Copper - geleid je naar het juiste nummer

Thijs Broenink, Timothy Kuhn, Abdullah Rasool

27 juni 2014

1 Voorwoord

Voor het eerstejaarsvak Research & Development kregen we de opdracht om een Android applicatie te ontwikkelen. In dit eindverslag documenteren we de uiteindelijke app. We gaan uitleggen waarom we bepaalde keuzes hebben gemaakt bij het ontwikkelen, laten tussendoor screenshots zien van de applicatie en tenslotte zullen we terugkoppelen of het aan onze verwachtingen heeft voldaan.

2 Beschrijving

2.1 Inleiding

De app waar het om draait is een customer support app die de naam Copper draagt. Copper biedt contactinformatie van winkels van een aantal grote steden in Nederland en van bedrijven (merken) met als doel snel contact op te kunnen nemen met een winkel of het bedrijf. Dit kan zijn om een vraag te stellen over een bepaald product, maar ook om bijvoorbeeld een product te (laten) repareren. Momenteel bevat Copper nog enkel adressen, telefoonnummers en locaties; in de toekomst komen hier nog andere vormen van contactinformatie bij zoals e-mail, maar ook verschillende sociale media als Twitter en Facebook.

De eerste letters van het woord 'customer' in 'customer support' zijn 'cu'. In het periodiek systeem staan deze letters voor het element koper, wat 'copper' in het Engels betekent. Koper heeft een zeer groot geleidingsvermogen. Dit geleidingsvermogen staat symbool voor het doel van de app: het goede contact tussen klanten en bedrijven.

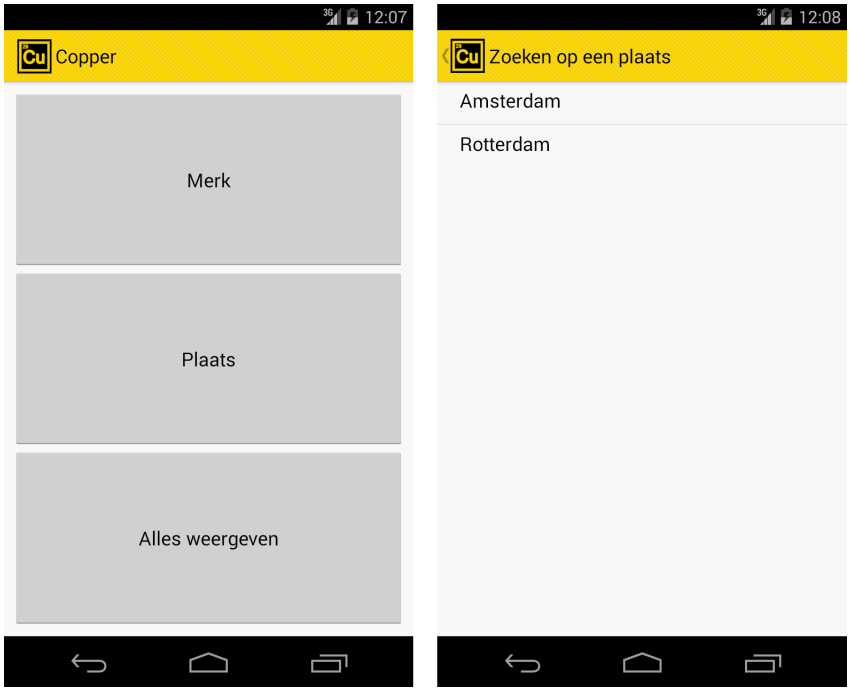
2.2 Productverantwoording

De reden voor het ontwikkelen van deze app was omdat er voor Nederland nog niet zoiets dergelijks bestond. Voor de Verenigde Staten en het Verenigd Koninkrijk bestaat wel een app die een soortgelijke functie heeft als die van ons met de naam GetHuman. GetHuman bestaat, net als Copper, uit een enorme lijst met bedrijven. Bij elk bedrijf staat een telefoonnummer en eventueel een e-mailadres en reviews. Het nadeel van deze app is dat er enkel Amerikaanse bedrijven in staan en dus ook Amerikaanse telefoonnummers. Dit maakt het voor Nederlanders niet erg geschikt.

Wat onze app uniek maakt is dat hij is gericht op Nederland en hun Nederlandse gebruikers. Onze app bevat contactgegevens van Nederlandse bedrijven

en multinationals met Nederlandse vestigingen/afdelingen. Alle contactgegevens in de app zijn afgestemd op de gebruikersdoelgroep en bevatten dus enkel Nederlandse adressen en telefoonnummers.

De grootste toevoeging aan het huidige app-aanbod begint al bij het feit dat er nu een app is die bruikbaar is voor Nederlanders. Als gebruikers een bepaald product hebben gekocht en daarover vragen, opmerkingen of klachten hebben, kunnen ze gemakkelijk binnen een paar tikken aan contactgegevens komen van het bedrijf of de winkel die ze willen bereiken. Men hoeft niet langer het web af te struinen naar correcte en actuele contactgegevens.



Figuur 1: Het hoofdscherm en de lijst van steden

2.3 Specificaties

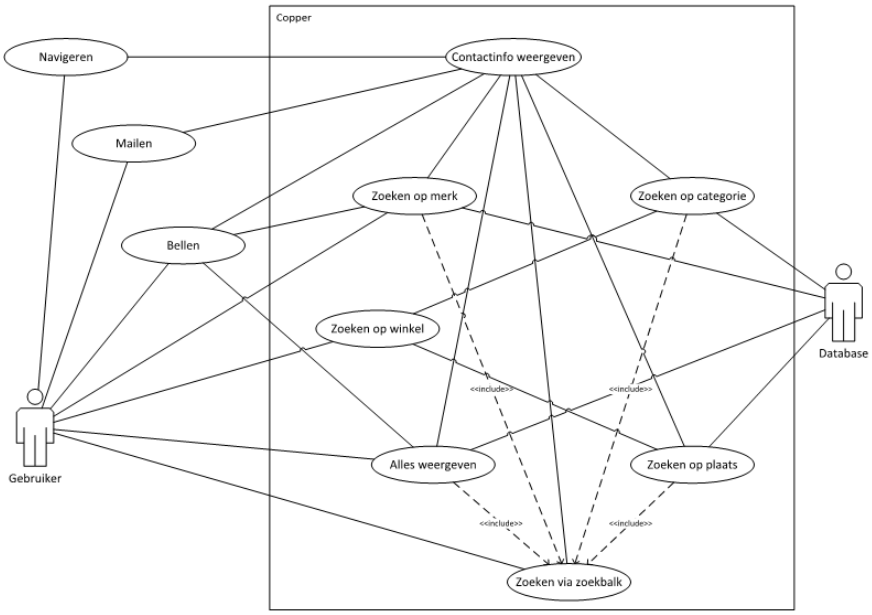
Functionele eigenschappen
Zoals gezegd is het met Copper mogelijk om te zoeken naar merken en winkels. Bij het zoeken naar winkels kan onderscheid gemaakt worden tussen het zoeken op categorie en het zoeken op plaats. Zodra er gekozen is voor een zoekmethode wordt er een lijst met resultaten weergegeven. De gebruiker kan ervoor kiezen om handmatig te zoeken door naar boven en beneden te scrollen. Daarnaast kan er ook gezocht worden door middel van de zoekbalk, wat dynamisch resultaten oplevert. Vanuit de lijst kan de gebruiker direct bellen naar een van de winkels (of merken). Daarnaast is het ook mogelijk om op een van de resultaten te klikken waarna er een pagina verschijnt met contactgegevens. Op die pagina staat de naam van het merk of de winkel, een telefoonnummer, een adres (inclusief Google Maps kaart) en eventueel een doorverwijzing om te mailen. Vanuit

dit scherm kan de gebruiker niet alleen bellen door het nummer aan te klikken, maar ook de locatie openen in Google Maps en eventueel mailen.

De hierboven beschreven functies hebben we grafisch weergegeven in een use case diagram in figuur 2. Op pagina 4 is de Use Case "Zoeken in zoekbalk" uitgewerkt.

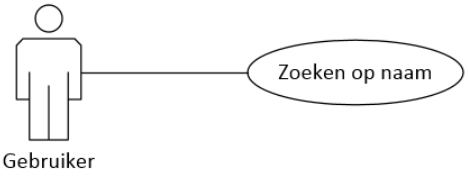
Niet-functionele eigenschappen

- Schaalbaarheid
- Beschikbaarheid
- Bruikbaarheid
- Efficiëntie



Figuur 2: Use Case model van Copper.

Uitwerking Use Case 01 - Zoeken via de zoekbalk

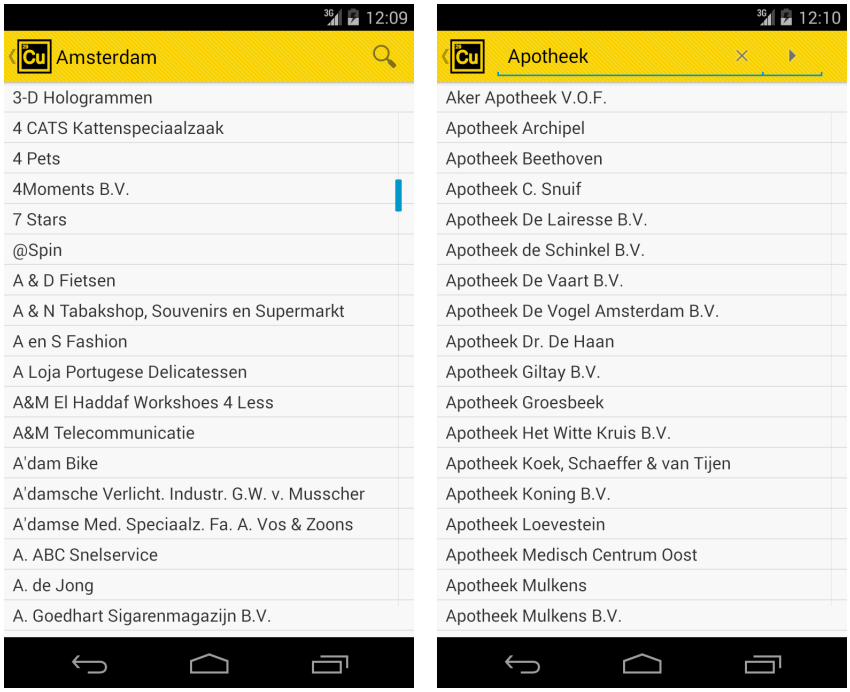
Use Case Diagram	
Beschrijving	Een gebruiker zoekt naar een winkel op naam.
Trigger	De gebruiker wilt contactgegevens van een bepaalde winkel opzoeken
Actor	De gebruiker
Versie	1.0
Basic Course of Events	1. De gebruiker drukt op het app-icoon 2. De app start op en toont 3 knoppen: 'Merk'; 'Winkel'; 'Alles weergeven'. 3. De gebruiker drukt op de knop 'Alles weergeven'. 4. De app toont een laadscherm en vervolgens een lijst met winkelnamen. 5. De gebruiker drukt op het zoekicoon. 6. De app toont een zoekbalk en een toetsenbord. 7. De gebruiker typt zijn zoekterm. 8. De app toont het resultaat/de resultaten.
Alternatieve paden	*Zoeken per stad* 3.1 De gebruiker drukt op de knop 'Winkel'. 3.2 De app toont twee knoppen: 'Plaats'; 'Categorie'. 3.3 De gebruiker drukt op de knop 'Plaats'. 3.4 De app toont een laadscherm en vervolgens een lijst met steden. 3.5 De gebruiker kiest een stad. *Zoeken per categorie* 3.1 De gebruiker drukt op de knop 'Winkel'. 3.2 De app toont twee knoppen: 'Plaats'; 'Categorie'. 3.3 De gebruiker drukt op de knop 'Plaats'. 3.4 De app toont een laadscherm en vervolgens een lijst met categorieën. 3.5 De gebruiker kiest een categorie.
Uitzonderingspad	*De database functioneert niet naar behoren* 4.1 De app toont een foutmelding en verzoekt de gebruiker om het later te proberen.
Precondities	Het apparaat waarop de gebruiker de app draait is verbonden met internet.
Postcondities	De app toont de winkel waar de gebruiker naar opzoek was.

3 Ontwerp

3.1 Globaal ontwerp

Een applicatie als Copper bestaat uit een aantal modules die uiteindelijk samen een werkend geheel vormen. Copper kent grofweg drie modules:

- Een keuzemodule
- Een lijstmodule
- Een contactmodule



Figuur 3: Overzicht van winkels en zoeken naar winkels

3.1.1 Keuzemodule

Deze module is om een gebruiker de keuze te laten maken uit een aantal opties. In het huidige prototype wordt dat gedaan met behulp van grote knoppen waarop de keuze staat die gebruiker zal gaan maken. Afhankelijk van de gemaakte keuze is het volgende scherm wederom een keuzemodule of een lijstmodule.

3.1.2 Lijstmodule

Copper kent twee soorten lijsten: selectielijsten en contactlijsten.

Selectielijsten - Copper biedt haar gebruikers de mogelijkheid om eerst te zoeken op een bepaalde categorie of in een bepaalde stad alvorens de gevonden winkels weer te geven. Dat zijn uiteraard ook lijsten, maar worden op andere plaatsen gebruikt en leiden naar andere modules.

Contactlijsten - Copper heeft veel gegevens die het moet presenteren naar de gebruiker toe. Lijsten zijn een goede manier om de namen van de winkels/merken op een overzichtelijke en compacte manier weer te geven. Om de gebruiker tijd te laten besparen en Copper dus nog efficiënter te maken is er in de contactlijst, naast de naam van de winkel of merk, een groene telefoonicoon weergegeven. Als de gebruiker op dit icoon drukt wordt er direct gebeld met de desbetreffende winkel of klantenservice van het genoemde merk. Dit werkt uiteraard alleen als er een telefoonnummer bekend is van de winkel of het merk.

3.1.3 Contactmodule

In deze module wordt de gebruiker gedetailleerde informatie van de gekozen winkel of merk gepresenteerd. Er worden adresgegevens (van winkels), telefoonnummer en eventueel een verwijzing naar een contactformulier (van merken) getoond. De gebruiker heeft de optie om dit eenvoudig te delen naar andere mobiele applicaties of naar andere contacten in zijn/haar telefoon.

3.1.4 Samenhang tussen modules

De samenhang tussen deze modules is in de app snel duidelijk. Dat geldt ook voor nieuwe gebruikers zoals is gebleken uit het usability verslag uitgevoerd door TITS. De gebruiker krijgt in het hoofdscherm de keuze om te filteren op specifiekere resultaten aan de hand van plaats of categorie. Het komt dan in de lijstmodules terecht (allereerst in een selectielijst, vervolgens in een contactlijst). Tenslotte komt de gebruiker bij het aanklikken van een winkel of merk in de contactmodule terecht. Vanuit ieder scherm is er de mogelijkheid om terug te gaan naar een vorige module of om verder te gaan naar een andere module. Dit alles zorgt voor dat je snel tussen modules kan wisselen en vergroot de samenhang ertussen.

3.2 Detailontwerp

Copper is ontwikkeld voor het Android besturingssysteem en ondersteunt versies vanaf Android 3.0. Android applicaties worden geschreven in de object-georiënteerde taal Java met de door Android geleverde uitbreiding. Hierbij komen klasses, methodes en attributen kijken. In de bijlage is het complete klassendiagram van Copper bijgevoegd. Hieronder staan de belangrijkste klasses uit Copper uitgelegd.

3.2.1 JSONHelper.class

Copper maakt veelvuldig gebruik van lijsten die uit een externe database worden gehaald. Om dit op een snelle en effectieve manier te doen is hier een aparte klasse voor gemaakt. Hoe de database precies is gevuld en klaargemaakt om door de `JSONHelper` te worden uitgelezen wordt uitgelegd in sectie 3.3.2 - Technische werking. Deze klasse heeft de URL waar het de gegevens uit kan halen daar wordt een `HTTP` verbinding mee opgezet en dit resulteert uiteindelijk in een `JSONArray` gevuld met de objecten uit de database. Eventuele foutmeldingen worden opgevangen en uiteindelijk kan de `JSONArray` door andere klasses worden opgehaald en gebruikt.

3.2.2 ContactsAdapter.class

Zodra de `JSONArray` is opgehaald komen de Adapters aan te pas. Adapters worden gebruikt om de lijstmodules, `ListViews`, te vullen en interactief te maken. We hebben hiervoor een eigen adapter geschreven, om het meer mogelijkheden te geven en specifieker te maken voor onze lijsten, maar het breidt wel de normale `ArrayAdapter` voor strings uit. In de constructor voor deze klasse geven we de context, de grafische representatie (Android `XML`) van iedere lijstitem en de opgehaalde `JSONArray` mee. Vervolgens is er een `getView` methode die het `XML` bestand pakt. Voor ieder item die het moet toevoegen aan de lijst, past het de tekstvelden zodanig aan dat de gegevens overeenkomen met die uit de database. Verder zit hier ook nog wat code in om een alfabetisch geordende "scrollbar" aan de zijkant van het scherm te krijgen. Tenslotte zit er nog een filter methode in. Deze methode wordt gebruikt op het moment dat de gebruiker wilt zoeken naar een specifiek item. De methode loopt de gehele lijst door en laat alleen de items zien die voldoen aan de zoekcriteria van de gebruiker.

Qua attributen bevat het de `JSONArray` en een kopie hiervan, deze wordt gebruikt tijdens het zoeken: de oorspronkelijke lijst wordt namelijk leeggemaakt en er wordt in de kopie gezocht naar overeenkomende items en die worden weer in de oorspronkelijke lijst gezet. Verder bevat het nog de context (de klasse die deze methode klasse heeft aangeroepen). Tenslotte bevat het nog een string die wordt meegestuurd als er op een item wordt gedrukt.

3.2.3 ContactShow.class

Als de gebruiker een winkel of merk heeft gekozen uit de lijst wordt deze klasse aangeroepen. De `ContactShow` krijgt in zijn `intent`, methode die hem aanroept, de naam van deze winkel of merk mee. Vervolgens worden de contactgegevens van deze winkel of merk opgehaald met de eerder beschreven `JSONHelper`. De `JSONArray` wordt gelijk opgehaald zodat deze klasse ook gelijk de lijst met gegevens heeft. Ditmaal zit daar dus de naam, het adres, het telefoonnummer en de geo-coördinaten in. Deze gegevens worden vervolgens in de `updateUI` methode verwerkt en net als bij de `ContactsAdapter` klasse worden de tekstvelden gevuld met de juiste gegevens, die zojuist zijn ontvangen. Enkele eigenschappen worden interacteerbaar gemaakt, zo moet de app bellen op het moment dat de gebruiker op het telefoonnummer drukt. Een ander interacteerbaar veld is het kaartje, die met behulp van de geo-coördinaten en een `HTTP` request aan de Google Maps API wordt verstuurd en ook in beeld verschijnt.

Om tijd te besparen maakt de klasse gebruikt van multithreading, door het gebruik van `AsyncTask`. Hierdoor kunnen meerdere taken (zoals het laden van de contactgegevens en het laden van het kaartje) tegelijkertijd worden uitgevoerd en hoeft de gebruiker zo kort mogelijk te wachten.

Tenslotte is er ook interactie in de `ActionBar`, hiervoor hebben we een `onPrepareOptionsMenu` methode gemaakt en iedere keer als er op het "share" icoontje wordt gedrukt, wordt deze methode aangeroepen die de juiste gegevens ophaalt en doorstuurt naar een andere applicatie.

Op het gebied van attributen is deze klasse goed gevuld, zo worden de gegevens van de winkel/het merk in strings opgeslagen, de context opgeslagen (dat is de `ContactShow` klasse zelf en kan mee worden gegeven aan andere klasses), de opgehaalde `JSONArray` met contactgegevens, de grafische weergave die wordt gebruikt, een `shareActionProvider` die het delen regelt en een boolean waarde of het om een winkel of merk betreft.

3.3 Ontwerpverantwoording

3.3.1 Uiterlijk

Copper bevat een minimalistische interface die vooral uit grote knoppen bestaat. Hier is voor gekozen om het voor de gebruiker direct inzichtelijk te maken hoe de navigatie binnen de app werkt. Allereerst is er een keuzemodule waarin de gebruiker kan kiezen uit 'merk', 'winkels' en 'alles weergeven'. Onder de eerste categorie 'merk' vallen alle bedrijven, voornamelijk multinationals met een Nederlandse afdeling. De tweede categorie bevat de contactgegevens van alle winkels en de derde optie geeft zowel winkels als merken weer. Kiest de gebruiker voor de tweede optie, de winkels, dan krijgt het een selectielijst-module met een de keuze om te filteren op plaatsen en winkelcategorieën. Uiteindelijk worden de contactgegevens van zowel de winkels als merken in een contactlijst weergegeven, zodat de gebruiker gemakkelijk zijn favoriete winkel of merk kan vinden.

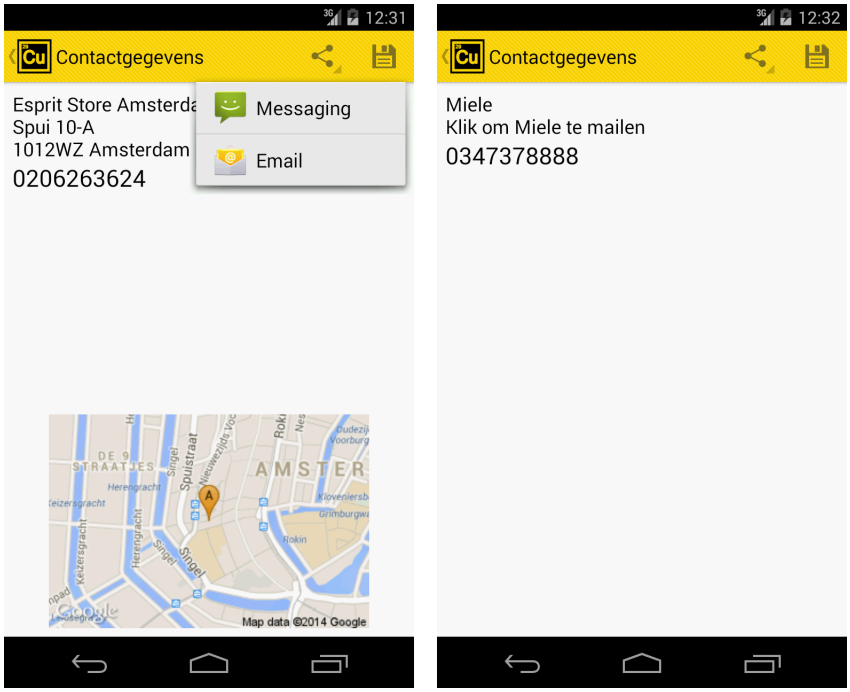
Een andere feature is de zoek-knop rechtsboven in iedere lijst die kan worden gebruikt om te filteren op winkelnaam. Ook dit is gedaan om het gebruik efficiënt te maken. In de screenshots in Figuur 3 is de knop duidelijk zichtbaar.

Onze toekomstplannen met Copper zijn groots. Het uiterlijk zal op de schop moeten om het voor zowel jong als oud aantrekkelijk te maken. De knoppen die we nu gebruiken zijn standaardknoppen zoals Android ze meelevert. De gemaakte afweging op het moment was inhoud boven vorm, we willen eerst de belangrijke features implementeren alvorens we naar het uiterlijk van de applicatie gaan kijken. Deze willen we vervangen door zelfontworpen knoppen die er wat speelser uit zien. Verder zullen we mogelijk de kleuren moeten heroverwegen. Koper heeft een roodbruine kleur, maar dat is een kleur die wij niet vinden passen binnen een app. Daarom hebben we in ons prototype voor goud gekozen.

3.3.2 Technische werking

De app staat in verbinding met een externe database om alle winkelgegevens in te laden. Deze server bevat momenteel ongeveer 6000 winkels per stad. Dit is bij elkaar verzameld door middel van een "crawler": een programma die webpagina's op het internet op een geautomatiseerde manier doorloopt. Deze hebben

we geschreven in de webtaal PHP en de opdracht gegeven om één enkele website te doorlopen waar de gegevens op staan. Dit leek ons de eenvoudigste manier, want 6000 winkels per stad handmatig intypen was geen beginnen aan. Volgens een onderzoek dat is uitgevoerd aan de Faculty of Computers and Information aan de Cairo University door onder andere Marwa Badawi, om zoekmachines actueel te houden door het gebruik van crawlers, staat ook in dat 40% van het webverkeer voornamelijk crawlers/spiders zijn en dat ze ook voornamelijk worden ingezet voor het doel waarvoor wij ze ook hebben gebruikt in Copper, snel webpagina's door te zoeken naar informatie. De merken zijn wel handmatig ingevuld omdat we geen lijst hebben gevonden waar de bedrijven in een overzicht bij elkaar stonden. We hopen hier nog een oplossing voor te vinden om het toch te automatiseren zodat we een grotere lijst merken kunnen aanbieden.



Figuur 4: Contactgegevens van een winkel, met Google Maps navigatie en delen naar andere apps naast de contactgegevens van het merk Miele.

4 Reflectie

Na ongeveer acht weken te hebben gewerkt aan Copper kunnen we wel stellen dat de belangrijkste eisen in de applicatie zijn verwerkt. Het is mogelijk om contactgegevens van zowel winkels als bedrijven snel en gemakkelijk op te zoeken en er kan vanuit Copper direct contact worden gelegd met de gevonden winkel. Door de relatief korte periode voor de ontwikkeling zijn een aantal zaken die ook van belang zijn bij een mobiele applicatie achter wegen gelaten. Zo gaan we zoals ook in de ontwerpverantwoording is aangegeven een mooiere grafische

vormgeving maken voor Copper en ook offline functionaliteiten toevoegen, zodat de gebruiker niet constant online hoeft te zijn om contactgegevens te zoeken. Behalve het product zijn we ook over het proces zeer tevreden. We hebben weinig technische problemen gehad: op een gegeven moment hadden we wel een bug tijdens het zoeken, wat toch wel een essentieel onderdeel is. Die is op dit moment verholpen. Gedurende het proces zijn we ook veel te weten gekomen over het ontwikkelen voor Android en de verschillen tussen "normale" Java applicaties. We zien het gehele proces en het uiteindelijk product als een positieve ervaring, dit komt mede door de groep waarmee we de applicatie hebben gemaakt en ook door het niet al te complexe idee voor deze applicatie. Er waren weinig zaken die wij als negatief ervaren, maar het gebrek aan contactgegevens van grote bedrijven komt dan wel in de buurt. Dit zou je kunnen voorkomen door in het begin, voordat je dit in de applicatie gaat opnemen, eerst te kijken of zo'n overzicht bestaat en vervolgens besluiten om het in de applicatie over te nemen. Dat is uiteraard eerder een "logistiek" probleem en verschilt uiteraard per applicatie en idee dat je hebt. Al met al zijn we tevreden over hoe Copper op dit moment is geworden en zijn benieuwd hoe het later door het publiek wordt ontvangen in de Google Play Store.

Referenties

[1] Ahmed Husseinb Mervat Gheitha Marwa Badawia Ammar Mohameda. "Maintaining the search engine freshness using mobile agent". In: *Egyptian Informatics Journal* 14.1 (2012).