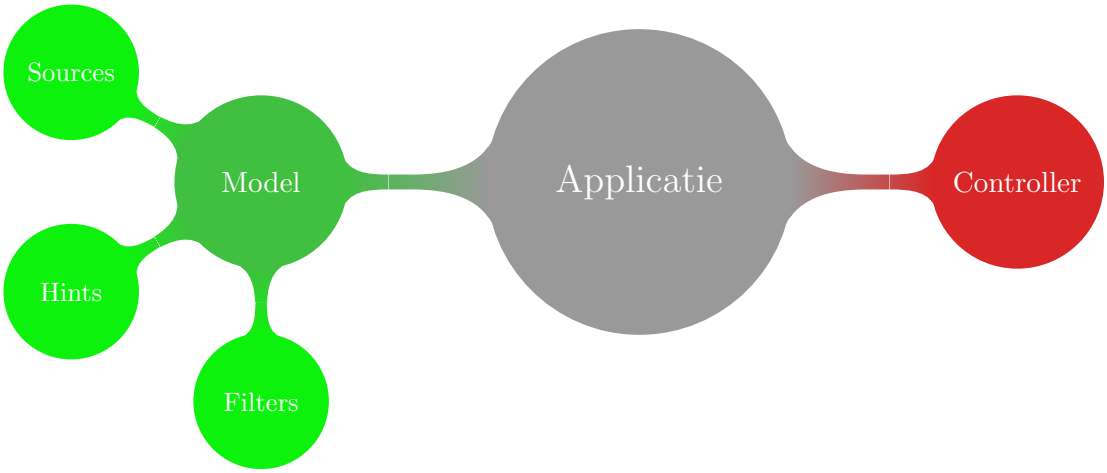
4.1 Structuur

De Java klassen zijn ondergebracht in vijf packages:

- **controller:** bevat alle schermen van de applicatie.
- **model:** bevat alle hulpklassen, interfaces en abstracte implementaties voor de bronnen en filters. De abstracte implementaties van bronnen en filters bevatten gemeenschappelijke programmacode.
- **model.filters:** bevat alle filters die met de applicatie meegeleverd worden.
- **model.sources:** bevat alle concrete uitwerkingen van publieke bronnen.
- **model.hints:** bevat alle concrete uitwerkingen van hint genererende systemen. De interface waar ze allen aan voldoen, zit in de model package.

Een Android-applicatie is een container met daarin een aantal mappen die resources en Java class-bestanden bevatten. Een aantal mappen met hun inhoud worden besproken.

- Alle constanten worden ondergebracht in de *res/values* map, van die container, in xml-bestanden. Elk soort variabele zit in een apart bestand. Bijvoorbeeld: *colors.xml of strings.xml* Het voordeel hiervan is dat er op deze manier gemakkelijk andere xml-bestanden kunnen geserveerd worden voor andere instellingen.



Figuur 4.1: Overzicht van de packages in de applicatie.

Zo kunnen andere vertalingen weergegeven worden op basis van de taal of de locatie van de telefoon, of worden bepaalde omgevingsvariabelen anders ingesteld als de telefoon in landscape modus wordt gehouden.

```
<resources>
    <color name="text">#fff</color>
    <color name="background">#000</color>
</resources>
```

Codefragment 4.1: Voorbeeld van een resource bestand: colors.xml.

- In de *res/raw*-map van de container wordt ook een binair bestand meegeleverd dat de *gestures* bevat. *Gesture recognition* is een techniek die poogt om menselijke gebaren, uitgevoerd met het gezicht of de hand, te herkennen. Zo is interactie met machines mogelijk zonder mechanische toestellen [34]. In het programma is het namelijk mogelijk om naar de volgende of de vorige afbeelding te navigeren door een horizontale wrijfbeweging te maken over het touchscreen door respectievelijk van links naar rechts en van rechts naar links te glijden.

- De applicatiecontainer bevat ook een *assets* map waarin een SQLite 3 databank meegeleverd wordt. Die bevat ongeveer 350 duizend flexvormen, samen met hun basiswoord en het type woord. De data om deze databank aan te maken werd aangeleverd door het OpenTaal-project. Omdat deze databank niet rechtstreeks uit deze map gelezen kan worden, wordt deze binair gekopieerd naar het toestel bij de eerste keer dat de applicatie

opgestart wordt.

- De layouts van de schermen worden ook in xml-bestanden bijgehouden. Deze bevinden zich in de *res/layouts*-map. Vormgevingen definiëren in xml-bestanden maakt aanpassingen doorvoeren eenvoudiger. Je kan op een eenvoudige manier een andere vormgeving instellen voor de landscape-modus.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/
       android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <view
      class="be.hogent.laurens.model.MyViewFlipper"
      xmlns:android="http://schemas.android.com/apk/res/
         android"
        android:id="@+id/OCRFlipper"
        android:layout_width="fill_parent"
        android:layout_gravity="center"
        android:layout_span="2"
        android:layout_weight="1.0"
        android:layout_height="0dip">

        <ProgressBar
          android:id="@+id/ProgressBar"
          android:layout_width="wrap_content"
          android:layout_gravity="center"
          android:layout_height="wrap_content"
      />

        <TextView
          android:id="@+id/OCRResult"
          android:gravity="center"
          android:textSize="13pt"
          android:layout_width="fill_parent"
          android:layout_height="fill_parent"
          android:maxLines="999"
          android:scrollbars="vertical"
          >
        </TextView>

    </view>
```

```
    <LinearLayout
        style="@android:style/ButtonBar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <Button
            android:id="@+id/OCROk"
            android:layout_width="0dip"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:onClick="onConfirm"
            android:enabled="false"
            android:text="@string/OCROk" />

        <Button
            android:id="@+id/OCRRedo"
            android:layout_width="0dip"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:enabled="false"
            android:onClick="onRedo"
            android:text="@string/OCRRedo"/>

    </LinearLayout>
</LinearLayout>
```

Codefragment 4.2: Voorbeeld van een layout bestand: ocr_result.xml.

4.1.1 Alternatieve resource-bestanden

Verschillende mappen kunnen in de *res*-map aangemaakt worden met daarin alternatieve configuratiebestanden voor andere omgevingen of instellingen. Als het toestel bijvoorbeeld ingesteld staat in nachtmodus, versie 9 van de API draait en de landcode staat op BE ingesteld, dan zullen layout bestanden, als deze map bestaat, eerst in de map *layout-be-night-v9* gezocht worden. Eerst wordt gezocht in de meest limiterende mappen en zo steeds verder tot in de algemene *layout* map.

In de communicatiehulp zijn zo twee specifieke mappen aangemaakt:

1. De map "**layout-land**" bevat één bestand *main.xml*. Dit bestand geeft een alternatieve weergave van het begin menu voor als het toestel horizontaal gehouden wordt. Horizontaal staan dan 3 tot 4 iconen naast elkaar, terwijl in de verticale weergave slechts 2 iconen naast elkaar staan.

2. De map "**values-land**"bevat één bestand *numbers.xml*. Dit bestand bevat een alternatieve waarde voor de integer waarde *HideMenuAfter*. Het menu bij de resultaten wordt namelijk sneller verborgen als het toestel horizontaal gehouden wordt omdat het dan voor de beelden komt en dus als storend ervaren kan worden.

4.2 Eigen klassen

In deze afdeling wordt een overzicht gegeven van alle klassen die voor de applicatie aangemaakt zijn, de package waar ze in thuishoren en hun functie.

4.2.1 De package 'controller'

In de *package* 'controller' zitten alle activities of schermen.

Analysis.java Dit is het analysevenster. Dit vraagt de tekst op en start een nieuwe thread die de filtering en de zoekopdrachten regelt. Als de thread klaar is gaat deze een output venster openen en de resultaten aan dat venster doorgeven.

CustomWindow.java Om elk venster eenzelfde layout te geven wordt van eenzelfde klasse afgeleid. Dit is de klasse waar gewone vensters van afleiden.

CustomWindowListView.java Dit is de klasse waar ListView vensters van afleiden.

CustomWindowPreferences.java Van deze klasse leiden de Preferences (instellingen) vensters af.

EditTextAid.java Dit venster geeft hulp bij het schrijven van teksten. In het tekstblok kunnen teksten ingevoerd worden en als de betekenis van een woord onduidelijk is, kan men op dat woord klikken. Dat woord wordt dan uitgebeeld in afbeeldingen.

Email.java Een e-mail kan in deze activity geselecteerd worden.

Internet.java Een browser wordt opgestart waar moeilijke zinnen kunnen aangeduid en doorgestuurd worden.

Keyboard.java Tekst kan in dit venster rechtstreeks ingevoerd worden.

OCR.java Deze activity staat in voor het nemen van de foto voor de tekstherkenning.

OCRResult.java Nadat de foto genomen is, gaat deze activity de verdere verwerking van de foto afhandelen.

Output.java Het Output venster staat in voor de weergave van de resultaten en het afspelen van filmpjes.

PartSelection.java Alle activiteiten die grote hoeveelheden tekst als input krijgen, sturen deze tekst eerst naar de PartSelection activity, zodat de gebruiker een deel van een zin kan selecteren.

Settings.java Deze preferences activity geeft een overzicht van de gebruikersinstellingen van de applicatie en geeft de mogelijkheid om deze aan te passen.

SMS.java Een SMS bericht uit de inbox kan geselecteerd worden.

Sources.java Hier kunnen instellingen voor de bronnen aangepast worden.

Speech.java Dit venster staat in voor de spraakherkenning.

Statistics.java Het statistiekenvenster geeft alle statistieken weer.

Welcome.java Het opstartmenu. Deze activity is eveneens de eerste die opgestart wordt als het programma start.

4.2.2 De package 'model'

In deze package zitten logische klassen, interfaces, abstracte filters en sources.

AbstractFilter.java Dit is de basis van elke filter.

AbstractSource.java Elke bronimplementatie is afgeleid van deze klasse.

DictionaryOpenHelper.java Deze klasse staat in voor de databanktoegang naar het Nederlandstalige woordenboek met de woordsoorten. Het is ook deze klasse die de databank naar het toestel kopieert als deze nog niet op het toestel aanwezig is.

DutchDictionaryFilter.java Deze klasse bevat de logica voor het bouwen van een filter, die gebruik maakt van het Nederlandstalige woordenboek met woordsoorten.

FrogFilter.java In FrogFilter.java zit logica om een filter te bouwen die gebruik maakt van de Frog service.

Goggles.java Goggles.java werd in een vorige versie van de applicatie gebruikt om tekstherkenning door de Goggles servers te laten uitvoeren. In de huidige versie wordt deze klasse niet meer gebruikt.

Helper.java Deze singleton klasse bevat allerhande code die centraal bewaard moet worden. Deze kan een referentie leveren naar de centrale HTTP client, debug informatie tonen en wegschrijven, alle filter, bronnen en hints klassen teruggeven, de telefoon laten vibreren en teksten opsplitsen in zinnen en kleinere delen.

iFilter.java iFilter.java is de interface die elke filter moet implementeren.

iHint.java Dit is de interface die elke hint genererend systeem moet implementeren.

ImageDownloader.java In een eerdere versie van de applicatie werd deze klasse gebruikt om een dialoogvenster weer te geven terwijl afbeeldingen gedownload werden. In de huidige versie worden deze op de achtergrond gedownload en is het niet meer nodig om te wachten tot alle afbeeldingen binnengehaald zijn.

ImageSwitcherFiller.java Deze klasse gaat in een nieuwe thread een afbeelding binnenhalen. Als de afbeelding binnengehaald is, gaat hij de afbeelding op de juiste plaats in de ImageSwitcher plaatsen.

ImageViewFiller.java Deze klasse doet hetzelfde als de ImageSwitcherFiller, maar dan voor een ImageView.

iSource.java Elke bron moet deze interface implementeren.

MyViewFlipper.java In de standaard ViewFlipper op het Android besturingssysteem zit een fout. Deze klasse omzeilt deze fout.

OCRengine.java Elke klasse die een vertaling kan leveren van een foto naar een afbeelding moet deze interface implementeren.

Result.java Resultaten worden in een instantie van deze klasse gestopt.

StatsHelper.java Deze klasse staat in voor het aanmaken en updaten van de databank met statistieken.

Wisetrend.java Deze klasse staat in voor de vertaling van afbeelding naar tekst met behulp van de API van WiseTrend.

4.2.3 Het package 'filter'

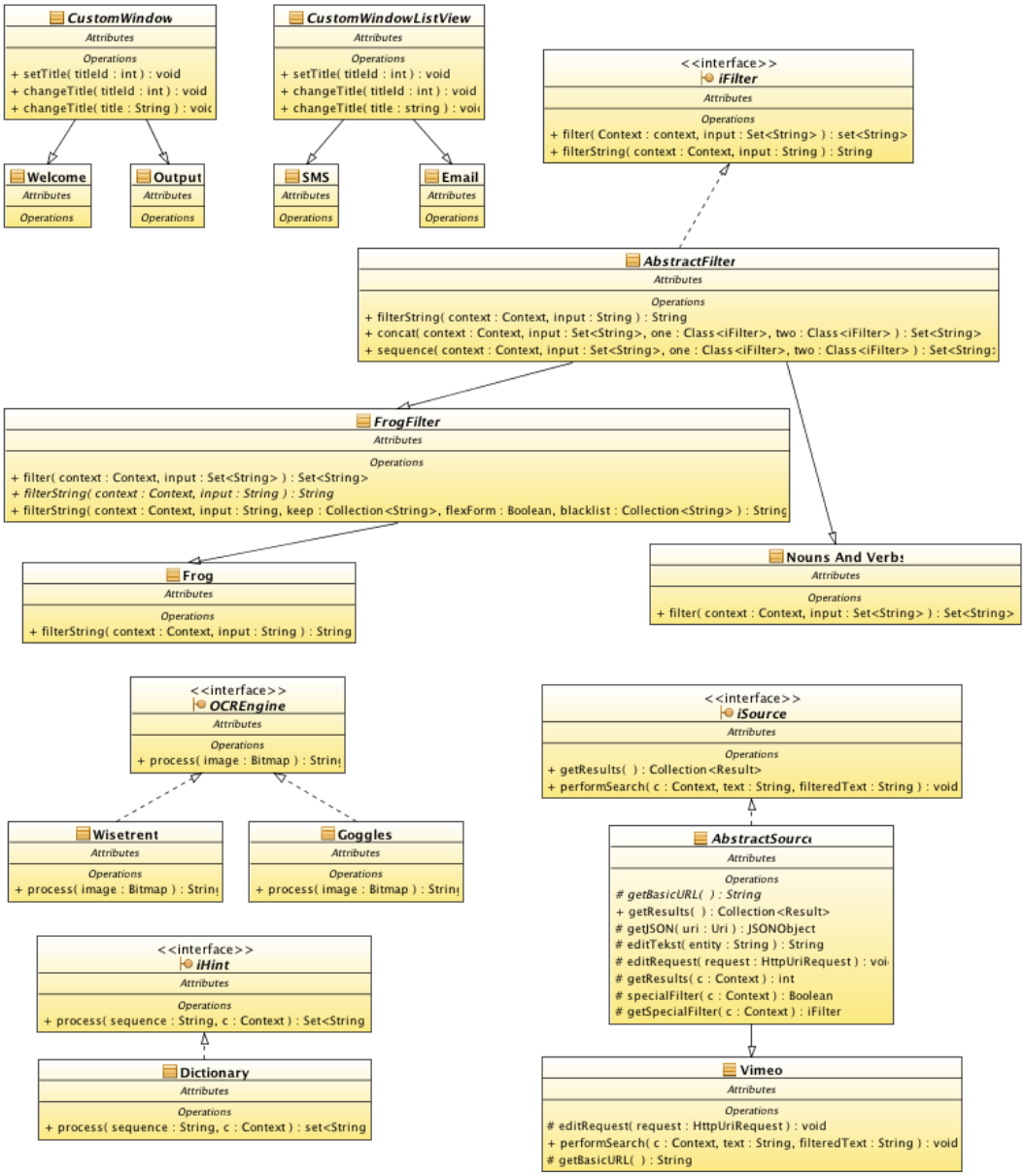
In de *package* 'filter' worden alle concrete implementaties van filters geplaatst.

Combination.java Deze klasse is een voorbeeldimplementatie van een filter die opgebouwd is uit twee andere filters. De woorden van beide filters worden samengenomen en als resultaat teruggegeven.

Frog.java Deze filter gaat de Frogservice gebruiken om een syntactische analyse van de ingevoerde zin te bekomen. In de klasse zit een soort witte lijst en een zwarte lijst. Als een woord een eigenschap heeft die op de witte lijst voorkomt, dan wordt dat woord behouden, tenzij er ook een eigenschap van de zwarte lijst in voorkomt. Het openen van een socket naar de Frog server gebeurt in de FrogFilter.java klasse in het model package.

NoFilter.java Deze filter laat geen woorden weg en stuurt de volledige zin naar de bronnen door.

NounsAndVerbs.java De Nouns And Verbs filter gaat met behulp van het meegeleverde Nederlandstalige woordenboek alle woordsoorten opzoeken. Daarna bewaart hij enkel de zelfstandige werkwoorden en al dan niet vervoegde werkwoorden.



Figuur 4.2: Vereenvoudigde voorstelling van het klassendiagram.

Sequence.java Deze voorbeeldimplementatie plaatst enkele filters achter elkaar. De woorden die overblijven van de eerste filter worden doorgestuurd naar een tweede filter.

4.2.4 De package 'hints'

In de *package* 'hints' worden alle concrete implementaties van hint genererende systemen geplaatst.

Dictionary.java Deze hints genererende klasse bevat een ingebouwd woordenboek wat door de gebruiker aangevuld kan worden. Bevat een zin een woord uit het woordenboek, dan wordt een hint gegenereerd van de beschrijving.

Verbs.java Deze klasse gaat voor elk vervoegd werkwoord opzoeken wat de infinitief is en daar bijgevolg een hint van genereren.

4.2.5 De package 'sources'

Hierin vind je alle concrete implementaties van bronnen.

Flickr.java Deze bron gaat afbeeldingen van de Flickr website halen.

GoogleImageSearch.java Afbeeldingen van de Google Image Search API worden met deze klasse opgezocht.

Vimeo.java Deze bron zoekt relevant filmmateriaal op op de Vimeo website. Alle API calls worden met OAuth geautoriseerd.

YouTube.java Deze bron zoekt naar YouTube filmpjes die de context kunnen verklaren.

4.3 Externe klassen

Naast de ingebouwde Android klassen wordt één externe bibliotheek gebruikt in de applicatie: de oauth-signpost bibliotheek.

4.3.1 De OAuth-signpost bibliotheek

De oauth-signpost bibliotheek maakt het eenvoudiger om HTTP-berichten op het Java-platform te ondertekenen volgens de *OAuth Core 1.0a*-standaard (meer informatie over OAuth is terug te vinden in hoofdstuk 3.10). Door zijn modulariteit en zijn flexibel design is de bibliotheek eenvoudig te combineren

met verschillende HTTP lagen. Ze is vrijgegeven onder de Apache licentie versie 2.

De auteurs van de bibliotheek hebben geprobeerd een API te ontwikkelen die met een minimaal aantal regels kan aangesproken worden en in staat is om HTTP-berichten te ondertekenen en tokens aan te vragen bij een OAuth service provider. Wat geen deel uitmaakt van de OAuth specificaties wordt overgelaten aan de HTTP-lagen.

In Android zit een fout in de *java.net.HttpURLConnection*-klasse die er voor zorgt dat bepaalde service providers niet aangesproken kunnen worden. Daarom moeten de *CommonsHttpOAuth** klassen gebruikt worden op Android. Deze zijn bedoeld om te gebruiken met de Apache Commons HTTP-klassen. Dat zijn de klassen die Android hoe dan ook al gebruikt voor HTTP bewerkingen.

De bibliotheek is in de applicatie geplaatst door het invoegen van twee .jar containers: *signpost-core-1.2.1.1.jar* en *signpost-commonshttp4-1.2.1.1.jar*. Ze wordt enkel maar gebruikt bij het spreken met de Vimeo API, zoals gedemonstreerd in codefragment 4.3. Een consumer wordt aangemaakt die in staat is de nodige sleutels op te halen en de HTTP request te ondertekenen.

```
OAuthConsumer consumer = new CommonsHttpOAuthConsumer(
   VIMEO_CONSUMER_KEY, VIMEO_CONSUMER_SECRET);
OAuthProvider provider = new CommonsHttpOAuthProvider(
 VIMEO_REQUEST_TOKEN_URL,
 VIMEO_ACCESS_TOKEN_URL,
 VIMEO_AUTHORIZATION_URL);
try {
 provider.retrieveRequestToken(consumer, OAuth.OUT_OF_BAND);
}
catch (OAuthException oe) {}
HttpUriRequest request;
consumer.sign(request);
```

Codefragment 4.3: Het ondertekenen van een Vimeo API request met de oauth-signpost bibliotheek.

4.4 Databankstructuur

In het programma komen twee databanken voor. Één databank werd reeds vooraf aangemaakt en wordt naar het toestel overgezet de eerste keer dat het programma opgestart wordt. Deze bevat het Nederlandstalige woordenboek

met de woordsoorten dat verkregen is via het OpenTaal-project. De andere databank wordt aangemaakt wanneer de eerste keer statistieken geüpdatet worden. Deze bevat logischerwijze de statistieken van het programma.

Het zijn beide SQLite 3 databanken.

4.4.1 De OpenTaal-databank

Deze databank bevat één tabel met de naam *dutch*. Hierin zitten drie VARCHAR kolommen. De *flexibleForm* kolom bevat afgeleide woorden, de *basic* kolom de basisvorm en de *tag* kolom de woordsoort.

De waarden onder de *flexibleForm* kolom moeten niet uniek zijn. Zo kan *"bal"* voorkomen als zelfstandig naamwoord en als vervoegd werkwoord.

Er is een index aangemaakt op de eerste kolom om zoekopdrachten te versnellen.

Er zitten momenteel 345301 rijen in de tabel. De code die gebruikt is om deze rijen te genereren is te vinden in bijlage C.

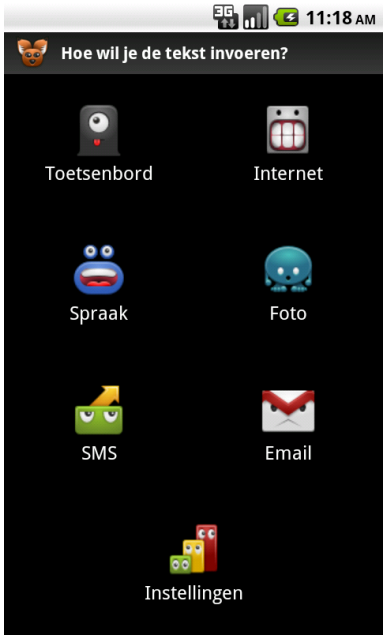
4.4.2 De statistiekendatabank

Deze databank bevat één tabel *statistics* met daarin twee kolommen: *definition* en *value*. De definition kolom heeft als eis dat de waarden erin uniek moeten zijn en bevat de beschrijving van de statistiek. Beide kolommen zijn tekst kolommen, al wordt de value kolom meestal gebruikt om getallen in te bewaren.

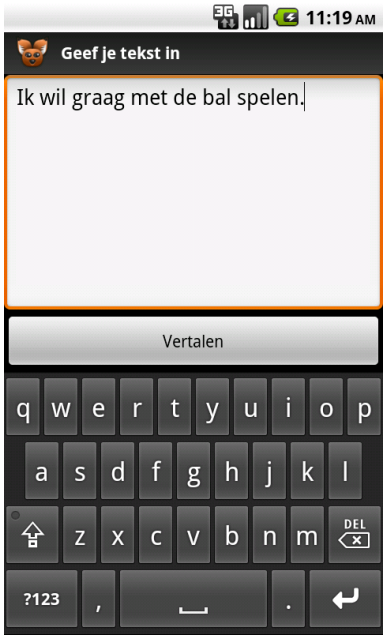
4.5 Voorbeelden van use cases

4.5.1 Invoer via het toetsenbord

Selecteert de gebruiker de optie *"Toetsenbord"* in het hoofdscherm, dan krijgt deze een tekstvak te zien waarin hij rechtstreeks tekst kan typen. Als geen hardwaretoetsenbord gevonden is, wordt automatisch een softwaretoetsenbord getoond. Schermafbeelding 4.4 verduidelijkt dit. Na het klikken op *"Vertalen"* wordt deze tekst doorgestuurd voor analyse.



Figuur 4.3: Menu van de applicatie.



Figuur 4.4: Invoer via het toetsenbord.

4.5.2 Webhulp

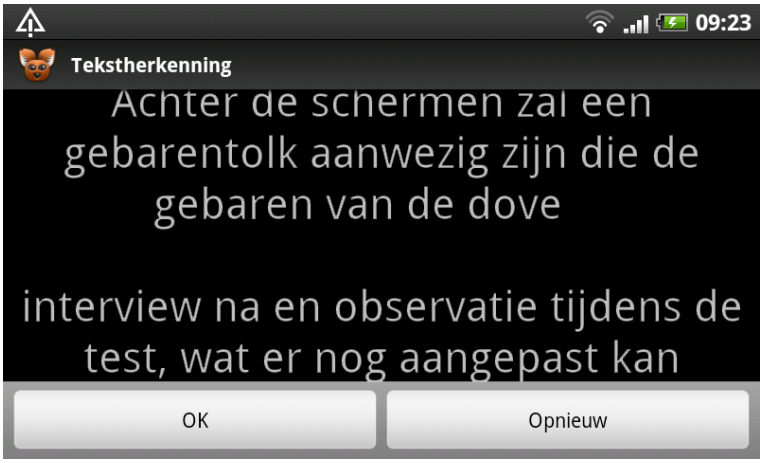
De gebruiker opent het programma. Deze krijgt hierna een keuzemenu te zien. In het keuzemenu kiest de gebruiker voor de optie *"Internet"*. De startpagina van de zoekmachine van Google opent. De gebruiker kan nu een zoekopdracht starten en naar een webpagina navigeren.

Dit betekent dat de gebruiker hier ook naar andere pagina's kan navigeren of formulieren kan invullen. Als de gebruiker een deel van een bepaalde zin of een volledige zin verduidelijkt wil zien, kan hij op de menu knop klikken en de virtuele *"Selectie"*-knop aanraken. Hierna kan hij een zin aanduiden. Die selectie wordt daarna doorgestuurd naar de zinsdeelselectie.

4.5.3 Tekstherkenning

De gebruiker neemt een foto met de mobiele telefoon van een stuk tekst. Het nemen van de foto kan via een aanraking op het scherm of via de hoofdknop onder het scherm. De software gaat voor het nemen van de foto eerst automatisch het beeld scherp stellen. Via OCR wordt de afbeelding omgezet in leesbare tekst. Hierna kan hij een nieuwe foto nemen door op de virtuele

"Opnieuw"-knop te klikken of door te sturen naar de zinsdeelselectie met de virtuele *"OK"*-knop. Een voorbeeldresultaat van de analyse een foto is te zien in figuur 4.5.



Figuur 4.5: Weergave van het resultaat van de tekstherkenning.

4.5.4 Spraakherkenning

De gebruiker activeert het programma op de mobiele telefoon. Hij krijgt een keuzemenu te zien en kiest voor het item *"Spraak"*. Wanneer de gebruiker klaar is om geluid op te nemen kan hij de virtuele *"Start"* aanraken. Er komt een microfoon op het scherm en een tekstuele melding: *"Nu spreken"*.

Via *"Annuleren"* kan de herkenning afgebroken worden. Als geen tekst kon herkend worden komt een foutmelding op het scherm met de tekst *"Geen resultaten gevonden"* en twee opties: *"Opnieuw spreken"* en *"Annuleren"*. Als alles goed gaat wordt de ingesproken zin omgezet naar tekst. Het resultaat verschijnt. Via de virtuele knop *"Opnieuw"* kan een nieuwe poging ondernomen worden of via *"OK"* kan het resultaat doorgestuurd worden voor analyse.

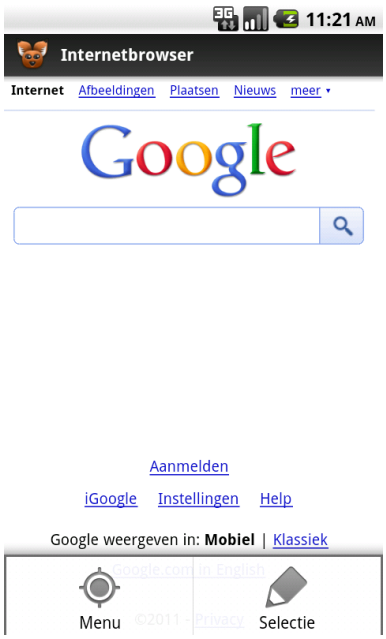
Uit een gebruikerstest met de applicatie is gebleken dat deze methode verfijnd zou kunnen worden: in plaats van de volledige zin opnieuw in te spreken zou de gebruiker een woord kunnen aanklikken. Enkel dat woord kan dan vervangen worden.

4.5.5 Hulp voor sms- of e-mailberichten

In het hoofdmenu kan gekozen worden voor de opties *"SMS"* of *"E-mail"*. Wordt één van beide opties gekozen, dan krijgt men een lijst te zien met alle berichten. Na het aanklikken van een bericht wordt het naar de zinsdeelselectie doorgestuurd.

4.5.6 Zinsdeelselectie

De zinsdeelselectie ontvangt tekst van een andere activity en gaat die opdelen in verschillende stukken. De tekst wordt gesplitst op leestekens en haakjes. De gebruiker kan de lijst bekijken met alle delen, een deel aanduiden en doorsturen voor analyse.



Figuur 4.6: Selectie op een internetpagina.

Figuur 4.7: Zinsdeel selectie in de applicatie.

4.5.7 Filteren en multimediabestanden ophalen

De tekst komt binnen en onbelangrijke woorden worden er uit gefilterd. Overgebleven woorden worden doorgestuurd naar publieke API's als zoekopdracht. Publieke API's krijgen een score die mee zal bepalen waar de resultaten in de lijst zullen verschijnen.

Deze score bestaat uit twee elementen: de score die de gebruiker ingesteld heeft bij de instellingen en een score die afhangt van hoeveel keer een item van deze publieke API als goed beeld aangeduid is. Noem het een soort kwaliteitsmeter. Als alle resultaten binnen zijn, kan de gebruiker de multimediabestanden bekijken.

4.5.8 Multimediabestanden bekijken

De gebruiker krijgt een lijst met resultaten te zien die afgehaald zijn bij verschillende publieke API's. De multimediabestanden die de grootste kans geven om de boodschap te verduidelijken worden voorin de lijst getoond. Er is telkens maar één afbeelding die getoond wordt. Men kan navigeren naar de volgende afbeelding door op het scherm met de vinger van rechts naar links te bewegen of naar de vorige afbeelding door van links naar rechts te bewegen.

Bij elke aanraking op het scherm verschijnt bovenin een lijst met alle afbeeldingen in het klein. Deze lijst kan ook van links naar rechts verschoven worden of de kleine beeldjes kunnen aangeklikt worden om in het groot te bekijken. Deze lijst verdwijnt terug automatisch na twee seconden.

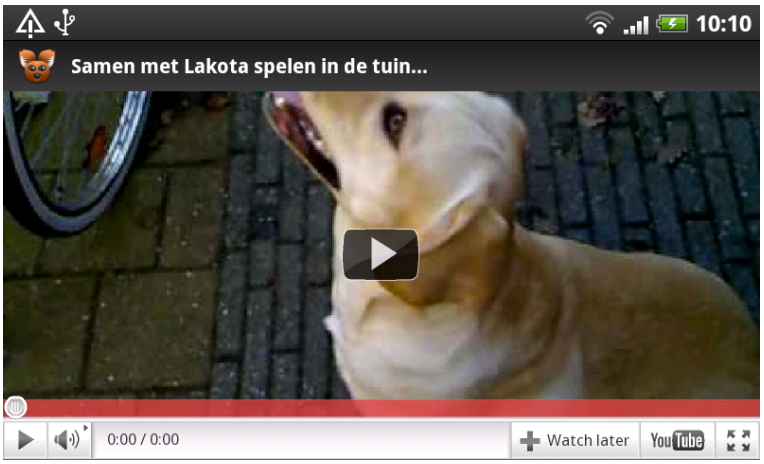
Onderin komen er eventuele hints in tekstvorm. Deze geven extra tips om de zin te begrijpen. De tips kunnen verborgen worden door ze aan te raken met de vinger. Als de laatste tip nog niet getoond is, komt er een nieuwe tip tevoorschijn.

Vindt men dat een beeld prima de context omschrijft, dan kan met de groene duim aanraken. De groene duim zal dan onder dit beeld verdwijnen en niet meer terugkomen. Het klikken op de groene duim zorgt er voor dat in toekomstige zoekopdrachten de resultaten van deze bron meer voorin in de lijst zullen verschijnen.

4.5.9 Statistieken bekijken

In het systeemmenu kan de gebruiker kiezen voor de optie statistieken. Je kan er de volgende gegevens bekijken:

- Totaal aantal opzoekingen
- Aantal teruggegeven multimediamateriaal
- Totaal aantal geaccepteerde afbeeldingen en filmpjes



Figuur 4.8: Weergave van een YouTube filmpje.

- Totaal aantal resultaten voor elke bron
- Totaal aantal acceptaties voor elke bron

4.6 Opbouw van de schermen

Een Android-scherm dient steeds afgeleid te zijn van de klasse *Activity*, maar omdat vormgevingsinstellingen (bijvoorbeeld de grotere titel met logo wat in elk scherm terugkomt) niet bij elk scherm opnieuw ingesteld zouden moeten worden, werd gebruikt gemaakt van een tussenliggende *CustomWindow*-klasse.

Instellingenschermen worden normaal afgeleid van *PreferenceActivity*, maar om dezelfde reden worden deze in de communicatiehulp afgeleid van *CustomWindowPreferences*. Daarom zijn ook de ListView activities afgeleid van de *CustomWindowListView* klasse.

4.7 Plugin structuur van bronnen, filters en hints

Zinnen die in de applicatie ingevoerd worden, worden eerst door één of meerdere filters gestuurd om de context van een zin te bepalen. Kernwoorden die

uit de filter komen, worden naar verschillende bronnen gestuurd. Die bronnen geven dan afbeeldingen en films terug die in de beschrijving zoektermen bevatten waar we naar op zoek zijn. Hintklassen kunnen hints genereren die extra informatie verschaffen over bepaalde woorden. Extra informatie tonen kan bijvoorbeeld inhouden dat een vervoegd werkwoord getoond wordt, samen met de infinitief of extra uitleg over een bepaald woord.

De structuur van de filters, bronnen en hints is zo ontwikkeld dat deze eenvoudig uitgebreid kunnen worden. Je kan bronnen, filters of hints toevoegen zonder bekend te zijn met de volledige programmacode. Zo kunnen uitbreidingen zeer snel en eenvoudig gebeuren.

Elke Filter en bron is een aparte klasse die afgeleid is van een abstracte implementatie die reeds gemeenschappelijke code bevat. Samen met de interface die deze implementeren worden zo bepaalde vereisten gesteld aan een concrete implementatie van een filter of een bron. Zo moet een bron implementaties voorzien voor *getBasicURL* (om het basis adres in te vullen waar de REST aanvraag aan moet gebeuren), *editTekst* (om aanpassingen te kunnen doorvoeren in het resultaat en zo een string te kunnen teruggeven die geldige JSON data bevat) en *performSearch* (om het uiteindelijke internetadres te kunnen vervolledigen, de aanvraag te doen en de resultaatlijst op te vullen).

Een extra bron of filter toevoegen kan eenvoudig gebeuren door een klasse te creëren die de bron of filter interface (en beter de abstracte klasse) implementeert en een lijn toe te voegen in het plugin.xml-bestand. De plugins vermeld in de xml worden automatisch opgenomen in de instellingen (ook het scherm met de broninstellingen wordt automatisch gegenereerd) en deze worden automatisch geraadpleegd bij een zoekactie.

Een nieuwe hintklasse die bijgemaakt wordt, moet voldoen aan de iHint interface. Alle hints komen ook in de hints package, zodat deze verzameld zijn op een centrale plek.

In de ontwikkelde communicatiehulp wordt in de filters beroep gedaan op verschillende publieke API's, namelijk die van Google Image Search, Flickr, YouTube en Vimeo. Er is telkens voor REST gekozen als technologie om de aanvraag door te voeren en in elk geval is er een JSON antwoord dat verwerkt wordt. Bij Vimeo worden er ook OAuth authenticatiekoppen met de aanvraag doorgestuurd, zoals ook besproken in 3.10.

4.8 Combineren van filters

Filters kunnen op twee manieren gecombineerd worden: *sequentieel* (zodat in elke filter woorden kunnen verdwijnen of bijkomen) of via *combinatie* (waar de woorden, die als resultaat uit beide filters komen, samengevoegd worden).

Natuurlijk kunnen ook deze combinaties terug gecombineerd worden. Om de keuze te maken welke combinaties en welke elementaire filters in de applicatie moeten verschijnen, kan de programmeur lijnen toevoegen en verwijderen in het plugin xml-bestand.

Een voorbeeld van een combinatie van twee filters is te zien in codefragment 4.4.
De *Nouns And Verbs* filter wordt er gecombineerd met de *No Filter* filter. Aangezien de *No Filter* geen woorden weglaat, betekent dit concreet dat het resultaat van de *Nouns And Verbs* filter in dit fictieve voorbeeld weinig bepalend is voor het eindresultaat.

```
public class Combination extends AbstractFilter {
 @SuppressWarnings("unchecked")
 public Set<String> filter(Context context, Set<String>
    input) {
  try {
   return this.concat(context, input, (Class<iFilter>)Class.
      forName("be.hogent.laurens.model.filters.NounsAndVerbs
      "), (Class<iFilter>)Class.forName("be.hogent.laurens.
      model.filters.NoFilter"));
  } catch (ClassNotFoundException e) {}
  return null;
 }
}
```

Codefragment 4.4: Een combinatie van twee filters.

4.9 Het gebruik van de applicatie

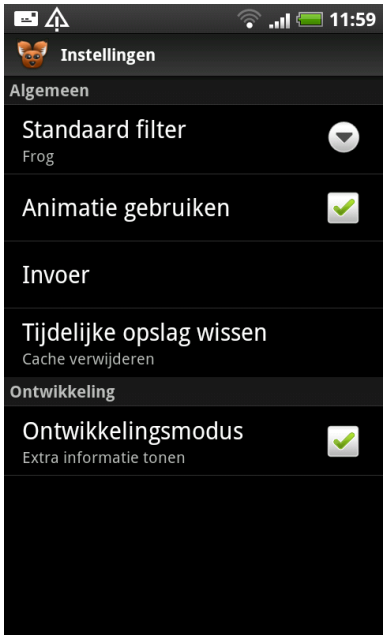
4.9.1 Gebruikersinstellingen

De gebruiker kan enkele aanpassingen doorvoeren die het gedrag van de applicatie veranderen (zie afbeelding nummer 4.9).

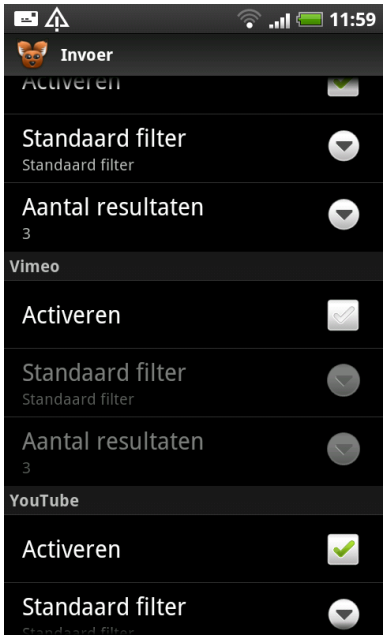
Hij kan de **standaardfilter** kiezen die gebruikt moet worden, al kan deze instelling per bron overschreven worden.

Animaties kunnen in- of uitgeschakeld worden. Als deze ingeschakeld zijn, gaat de overgang tussen de afbeeldingen bij de resultaten iets zachter verlopen. De huidige afbeelding zal eerst zacht wegtrekken waarna de nieuwe afbeelding langzaam zichtbaar wordt. Men kan dit uitschakelen om de overgang sneller te laten verlopen.

Het is mogelijk het programma**cache** te **legen**. Normaal zouden afbeeldingen na afloop automatisch moeten verdwijnen, maar dat kan mislopen als het programma niet op de juiste manier is afgesloten.



Figuur 4.9: Instellingen die de gebruiker kan aanpassen.



Figuur 4.10: Het instellen van bronnen.

De gebruiker kan ervoor kiezen om de **ontwikkelmodus** in te schakelen. Bij het gebruik van het programma worden dan extra debugmeldingen weergegeven die handig zijn voor de programmeur.

Via de knop **Invoer** kom je op een tweede scherm (zie afbeelding 4.10) terecht waar je voor elke bron apart instellingen kan wijzigingen. Je kan ervoor kiezen een bron uit te schakelen, de filter voor deze bron aan te passen en eventueel kan de gebruiker het maximum aantal resultaten wijzigingen die de bron kan weergeven. Als geen filter opgegeven wordt, wordt de standaard filter gebruikt.

4.9.2 Gebruikersinteractie doormiddel van vibratie

De applicatie is zo ontwikkeld dat een kleine vibratie te voelen is elke keer de gebruiker een toets aanklikt. Dit gebeurt reeds zo bij touchscreentoetsen onder het scherm bij sommige modellen. Zo weet de gebruiker dat zijn input overgekomen is en wordt vermeden dat een gebruiker meermaals zou klikken en zo onbedoelde acties zou genereren.

4.9.3 Extra hints in tekstvorm

Naast de verklaring met behulp van afbeeldingen en filmpjes worden ook hints in de vorm van tekstboodschappen weergegeven. De boodschappen proberen de zin extra te verduidelijken met uitleg die niet grafisch weergegeven kan worden.

Er kunnen verschillende klassen ontwikkeld worden die zulke hints kunnen genereren. Het is voldoende dat elke klasse aan een interface voldoet, in de hints package geplaatst wordt en een lijn toegevoegd wordt in het plugin.xml-bestand. Zo kunnen dus snel systemen bijgemaakt worden die hints genereren zonder op de hoogte te zijn van de volledige applicatiecode. De klasse krijgt bij aanroep telkens de volledig ingegeven zin mee.

In de afgeleverde applicatie zijn twee systemen ontwikkeld die hints genereren.

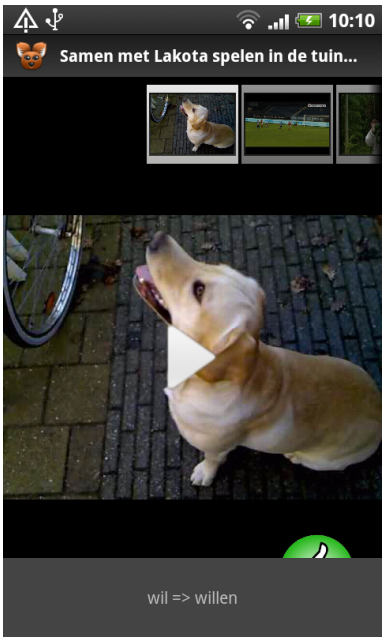
1. De **Dictionary** klasse bevat een woordenboek. Elke lijn bestaat hier uit een sleutel en een uitleg. Als de sleutel in de tekst voorkomt, wordt een hint getoond met de uitleg. In deze klasse kunnen bijvoorbeeld tips geprogrammeerd worden zoals:

   ```
   welke kleur => kleur wat?
   ```

   Dit voorbeeld illustreert hoe een zin in de structuur van de gebarentaal weergegeven zou worden.

2. De **Verbs** klasse gaat de zin doorsturen naar een Frog server (zie ook 3.8.7.1). Met behulp van die analyse weet de klasse welke woorden als werkwoord dienst doen en wordt de infinitief van die werkwoorden opgezocht in het meegeleverde woordenboek. Een hint wordt aangemaakt voor elk vervoegd werkwoord (dat geen infinitief) is in de vorm van:

   ```
   vervoegd werkwoord => infinitief
   ```



Figuur 4.11: Een hint die de infinitief van een werkwoord weergeeft.

> Deze hulp is handig omdat in gebarentaal geen vervoegde werkwoorden voorkomen. Vaak is het voor de dove kinderen dus moeilijk om te achterhalen wat de infinitief is van een werkwoord. In de structuur van gebarentaal wordt *"liep"* eerder weergegeven met gebaren die *"vroeger lopen"* illustreren.

Elke klasse kan meerdere hints genereren en alle hints worden achter elkaar getoond. Een hint verschijnt onderaan in beeld. Klikt de gebruiker op een hint dan gaat dat deel naar beneden en komt het erna terug naar boven met een nieuwe hint. Tenzij alle hints reeds getoond zijn, blijft het hint-gedeelte verborgen.

4.9.4 Volgorde van de resulaten

Onder elke afbeelding en elk filmpje verschijnt in de output een groene duim. Als dat beeld een goede verklaring geeft, dan kan de gebruiker hierop klikken. In de databank met de statistieken wordt vervolgens een teller verhoogd bij de bron die het beeld heeft aangeleverd. De duim verdwijnt en zal bij dit beeld verborgen blijven. Zo kan een beeld slechts eenmalig geaccepteerd worden.

Bij de instellingen van de bronnen is het mogelijk een prioriteit van één tot vijf te geven aan elke bron. Vijf is de hoogste prioriteit.

Deze twee parameters worden als input gebruikt in een formule die de volgorde bepaalt van de resultaten van de bronnen in het output venster. Het aantal acceptaties wordt vermenigvuldigd met de prioriteit. De bron met het hoogste resultaat in deze formule zijn resultaten komen vooraan in de lijst terecht.

4.9.5 Landschapsmodus

De applicatie kan zowel in landschaps- als portretmodus gebruikt worden. Een alternatieve layout is ontwikkeld voor het menu voor als de applicatie in landschapsmodus gebruikt wordt. In landschapsmodus worden horizontaal 3 tot 4 iconen naast elkaar getoond, terwijl dat er in portret modus maximum twee zijn.

Bij het wisselen van modus wordt de huidige activity afgebroken en een nieuwe opgezet. Zonder extra aanpassingen zorgt dit er voor dat berekeningen opnieuw gebeuren en informatie opnieuw opgehaald wordt. Om dit te vermijden wordt bij het sluiten van een activity informatie opgeslaan in de *savedInstanceState*-bundle (zie referentie 3.2). Het wisselen kan hierdoor dus een stuk sneller gebeuren. Naast opgehaalde informatie en berekeningen wordt in de bundle ook status informatie opgeslaan zoals het huidige actieve beeld, de ingevoerde tekst, ..

4.9.6 Pluginfunctionaliteit in Android

Een intent-filter (zie het hoofdstuk over intents: 3.1.1.2) is aan de analyse activity toegevoegd. Er is ingesteld dat dit venster kan verschijnen als een intent opgestart wordt met als actie *SEND* en data van het type *text/plain*.

Dit kan bijvoorbeeld handig zijn als men het standaard sms-programma of standaard internetprogramma op zijn telefoon gebruikt. In het internetprogramma kan men tekst selecteren en dan voor de optie *share* kiezen. In de lijst die vervolgens verschijnt kan men nu voor "Communicatiehulp"kiezen. Maakt men die keuze, dan wordt het analysevenster opgestart. Deze activity kan dan de (geselecteerde) tekst ophalen, er een analyse op uitvoeren en de resultaten weergeven.

Op die manier kan de communicatiehulp dus ook als plugin in Android aanzien worden: elk programma kan tekst rechtstreeks doorsturen naar het pro-

gramma.

4.9.7 Het loggen van activiteiten en resultaten in het programma

Om na de gebruikerstesten te kunnen achterhalen welke acties de testpersonen uitgevoerd hadden, hoelang ze bij elk onderdeel waren blijven staan, welke teksten ze ingevoerd hadden, welke resultaten er waren en welke resultaten ze bekeken hadden was een uitgebreide logging noodzakelijk. Elk testtoestel werd geleverd met een SD geheugenkaartje. Daardoor konden de logs eenvoudig op deze kaarten opgeslaan worden. De singelton helperklasse is gebruikt als centrale plaats voor de logging.

Als de helperklasse aangemaakt wordt, wordt (indien deze nog niet zou bestaan) de juiste map aangemaakt op het externe geheugenkaartje. Dan wordt ook een nieuw loggingbestand gecreëerd en opengezet. Hoe dit kan gebeuren is te zien in codefragment 4.5. De naam van de logbestanden begint met *"log"* en worden bewaard in de map *Communicatiehulp*. Het getal 100 bij het aanmaken van een nieuwe FileHandler betekent dat er nooit meer dan 100 logbestanden zullen zijn, oude bestanden worden dan gewist wanneer er nieuwe gecreëerd worden.

```
File sdCard = Environment.getExternalStorageDirectory();
File dir = new File (sdCard.getAbsolutePath() +
   "/Communicatiehulp");
dir.mkdirs();
File file = new File(dir, "log.log");
handler = new FileHandler(file.getAbsolutePath(), 0, 100);
Logger logger = Logger.getLogger("Communicatiehulp");
logger.addHandler(handler);
```

Codefragment 4.5: Aanmaken van een nieuwe Logger die zijn logs naar een bestand wegschrijft.

In codefragment 4.6 is te zien hoe de nieuwe logger vervolgens gebruikt kan worden om logs weg te schrijven. Om ervoor te zorgen dat de applicatie niet blokkeert terwijl schrijfbewerkingen uitgevoerd worden wordt dit best gedaan in een thread die niet instaat voor de grafische weergave.

```
if (logger == null)
 return;
new Thread(new Runnable() {
 public void run() {
```

```
   logger.log(Level.INFO, message);
  }
 }).start();
```

Codefragment 4.6: De nieuwe Logger gebruiken om te loggen.

In fragment 4.7 is te zien hoe zo'n log entry er in xml kan uitzien. Met de eigenschappen '*date*' en '*milis*' kan het tijdstip gelezen worden van de log. De *sequence* eigenschap geeft aan hoeveel logs er voor kwamen. *Logger* geeft de string weer die opgegeven werd bij de constructie van de Logger. De ernst van de log kan je in de eigenschap "*level*"vinden. De klasse die de log geschreven heeft[1] is genoteerd en de *method*e die de log geschreven heeft vinden we terug in de *method*-eigenschap van de entry[2]. De identificatie van de *thread* staat in een eigenschap en de string zelf is terug te vinden in de *message* eigenschap.

```
<record>
 <date>May 19, 2011 10:25:32 AM</date>
 <millis>1305800732213</millis>
 <sequence>20</sequence>
 <logger>Communicatiehulp</logger>
 <level>INFO</level>
 <class>be.hogent.laurens.model.Helper\$2</class>
 <method>run</method>
 <thread>19</thread>
 <message>Wisselen naar afbeelding nummer 1</message>
</record>
```

Codefragment 4.7: Een voorbeeld van een log entry.

[1]Bij de communicatiehulp zal dit steeds de Helper klasse zijn. Het nummer twee slaat op het nummer van de thread.

[2]Als de log geschreven wordt vanuit een nieuwe thread zal die methode logischerwijze steeds *run* zijn.

Hoofdstuk 5

User experience

5.1 Soorten testen

De bruikbaarheid van de applicatie of van bepaalde concepten kan op verschillende manieren worden uitgetest. Dit kan zowel via *user experience*, aan de hand van een *sketchbook* of met behulp van de Wizard of Oz-techniek. Is de communicatiehulp een bruikbaar hulpmiddel om de communicatie tussen dove en slechthorende kinderen en hun omgeving te ondersteunen? Een gebruikerstest geeft antwoord.

5.1.1 Sketchbook

Er wordt een fictief verhaal verzonnen met gebeurtenissen die een mogelijke gebruiker kan beleven. Daarnaast worden de handelingen genoteerd die de gebruiker kan uitvoeren in de conceptapplicatie.

5.1.2 Wizard of Oz

Achter de schermen zal een gebarentolk aanwezig zijn die de gebaren van de dove vertaalt en iemand die de kernwoorden van het gesprek ingeeft in de fotozoeker. Het doel ervan is te ontdekken via interview na en observatie tijdens de test, wat er nog aangepast kan worden en of de proefpersoon het als een bruikbaar instrument voor dagelijkse communicatie beschouwt.

5.2 Praktijktest

Samen met Emilie Van Dijck (ondersteund door haar promotor Karin Slegers) werd een praktijktest afgenomen in het BuSO Sint-Gregorius te Gent-

brugge. Voorafgaand (om de test te evalueren) werd de test ook uitgevoerd met normaalhorenden in Leuven.

5.2.1 De toestellen

Vanuit het WiCa-onderzoekscentrum werden we vijf Google Nexus One Android toestellen ter beschikking gesteld. Hierop was Android 2.2 geïnstalleerd. Allen hadden ze een SD-geheugenkaart en geen simkaart.

Op elk toestel werd 'een sms van mama'[1] , Adobe Flash, K9-Mail en de communicatiehulp geplaatst. Een mailaccount werd aangemaakt met daarin 'een e-mail van de politie' en het account werd ingesteld in K9-Mail. In de communicatiehulp werd het maximum aantal resultaten dat Google Image Search mag teruggeven op 8 geplaatst en de prioriteit van die bron werd op het maximum ingesteld.

5.2.2 De presentatie

Aangezien dove kinderen sterk visueel ingesteld zijn, vooral kijken naar de tolk en niet zo snel kunnen lezen, was de PowerPointpresentatie een belangrijke ondersteuning bij de uitleg. De iconen van de poster en de iconen in de applicatie zelf, werden ook in presentatie gebruikt, voor de herkenbaarheid.

Op het projectiescherm werd de Powerpointpresentatie soms ook ingeruild voor een virtuele gsm. Zo kond de *workflow* voor de kinderen worden geprojecteerd.

Een naamkaartje bij elk kind zorgde ervoor dat het persoonlijk kon worden aangesproken.

5.2.3 Voorbereidende test in Leuven

De testing in Leuven werd uitgevoerd ter voorbereiding en ter inoefening van de 'echte' test met de doven. Er waren vier testpersonen en de test werd gelijkaardig uitgevoerd als de 'echte' test.

In de applicatie werden enkele fouten ontdekt die verbeterd werden. Een afbeelding over huidirritatie die als niet zo fraai werd beoordeeld, werd gefilterd en enkele kleine optimalisaties aan de opdrachten werden gemaakt.

[1]Een simkaart werd in elk van de toestellen geplaatst. Het telefoonnummer ervan werd in het adresboek opgeslagen als *Mama* en een sms werd naar de simkaart gestuurd.

De locatie van de afbeelding over huidirritatie werd manueel op een zwarte lijst in de applicatie geplaatst, zodat die specifieke afbeelding niet meer getoond werd.

Optimalisaties waren er voornamelijk in de volgorde van de woorden van de opdrachtzinnen. Het sms-bericht werd bijvoorbeeld aangepast naar "De zalf ligt op tafel, en de bijsluiter ook. Voor gebruik de bijsluiter lezen! Mama". Zo kon makkelijker apart op 'bijsluiter' gezocht worden.

De applicatie werd over het algemeen als zeer bruikbaar aanzien door de studenten.

5.2.4 Opmeten van de resultaten

De resultaten van de test zijn op verschillende manieren vastgelegd. Er werd een camera geplaatst, er werden notities genomen, de testtoestellen hebben alles gelogd (zie referentie 4.9.7), er waren testen met antwoordformulieren en na elke test was er een soort evaluatie die de kinderen moesten invullen. In die evaluatie stonden drie vragen: ik vind dit handig, ik vind dit leuk en ik zou dit wel kunnen gebruiken. Deze konden telkens beantwoord worden met: helemaal oneens, oneens, ik weet het niet, eens of helemaal eens.

Elke test had een andere manier van tekst invoeren in de applicatie. De bedoeling was om uit de resultaten van die evaluaties conclusies te kunnen maken over welke invoermethoden handig zijn en welke niet. Helaas was dat voor de kinderen niet altijd duidelijk en werden de keuzes meer bepaald door hoe mooi de afbeeldingen waren die ze terugkregen en hoe gemakkelijk ze de vraag vonden.

Zo werden slechte punten gegeven bij een test waar gevoelige foto's over huidirritaties stonden weergegeven en werden er goede punten gegeven als de vraag in korte tijd opgelost was.

5.2.5 De testpersonen

Niet alle testpersonen zijn gebleven tot de laatste test, een meisje is namelijk niet tot het einde kunnen blijven omdat ze in een andere les moest zijn. Er waren 4 kinderen met een leeftijd tussen 13 en 21 jaar. Er was één meisje en drie jongens. Slechts één jongen had een hoorapparaat.

5.2.5.1 Communicatie met de kinderen

Omdat de kinderen niet konden horen, onvoldoende konden liplezen en zich via spraak niet verstaanbaar konden maken, werd alles wat wij vertelden en wat de kinderen wilden vragen vertolkt met een tolk.

5.2.5.2 Gsm gebruik

Slechts één van de testpersonen had geen gsm-toestel. De andere drie hadden allemaal een telefoon van het merk Nokia, waarvan geen enkele met een touchscreen was. Ze gebruiken ook geen internet op de toestellen. Op het eerste zicht lijkt het alsof de kinderen weinig interesse hebben in gsm-toestellen.

Ze gebruiken de toestellen voornamelijk om foto's te nemen, een berichtje te sturen als iemand jarig is en het bekijken van filmpjes. Maar op school en in de leefgroepen mag deze niet gebruikt worden.

5.2.5.3 Bruikbaarheid van de communicatiehulp volgens de testpersonen

Voorafgaand aan de testen leek hen voornamelijk de spraakherkenning het meest bruikbare. Als horende mensen te snel tegen hen praten, zouden ze dit rustiger kunnen nalezen op het toestel. Deze functionaliteit is echter niet getest.

Vermoedelijk was het voor hen voor de testen nog niet duidelijk wat de applicatie allemaal kan doen. Zo kwamen er vragen zoals *"gebruikt het toestel veel elektriciteit?"* en *"geeft het toestel licht in het donker?"*.

De leerlingen waren alleszins bijzonder benieuwd naar de toestellen. Van zodra ze uitgedeeld waren, grepen ze deze vast en was het moeilijk terug hun aandacht te krijgen.

Tijdens de testen waren ze over het algemeen positief over het programma. Ze vinden het aangenaam om mee te werken en zien er de bruikbaarheid wel van in.

De kostprijs van een Android-toestel, samen met de kostprijs van een mobiel abonnement schrikten de kinderen wel af. Ze denken daarom dat ze het toestel pas zouden kunnen aankopen als ze achttien jaar oud zijn. De jongeren schatten zo'n twee weken om de applicatie aan te leren, al lijkt ons dat wel veel.

5.2.6 De begeleiders

Er zijn korte interviews gebeurd met een logopediste-gebarentolk, een ICT-verantwoordelijke, een dove leerkracht en een dove stagiaire. De conclusies van deze gesprekken zijn verwerkt in de scriptie.

5.2.7 De testen

Emilie had verschillende testen voorbereid. Bij elke test moesten de kinderen tekst ingeven op een andere (vooropgestelde) manier en zo de betekenis van een woord proberen te achterhalen. Om zeker te zijn dat ze de mogelijke antwoorden van de multiplechoicevragen begrepen, werden deze antwoorden ook als afbeeldingen voorgesteld.

Er waren telkens twee kolommen. De kinderen moesten in de eerste kolom voor elke vraag aanduiden wat ze dachten dat het antwoord op de vraag was en na het onderdeel de tweede kolom.

5.2.7.1 Test 1

Alle jongeren kregen een blad met daarop een stripverhaal. Hierin stond het woord *juwelenkistje*. Dat woord moesten ze via het aanraaktoetsenbord van het toestel ingeven en de betekenis er van op zoeken.

De jongeren vonden het erg gemakkelijk om een woord in te geven en op te zoeken. Soms ging het bij hen zelfs vlugger dan bij de (universitaire) testpersonen in Leuven. We hebben weinig moeten bijspringen om hen te helpen.

5.2.7.2 Test 2

De jongeren werd uitgelegd hoe ze via de communicatiehulp een e-mail konden lezen van de politie. In deze fictieve mail stond het woord *huidirritatie* wat de jongeren moesten opzoeken. De test is enigszins mislukt, omdat de tolk het woord reeds vertolkt had in gebarentaal en dus zo de betekenis er reeds van duidelijk gemaakt had.

5.2.7.3 Test 3

Met behulp van de virtuele Android omgeving die geprojecteerd werd, werden de kinderen begeleid naar een sms-bericht van mama. Ze heeft namelijk zalf gekocht en wil kenbaar maken dat eerst de bijsluiter gelezen moet worden. *Bijsluiter* was dus meteen ook het woord wat de jongeren moesten opzoeken.

In de mogelijke antwoorden stond een foto van een doktersbriefje, een bijsluiten en een doosje met zalf. Sommige kinderen maakten de link van dokter met doktersbriefje en duidden dus dat aan, zowel voor als na de test. Andere jongeren hadden in de lijst met resultaten naast een bijsluiter ook een doosje gezien met zalf en hebben dat doosje aangeduid.

De mogelijke antwoorden lagen hier dan ook wel erg dicht bij elkaar. Het leek voor de jongeren moeilijk om de juiste verbanden te leggen.

5.2.7.4 Test 4

Een recept werd uitgedeeld aan de kinderen. De jongeren moesten zelf kiezen welke woorden ze wilden opzoeken om zo te bepalen welke actie eerst moest uitgevoerd worden. Het recept moest via tekstherkenning in het toestel ingevoerd worden.

Het was niet altijd gemakkelijk om de jongeren een scherpe foto te laten maken. Het was soms een relatief lange tijd wachten op resultaat (tot een tekstherkenningsresultaat terugkwam) en er kwam veel tekst als resultaat terug. *Zaadlijsten* was het woord wat ze eigenlijk moesten opzoeken, maar ze kwamen daar niet altijd toe. De test was misschien beter uitgevoerd met iets minder tekst.

5.2.7.5 Test 5

In de laatste test moesten de kinderen via de ingebouwde browser in de zoekmachine van Google enkele termen invoeren, een nieuwsartikel zoeken, lezen en aanduiden wat er in het artikel staat.

Wegens tijdsgebrek hadden we niet meer zoveel tijd voor de test en die was misschien ook niet geschikt voor deze doelgroep, dus hebben we hen gewoon getoond hoe deze functionaliteit werkt, zonder antwoordformulier.

Besluit

Over het algemeen wordt goed op de ontwikkeling van de applicatie gereageerd. Voor doven zijn nog niet zo veel verduidelijkende programma's uitgebracht en men is zeer opgetogen dat er nagedacht wordt over oplossingen voor de problemen van de dove populatie. Het BuSO Sint-Gregorius hoopt alvast op een vervolg van het onderzoek.

Om het nut van de applicatie te vergroten, zouden minder irrelevante beelden in de resultaten moeten verschijnen. Aangezien de kinderen geen idee hebben van de betekenis van een woord, kunnen ze ook moeilijk oordelen welke afbeelding nu correct de context van een zin weerspiegelt.

Een soort databank aanleggen waar voor elk woord zo'n twee à drie afbeeldingen per woord gekozen wordt, zou een oplossing kunnen zijn. Daarmee wordt echter het aantal mogelijke opzoekingen beperkt. Een leerkracht zou een dergelijke databank wel kunnen aanvullen om zo de applicatie te gebruiken ter ondersteuning van de les.

Verschillende types uit de BuSO indeling zouden de applicatie kunnen gebruiken:

- De jongeren uit opleidingsvorm 1 (**OV1**) kunnen de applicatie gebruiken om bepaalde hiaten in hun woordenschat aan te vullen. Vaak zijn dat namelijk nog vrij concrete woorden. Deze jongeren krijgen een sociale vorming als voorbereiding tot wonen in een beschermde leefomgeving.
- De groep **OV2** heeft meer taalgevoel, maar heeft meer problemen met iets abstractere woorden. Toch wordt meer rekening gehouden met bepaalde woordsoorten. Deze jongeren krijgen een sociale vorming en arbeidsvoorbereiding tot werken in een beschutte werkplaats en beschermd of begeleid wonen.
- De e-mail- en sms-functie zou vooral bij de **OV3** groep nuttig zijn, omdat de teksten die zij ontvangen meestal van betere kwaliteit zijn. Deze jongeren krijgen een sociale- en beroepsvorming voor de gewone leefomgeving

en inschakeling in het gewone arbeidscircuit. In teksten die jongeren naar elkaar sturen, is het soms lang zoeken naar de eerste klinker. Veel mensen van deze groep zoeken echter eerder een woord rechtstreeks op in een woordenboek.

De kostprijs van de toestellen en van een abonnement voor mobiele bandbreedte zijn voorlopig nog te hoog om aan de doelgroep voldoende toestellen te leveren.

De broncode zou eventueel na de scriptie vrijgegeven kunnen worden in een Open Source-versie zodat anderen dit onderzoek verder kunnen zetten. De problematiek zou verder in de aandacht komen indien de resultaten uit Emilie's onderzoek worden gepubliceerd.

De applicatie is volgens mij een goed begin, maar biedt nog niet in alle gevallen een goede verklaring voor de ingegeven zinnen. irrelevante afbeeldingen en filmpjes zouden nog meer uitgefilterd moeten worden.

Lijst met afkortingen

API	Application programming interface
AT	Assistive Technology
CI	Cochleair Inplantaat
IBBT	Interdisciplinair Instituut voor Breedband Technologie
ICR	Intelligent Character Recognition
JSON	JavaScript Object Notation
KIDS	Koninklijk Instituut voor Doven en Spraakgestoorden
OCR	Optical Character Recognition
OSI	Open Systems Interconnection
OV	Opleidingsvorm
QoE	Quality of Experience
QoS	Quality of Service
REST	Representational State Transfer
RPC	Remote procedure call
RSS	Really Simple Syndication
SDK	Software Development Kit
SOAP	aanvankelijk een afkorting voor Simple Object Access Protocol
SQL	Structured Query Language
URI	Uniform Resource Identifier
VoIP	Voice over IP

WiCa Wireless & Cable

XML Extensible Markup Language

Lijst van figuren

2.1 Abstracte voorstelling van de communicatiehulp. 7
2.2 Verwerking van de input. 9
2.3 Navigeerbare verzameling resultaten. 9
2.4 Een PocketPC met de Mobile Communicator van Oralys Inc. . 20

3.1 Google Android Levenscyclus van een activity. 24
3.2 De applicatie is aan het wachten tot Wisetrend de afbeelding verwerkt heeft. 38
3.3 Spraakherkenning in de applicatie. 42
3.4 Resultaat van de spraakherkenning. 42

4.1 Overzicht van de packages in de applicatie. 64
4.2 Vereenvoudigde voorstelling van het klassendiagram. 71
4.3 Menu van de applicatie. 75
4.4 Invoer via het toetsenbord. 75
4.5 Weergave van het resultaat van de tekstherkenning. 76
4.6 Selectie op een internetpagina. 77
4.7 Zinsdeel selectie in de applicatie. 77
4.8 Weergave van een YouTube filmpje. 79
4.9 Instellingen die de gebruiker kan aanpassen. 82
4.10 Het instellen van bronnen. 82
4.11 Een hint die de infinitief van een werkwoord weergeeft. 84

Lijst van codefragmenten

3.1 Alle mogelijke hooks voor een activity. 23
3.2 Het gebruik van een intent om een nieuwe activity op te starten. 26
3.3 Het gebruik van intents in het AndroidManifest.xml-bestand. . 27
3.4 Alle permissions voor de communicatiehulp. 28
3.5 De applicatie zelf rechten laten declareren. 29
3.6 Het vullen van een savedInstanceState bundel. 31
3.7 Het uitlezen van een savedInstanceState bundel. 31
3.8 Het aanvullen van een internetadres met extra parameters. . . 31
3.9 De internetlocatie van een afbeelding wordt verkregen na het uploaden van een afbeelding. 39
3.10 Voorbeeld output van het Harvester programma. 52
3.11 Fragment van een *dictionarie* van het OpenTaal-project. . . . 53
3.12 Het onderzoeken van het gedrag van de Frog server software met behulp van telnet. 58
3.13 Frog aanspreken vanuit Java. 59
4.1 Voorbeeld van een resource bestand: colors.xml. 64
4.2 Voorbeeld van een layout bestand: ocr_result.xml. 65
4.3 Het ondertekenen van een Vimeo API request met de oauth-signpost bibliotheek. 73
4.4 Een combinatie van twee filters. 81
4.5 Aanmaken van een nieuwe Logger die zijn logs naar een bestand wegschrijft. 86
4.6 De nieuwe Logger gebruiken om te loggen. 86
4.7 Een voorbeeld van een log entry. 87
A.1 Contactpersoon bij KIDS. 100
A.2 Contactpersoon bij het GR@SP project. 100
A.3 Contactpersoon bij het MPI Sint-Lievenspoort. 101
A.4 Contactpersoon bij het BuSO Sint-Gregorius. 101
A.5 Contactpersoon bij het het OpenTaal-project. 101
C.1 Inhoud van omzetter.java. 104

Bijlage A

Contactpersonen

Stefan Coenen
Verantwoordelijke voor onder andere externe hulpmiddelen op school en thuis bij de kinderen
stefan.coenen@kids.be

KIDS
Borggravevijversstraat 9
3500 Hasselt

Contactgegevens A.1: Contactpersoon bij KIDS.

Karin Slegers
Senior Researcher
Centre for User Experience Research (CUO)
Katholieke Universiteit Leuven / IBBT
karin.slegers@soc.kuleuven.be

Pieter Duysburgh
Onderzoeker aan het IBBT
SMIT/Vrije Universiteit Brussel
pieter.duysburgh@vub.ac.be

Contactgegevens A.2: Contactpersoon bij het GR@SP project.

```
Trui Van Huylenbrouck
t.vanhuylenbrouck@slp-gent.be

Emelie Vanwynsberghe
e.vanwynsberghe@slp-gent.be

MPI Sint-Lievenspoort
Keizervest 18
9000 Gent
```

Contactgegevens A.3: Contactpersoon bij het MPI Sint-Lievenspoort.

```
Griet Geysels
Adjunct Directeur BuSO (GON)
bsg.gon.grietgeysels@gmail.com

BuSO Sint-Gregorius
Jules Destr elaan 67
9050 Gentbrugge
```

Contactgegevens A.4: Contactpersoon bij het BuSO Sint-Gregorius.

```
Ruud Baars
baarsrj@xs4all.nl
```

Contactgegevens A.5: Contactpersoon bij het het OpenTaal-project.

Bijlage B

Poster

COMMUNICATIEHULP VOOR DOVE KINDEREN OP GOOGLE ANDROID

Kinderen en jongeren die doof of slecht horend zijn beheersen de gesproken en geschreven taal niet zo goed. Bovendien spreken hun vrienden en familieleden niet altijd gebarentaal.

De communicatiehulp probeert deze hindernissen te reduceren door tekst om te zetten naar relevante plaatjes en filmpjes. Deze worden gevonden op online bronnen zoals Flickr, YouTube, Google Image Search, Vimeo, ..

UITGEVOERD DOOR

Laurens Van Acker
student aan de HoGent
laurens@van.acker.be

Veerle Ongenae
interne promotor

Wout Joseph
externe promotor

Luc Martens
externe promotor

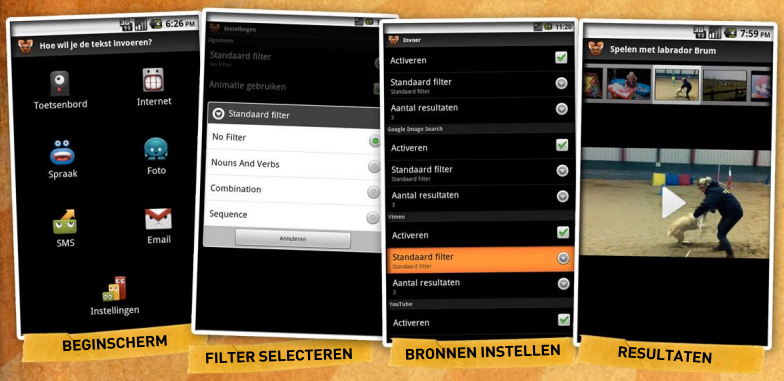
Adrian Juan
begeleider

Toon De Pessemier
begeleider

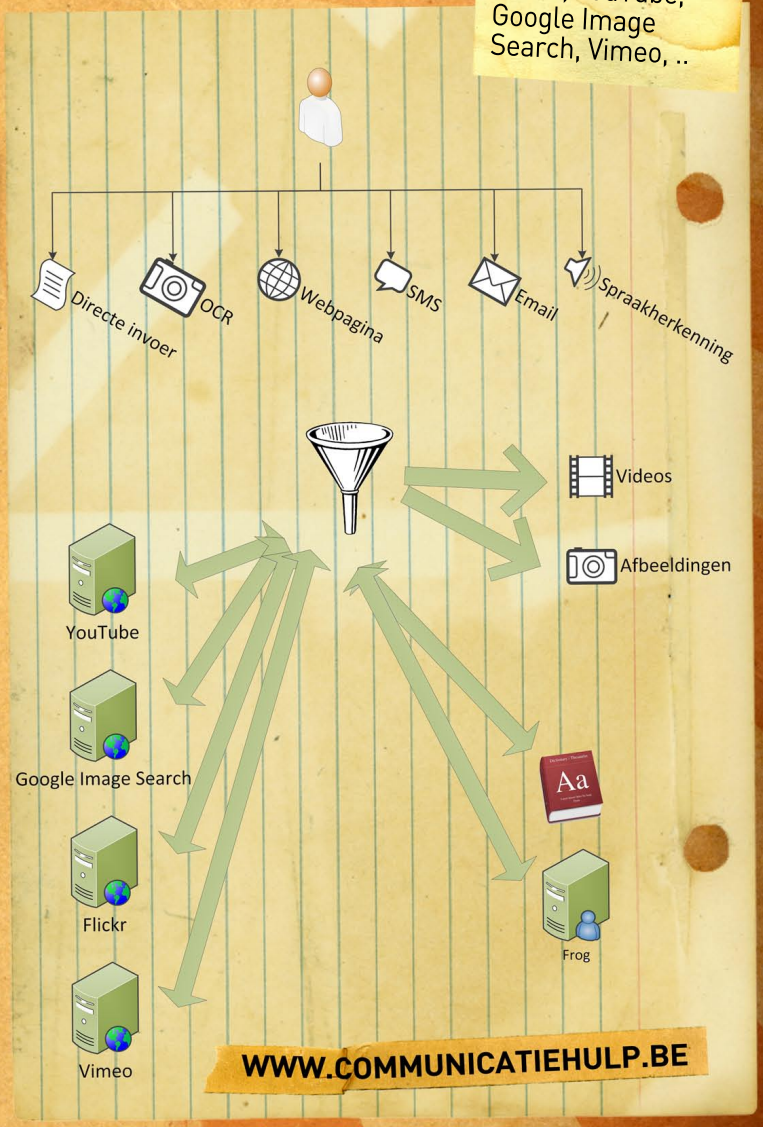
Kris Vanhecke
begeleider

WORKFLOW

De gebruiker kan op verschillende manieren gegevens in de applicatie invoeren: via directe invoer, door het maken van een selectie uit een sms of e-mailbericht of uit een webpagina of doormiddel van tekst- of spraakherkenning. Hierna zal een filter de kernwoorden uit de zin trachten te halen door onder andere gebruik te maken van intelligente software die woordsoorten en andere kenmerken aanduidt, een woordenboek en frequentietabellen. Hierna wordt een verzameling multimediabestanden weergegeven die de context van de zin proberen te verduidelijken.



BEGINSCHERM | FILTER SELECTEREN | BRONNEN INSTELLEN | RESULTATEN



PRAKTIJK

De use cases voor de vooropgestelde applicatie zijn opgesteld in samenspraak met begeleiders van dovenstellingen. Samen met Emilie Van Dijck, student aan de KU Leuven, met een vooropleiding als logopedist-audioloog zullen in een later stadium gebruikersevaluaties afgenomen worden. Zo kan de applicatie geëvalueerd worden aan de doelgroep. Hopelijk kunnen deze resultaten als inspiratie helpen bij toekomstige vergelijkbare projecten.

TECHNOLOGIE

Google Android is een opensourceplatform voor mobiele telefoons. Dit platform is gebaseerd op de Linux-kernel en het Java-programmeerplatform. Android is op dit moment het meest verkochte besturingssysteem op mobiele telefoons. Applicaties kunnen op het platform gemakkelijk verspreid worden via de Android Market.

INTEC | Wireless & Cable research group | HOGESCHOOL GENT LID VAN DE ASSOCIATIE UNIVERSITEIT GENT

Bijlage C

Broncode OpenTaal naar SQL

```
import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

// Programma als volgt uitvoeren; java -cp .:sqlitejdbc-v056
    .jar Omzetter

public final class Omzetter {

  public static void main(String[] args) {
    try {
          // Open the file that is the first
          // command line parameter
          FileInputStream fstream = new FileInputStream("
              dutch.txt");
          // Get the object of DataInputStream
          DataInputStream in = new DataInputStream(fstream);
            BufferedReader br = new BufferedReader(new
                InputStreamReader(in,"UTF-8"));
          String strLine;

          // Verbinding maken met de databank om de
              resultaten te kunnen in wegschrijven
            Class.forName("org.sqlite.JDBC");
            Connection conn = DriverManager.getConnection("
                jdbc:sqlite:database.db3");
            PreparedStatement prep = conn.prepareStatement("
                INSERT␣INTO␣dutch␣('flexibleForm','basic','
```

```
                tag')␣VALUES␣(?,?,?);");


            //Read File Line By Line
            while ((strLine = br.readLine()) != null)   {
              String[] delen = strLine.split("␣␣");
              // Print the content on the console
//              System.out.println ("INSERT INTO dutch ('
    flexibleForm','basic','tag') VALUES (\"" + delen[0] +
    "\",\"" + delen[1] + "\",\"" + delen[2] + "\");");
                prep.setString(1, delen[0]);
                prep.setString(2, delen[1]);
                prep.setString(3, delen[2]);
                prep.addBatch();
            }
            //Close the input stream
            in.close();

            // Close database connection after execution
              conn.setAutoCommit(false);
              prep.executeBatch();
              conn.setAutoCommit(true);

              conn.close();

          } catch (Exception e){
              //Catch exception if any
              System.err.println("Error:␣" + e.getMessage());
    }
  }

}
```

Codefragment C.1: Inhoud van omzetter.java.

Literatuurlijst

[1] Meier R. (2010). *Professional Android 2 Application Development.* Indianapolis: Wiley Publishing, Inc.

[2] Slegers K. *(Blog over dove kinderen in het GR@SP project).*, Geraadpleegd op 10 November 2010 via http://deafkidsingrasp.blogspot.com.

[3] *Android Developers.* Geraadpleegd op 19 april 2011 via http://developer.android.com.

[4] Draft scenario: *Photo conversation.* (e-mail naar K. Slegers), [online]. Beschikbare e-mail: karin.slegers@soc.kuleuven.be.

[5] Broekaert E. ,vav Hove G. (2007) *Handboek bijzondere Orthopedagogiek.* Antwerpen: Garant.

[6] *Oralys Inc. Solution: Talk: Mobile Communicator.* Geraadpleegd op 21 april 2011 via http://www.oralys.ca/en/solutions/talk/mobile_communicator.htm.

[7] Van den Bosch A., Busser G.J., Daelemans W., Canisius, S. *An efficient memory-based morphosyntactic tagger and parser for Dutch.* in Selected Papers of the 17th Computational Linguistics in the Netherlands Meeting. Leuven: België, 2007, p. 99-114.

[8] *ABBYY Mobile OCR Engine is a Software Development Kit (SDK).* Geraadpleegd op 22 april 2011 via http://www.abbyy.com/mobileocr.

[9] Daelemans W., Helmer S. (red.). *Het Nederlands in taal- en spraaktechnologie: prioriteiten voor basisvoorzieningen.* 2002.

[10] *Application programming interface.*, Geraadpleegd op 20 April 2011 via http://nl.wikipedia.org/wiki/Application_programming_interface.

[11] *Android-telefoons: overzicht van alle Android-smartphones.* Geraadpleegd op 20 april 2011 via http://www.androidplanet.nl/android-telefoons.

[12] *GR@SP - Overzicht projecten.* Geraadpleegd op 20 april 2011 via http://www.ibbt.be/nl/projecten/overzicht-projecten/p/detail/gr-at-sp.

[13] Centre québécois de la déficience auditive. *Consultation sur les technologies de l'information: Résultats de sondage auprès des organismes et des personnes sourdes et malentendantes.* Michel Brière and Médias Adaptés Communications, Montréal, 1995, 15.

[14] *Google Goggles.* Geraadpleegd op 22 april 2011 via http://www.google.com/mobile/goggles.

[15] Dubuisson C., Daigle D.. *Lecture,écriture et surditeé: visions actuelles et nouvelles perspectives* Revue des sciences de léducation XVII (1998), p. 229-234.

[16] *API documentation IQEAPI v1.2.0 documentation.* Geraadpleegd op 22 april 2011 via http://developer.iqengines.com/apidoc/current.

[17] Fortnum H., Marshall D.H., Bamford J.M., Sum- merfield A.Q.. . *Hearing-impaired children in the UK: education setting and communication approach* Deafness and Education International 4 (2002), p. 123-141.

[18] Fadi H.. *Not another code blog: Google Goggles API.* Geraadpleegd op 22 april 2011 via http://notanothercodeblog.blogspot.com/2011/02/google-goggles-api.html.

[19] Hotton M. . *Le comité sur les nouvelles technologies en déficience auditive de lI.R.D.P.Q.: projet de recherche sur les appareils de télécommunications pour les personnes sourdes et malentendantes.* Différences, Magazine de lInstitut de réadaptation en déficience physique de Québec 5 (2004), p. 32-38.

[20] *Wireshark User's Guide.* Geraadpleegd op 22 april 2011 via http://www.wireshark.org/docs/wsug_html_chunked.

[21] T. Johnston. .*W(h)ither the Deaf Community? Population, Genetics, and the Future of Australian Sign Language* American Annals of the Deaf 148 (2004), p. 358-375.

[22] *Google Goggles to Become App Platform.* Geraadpleegd op 22 april 2011 via http://phandroid.com/2010/04/14/google-goggles-to-become-app-platform.

[23] Vincent C., Deaudelin I., Hotton M.. *Pilot on Evaluating Social Participation Following the Use of an Assistive Technology Designed to Facilitate Face-to-Face Communication Between Deaf and Hearing Persons.* Technology and Disability 19 (2007), p. 153-167.

[24] *Google voegt Nederlandse spraakbesturing toe aan Android.* Geraadpleegd op 22 april 2011 via http://tweakers.net/nieuws/70584/google-voegt-nederlandse-spraakbesturing-toe-aan-android.html.

[25] *Welkom bij OpenTaal.* Geraadpleegd op 22 april 2011 via http://www.opentaal.org/.

[26] *Multilingual dictionary, thesaurus, translation.* Geraadpleegd op 22 april 2011 via http://lookwayup.com/free/dictionary.htm.

[27] *On Google's Unofficial Dictonary API.* Geraadpleegd op 23 april 2011 via http://googlesystem.blogspot.com/2009/12/on-googles-unofficial-dictionary-api.html.

[28] Piek V., Bloksma L., Boersma P.. *The Dutch Wordnet.* Amsterdam: University of Amsterdam (1999).

[29] *Thesaurus - Wikipedia.* Geraadpleegd op 23 april 2011 via http://nl.wikipedia.org/wiki/Thesaurus.

[30] *hunspell format of Hunspell dictionaries and affix files.* Geraadpleegd op 23 april 2011 via http://ovh.dl.sourceforge.net/project/hunspell/Hunspell/Documentation/hunspell4.pdf.

[31] *OpenTaal mailinglist.* [online] Ingeschreven via http://opentaal.org/mailinglists.

[32] *WikiWoordenboek:Info - WikiWoordenboek.* Geraadpleegd op 23 april 2011 via http://nl.wiktionary.org/wiki/WikiWoordenboek:Info.

[33] *MediaWiki API.* Geraadpleegd op 23 april 2011 via http://nl.wiktionary.org/w/api.php.

[34] *Gesture recognition.* Geraadpleegd op 23 april 2011 via http://en.wikipedia.org/wiki/Gesture_recognition.

[35] *IT Wizard : Mezzofanti : Augmented reality through text-recognition..* Geraadpleegd op 25 april 2011 via http://www.itwizard.ro/mezzofanti-augmented-reality-through-text-recognition-146.html.

[36] *tesseract-ocr - An OCR Engine that was developed at HP Labs between 1985 and 1995... and now at Google. - Google Project Hosting.* Geraadpleegd op 25 april 2011 via http://code.google.com/p/tesseract-ocr.

[37] *Java OCR — Download Java OCR software for free at SourceForge.net.* Geraadpleegd op 25 april 2011 via http://sourceforge.net/projects/javaocr.

[38] *Understanding User Interface in Android - Part 3: More Views — mobiForge.* Geraadpleegd op 25 april 2011 via http://mobiforge.com/designing/story/understandinguserinterface-androidpart3moreviews.

[39] *rTAKI0329's android : Google Image Search API android.* Geraadpleegd op 25 april 2011 via http://rtaki.blogspot.com/2010/11/googleimage-searchapiandroid_28.html.

[40] *Work in background: ProgressDialog and AsyncTask.* Geraadpleegd op 25 april 2011 via http://itprojects.spb.ru/?p=150&lang=en.

[41] *Android bitmap caching - Stack Overflow.* Geraadpleegd op 25 april 2011 via http://stackoverflow.com/questions/4994496/androidbitmap-caching.

[42] *Get all classes within a package [java] [introspection] [class] [package].* Geraadpleegd op 25 april 2011 via http://snippets.dzone.com/posts/show/4831.

[43] *caching - Android Webview - Completely Clear the Cache - Stack Overflow.* Geraadpleegd op 25 april 2011 via http://stackoverflow.com/questions/2465432/androidwebviewcom-pletelyclearthecache.

[44] *WordNet - Wikipedia.* Geraadpleegd op 25 april 2011 via http://nl.wikipedia.org/wiki/WordNet.

[45] *Using your own SQLite database in Android applications — ReignDesign Blog.* Geraadpleegd op 25 april 2011 via http://www.reigndesign.com/blog/using-your-own-sqlite-database-in-android-applications.

[46] *Part-of-speech tagging.* Geraadpleegd op 25 april 2011 via http://nl.wikipedia.org/wiki/Partofspeech_tagging.

[47] *Frog: Dutch morpho-syntactic analyzer and dependency parser.* Geraadpleegd op 25 april 2011 via http://ilk.uvt.nl/frog.

[48] Van Eynde F *Part Of Speech tagging en lemmatisering van het corpus gesproken Nederlands.* Leuven: Centrum voor Computerlinguïstiek (2004).

[49] Martens L. *Home - Wireless & Cable (WiCa).* Geraadpleegd op 25 april 2011 via http://www.wica.intec.ugent.be.

[50] *Fevlado FAQ.* Geraadpleegd op 25 april 2011 via http://www.fevlado.be/faq.aspx.

[51] *Encoding - Protocol Buffers - Google Code.* Geraadpleegd op 10 mei 2011 via http://code.google.com/intl/nl-NL/apis/protocolbuffers/docs/encoding.html.

[52] *JSON - Wikipedia.* Geraadpleegd op 20 mei 2011 via http://nl.wikipedia.org/wiki/JSON.

[53] *JSON.* Geraadpleegd op 20 mei 2011 via http://json.org.

[54] *OAuth - Hueniverse.* Geraadpleegd op 20 mei 2011 via http://hueniverse.com/oauth.

[55] *OAuth Community site.* Geraadpleegd op 20 mei 2011 via http://oauth.net/.

[56] *How to Use Three-Legged OAuth - O'Reilly Answers.* Geraadpleegd op 20 mei 2011 via http://answers.oreilly.com/topic/1381-how-to-use-three-legged-oauth.

[57] *2-legged OAuth for the OpenSocial REST API - Google's Internet Identity Research.* Geraadpleegd op 20 mei 2011 via http://sites.google.com/site/oauthgoog/2leggedoauth/2opensocialrestapi.

[58] *WiseTREND OCR Cloud 2.0 - Online OCR Help.* Geraadpleegd op 23 mei 2011 via http://www.wisetrend.com/WiseTREND_Online_OCR_API_v2.0.htm.

[59] *OCR: van papier naar scherm — Kennislink.* Geraadpleegd op 24 mei 2011 via http://www.kennislink.nl/publicaties/ocr-van-papier-naar-scherm.

[60] *OCR Introduction*. Geraadpleegd op 24 mei 2011 via http://www.dataid.com/aboutocr.htm.