Generieke urenregistratie software voor Smartphones.

De creatie en implementatie van een oplossing over meerdere platformen voor een enkel probleem.

Student	Max Postema
Studenten nummer	1534337
Cursus	Afstudeer scriptie
Datum	13-04-11

Generieke urenregistratie software voor Smartphones.

De creatie en implementatie van een oplossing over meerdere platformen voor een enkel probleem.

HOGESCHOOL UTRECHT

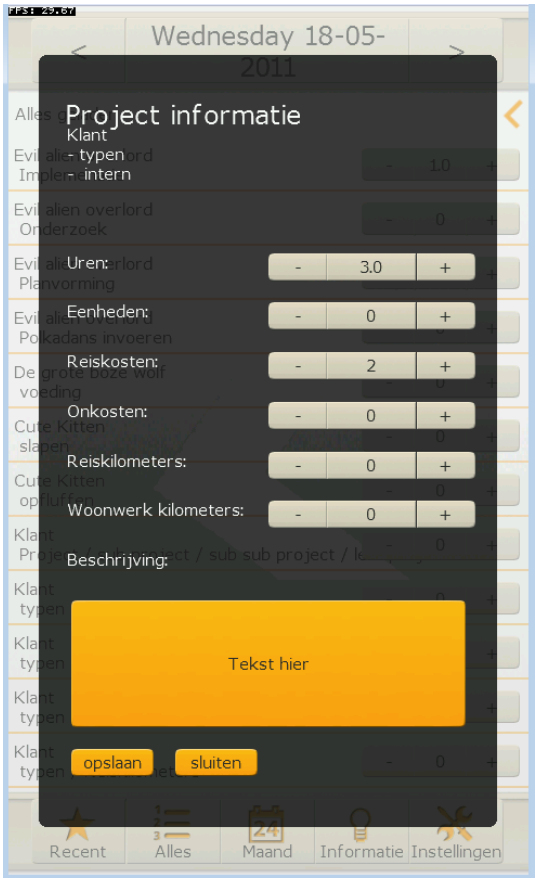
ClockWise
eenvoudig veelzijdig

Student	Max Postema
Studenten nummer	1534337
Cursus	Afstudeer scriptie
Datum	13-04-11
Locatie	Amsterdam
Versie	1.0
Docent	Gerald Ovink
Begeleider	Karel de Vos

Samenvatting

Het bedrijf ClockWise met het gelijknamige urenregistratie software pakket heeft de wens haar urenregistratie mobiel te maken door de ontwikkeling van een smartphone applicatie. Om dit te doen hebben ze een iPhone applicatie ontwikkelt. Dit is bevallen maar niet alle klanten hebben een iPhone, en om voor alle verschillende mogelijke mobiele platformen een eigen applicatie te schrijven zou te duur zijn. Om dit op te lossen hebben ze mij gevraagd een cross-platform applicatie te schrijven die in één keer op zoveel mogelijk verschillende smartphones werkt.

Ik heb onderzoek gedaan en een hoop verschillende cross-platform systemen geprobeerd. Ook heb ik een aantal prototypes ontwikkelt. Een belangrijke aspect is dat het systeem een connectie moet kunnen maken met de ClockWise database door middel van het SOAP protocol. Ook het grafische gebruikers interface (GUI) zal goed genoeg moeten zijn om de klant een betrouwbaar interface te geven waar hij goed mee kan werken. Na een hoop systemen te hebben geprobeerd bleek de AirPlay SDK als winnaar over te blijven.



Afbeelding 1: Uitgebreide Project informatie GUI versie 3

Ik ben uitgegaan van het prototype dat ik heb gemaakt tijdens het onderzoek naar de verschillende cross-platform systemen en ben bezig gegaan met het ontwerpen van de applicatie. Ik heb datatypes ontworpen en de structuur van het programma uitgedacht. Intern is er in de applicatie gebruik gemaakt van een Object georiënteerd gedachten goed. De aspecten van het programma zijn opgesplitst in gedeeltes die hun eigen “problemen” oplossen en zo een werkend programma opleveren. Ook heb een mogelijk interface uitgetekend om alvast te kijken welke aspecten ik tegen kan komen. Ik ben persoonlijk van mening dat tijdens het ontwerp proces pen en papier belangrijke hulpmiddelen zijn maar ik heb ook gebruik gemaakt van de meer digitale UML ontwerp tools.

Met het ontwerp en een kleine planning ben ik begonnen met het schrijven van de applicatie tijdens dit proces ben ik hier en daar kleine problemen tegen gekomen, zoals tijdens de overgang van serieel naar parallelle informatie overdracht. Dit zorgde ervoor dat niet meer bekend was in welke volgorde de informatie de applicatie binnenkwam. De kleine moeilijkheden zijn zonder al te veel problemen opgelost. Op afbeelding 1 is het product informatie scherm van de applicatie te zien. Hier kan extra meta informatie aan een project worden gehangen, niet alleen de uren maar ook reiskosten en dergelijke.

De applicatie is nog niet geheel door ontwikkeld op het moment dat dit document geschreven is. Maar het is nu al een goed werkend product en voor het einde van mijn afstudeer periode verwacht ik dat u de applicatie gratis kan downloaden voor uw smartphone. Vraag u gerust een proef acount aan via ClockWise.info.

Inhoudsopgave

Samenvatting....5
Inhoudsopgave....7
Voorwoord....9
Inleiding....11
1 - ClockWise....13
2 – Organisatie en werkwijze....14
3 – Project / Opdracht....15
3.1 ClockWise Applicatie....15
3.2 Opdracht....15
3.3 Productbeschrijving....15
3.4 Eisen en randvoorwaarden....16
3.5 Use case diagram....16
3.5 Eindproducten....17
4 – Analyse....18
4.1 Overzicht....18
4.2 Analyse mobiele markt....18
4.3 Analyse SDK/API....21
4.4 Analyse Applicatie....32
5 – Ontwerp....38
5.1 Het Class Diagram....38
5.2 Deployment diagram....39
5.3 Sequence diagram....40
6 – Realisatie....41
6.1 Globale fasering....41
6.2 Realisatie per fase....42
7 – Eindproduct....48
7.1 Resultaat....48
7.2 Evaluatie....49
7.3 Conclusie....49
8 – Proces en planning....50
8.1 Project aanpak....50
8.2 Strokenplanning....51
8.3 Project evaluatie....52
9 – Reflectie....53
9.1 Persoonlijke Reflectie....53
9.2 Competentie gebaseerde Reflectie....54
9.3 Profielschets....54
Afkortingen en begrippen....55
Bronnen....58
Bijlage A, afstudeer opdracht....59
Bijlage B, MoSCoW....60
Bijlage C, Onderzochte SDK's....61
Bijlage D, Enquête....62
Bijlage E, UML – Class diagram....63

Voorwoord

In het jaar 2004, voor mijn studie Mediatechnologie aan de Hoge school van Utrecht heb ik gewerkt als programmeur bij het bedrijf *Endendijk en Borst*. Dit onder leiding van de directeur Anita Borst. *Endendijk en Borst w*as een klein bedrijf te Groningen dat zich bezig hield met onder andere het maken van dynamische websites en 3d visualisaties. Er was altijd een gemoedelijke sfeer en de mensen aanwezig hadden een goed technische inzicht en vaardigheden. Ik heb bij *Endendijk en Borst* drie jaar met plezier gewerkt waarna ik heb besloten een studie mediatechnologie te volgen.

In mijn tijd bij *Endendijk en Borst* heb ik gewerkt met en aan een web-based urenregistratie systeem. Dit systeem genaamd ClockWise was gemaakt voor dat ik daar werkte en heb ik langzaam kunnen zien groeien. Nu in 2011 bestaat het bedrijf *Endendijk en Borst* niet meer. Wel bestaat er nu een groter bedrijf met de naam ClockWise. Het software pakket is groter gegroeid en heeft nu meer dan 350 bedrijven die er gebruik van maken als hun uren registratie oplossing. Dit bedrijf, met technisch directeur Karel de Vos en commercieel directeur Anita Borst heeft momenteel 4 werknemers en 3 stagiaires.

Inleiding

ClockWise heeft vele klanten die uren registreren via haar web-based software. Zou het niet mooi zijn als de klant waar hij dan ook maar is uren zou kunnen schrijven. Bijvoorbeeld via een smartphone. Uren schrijven word niet altijd als leuk gezien maar is wel iets wat moet gebeuren. Goed zou zijn als de uren van de dag geschreven kunnen worden in de trein reis naar huis, of misschien meteen al bij de klant op de stoep. In dit afstudeerverslag word dat idee werkelijkheid. Beschreven word het proces waarin een applicatie wordt ontwikkeld die werkt over meerdere smartphones die connectie maken met het internet en de ClockWise servers. Ik ga de software ontwikkelen en testen, onderzoeken doen over keuzes die gemaakt moeten worden en documentatie schrijven. De source code van de applicatie is te groot en complex om naar te refereren wel zal ik aan de hand van schema's de interne structuur van de applicatie weergeven.

Dit document is te lezen als chronologie oplossing van een probleem. Eerst wordt de situatie geschetst daarna zal er een analyse gemaakt worden van het probleem. Wanneer dit er is kan er een ontwerp gemaakt worden. Dit ontwerp moet gerealiseerd worden en levert een eindproduct. Als het project succesvol is uitgevoerd vorm ik een reflectie om toekomstige projecten nog beter te kunnen doen en gemiste punten als nog te herkennen.

Mochten er termen en begrippen zijn waarvan de lezer niet meteen weet wat ik bedoel verwijs ik graag naar de “Afkortingen en begrippen” pagina waar vele gebruikte afkortingen en begrippen kort worden uitgelegd in de context van dit document.

1 - ClockWise

Het bedrijf ClockWise houdt zich bezig met het maken, onderhouden en uitbreiden van het ClockWise uren registratie Systeem. Van nature is het uren registratie software maar tegenwoordig is het een complete ICT oplossing. Urenregistratie, projectmanagement, facturatie en boekhouden kan allemaal worden gedaan met ClockWise.

ClockWise is een web-based software pakket die de gebruikers de mogelijkheid geeft om overal hun uren te schrijven. Of de gebruiker nu thuis is, op de werkvloer of in de trein zit, als er internet is kan hij zijn uren schrijven. Deze vorm van cloud computing word tegenwoordig steeds vaker gebruikt in bedrijven en geeft de gebruikers van ClockWise een hoop voordelen. Het is voor een bedrijf ook mogelijk de software aan te schaffen zodat het op een beveiligd intranet geïnstalleerd kan worden. Niet iedereen vind het fijn dat alle informatie overal via het internet geraadpleegd kan worden. Natuurlijk heeft ClockWise de data goed beveiligt en is er om het in te zien een geldig wachtwoord nodig.

ClockWise is altijd bezig met het verbeteren van haar product, zo is er bijvoorbeeld een Desktop applicatie gemaakt om het uren schrijven te versnellen in kantoor omgevingen. En ook is er een iPhone applicatie zodat het nog makkelijker is om op locatie de uren te kunnen schrijven.

ClockWise heeft een mooi kantoor in Amsterdam Noord waar het bezig is met het product verbeteren en soms klanten worden ontvangen. De programmeurs van het systeem doen zelf de helpdesk, dit zorg er voor dat personen met vragen meestal prima en direct geholpen worden. Dit is ook gebleken uit een enquête gedaan door het bedrijf waarin blijkt dat de klanten zeer te spreken zijn over de helpdesk. Aangezien het software pakket zo intuïtief mogelijk gemaakt word zijn er ook niet heel veel problemen en vragen omtrent de software.

ClockWise heeft ook een maatschappelijke kant, meestal zijn er stages of afstudeerders aanwezig om het team te versterken en geeft het kortingen aan onderwijs instellingen en dergelijke.

Mijn rol in ClockWise is verder te gaan met het verbeteren van de uren registratie dienst. De iPhone applicatie werkt goed, maar doet dit alleen op de iPhone, iPod en de iPad. Zou er bijvoorbeeld een oplossing zijn die alle overige apparaten in een keer kan ondersteunen? Ik ga dit onderzoeken en een applicatie maken voor ClockWise die zo breed mogelijk inzetbaar is.

2 – Organisatie en werkwijze

2.1 Organisatie

De opdrachtgever is Karel de Vos technisch directeur van ClockWise. Anita Borst is de commercieel directeur van ClockWise. Karel en Anita zullen beide feedback geven tijdens de ontwikkeling van deze applicatie en het voorafgaande proces.
Ook aanwezig is Pim Huisman een HBO student die net als ik bezig is met zijn afstudeer project. Hij heeft eerder de ClockWise Desktop applicatie gemaakt en ik heb de mogelijkheid gebruik te maken van zijn kennis omtrent de connectie met de SOAP server.
Ik werk geheel zelfstandig aan het mobiele ClockWise applicatie project en heb de vrijheid nieuwe ideeën en technologieën te proberen die relevant zijn voor een goede ClockWise applicatie.

2.2 Werkwijze

Het afstudeer project zal 840 uur bedragen. Ik werk voor het grootste gedeelte vier dagen per week. Dit alles over de periode van 01-02-2011 tot 31-07-2011. Van week 5 tot week 25 zal het 4 dagen per week zijn en van week 25 tot 30 zal het vijf dagen per week zijn. 20 weken vier dagen per week met acht uur per dag komt op 640 uur. Vijf weken van 40 uur komt neer op 200 uur. Dit samen is de 840 uur. De reden dat ik in de eerste periode vier dagen per week werk is dat ik door kan gaan met de tools lessen als instructeur op de Hogeschool van Utrecht. Zo mogelijk probeer ik in de “vrije” dag thuis te werken aan de ClockWise applicatie.

Vergaderingen vallen op maandag of dinsdag waarbij de aanwezige mensen vertellen hoever ze zijn gevorderd met hun project. Samen word er dan gediscussieerd en bekeken hoe met problemen om te gaan. Aangezien ClockWise niet een erg groot bedrijf is, is er ook veel rechtstreekse omgang met de Karel en Anita. Op elk moment van de dag kunnen ze meekijken of kan ik ze vragen stellen.

Verdere gemaakte afspraken
Aangezien ClockWise zelf ook een helpdesk taak heeft, kan het zijn dat als er niemand aanwezig is , ik de telefoon netjes moet aannemen en de klant helpen met het software pakket.
De werkdag is 8 uur, wel is er de vrijheid tussen 9 en 10 op het kantoor te komen.

3 – Project / Opdracht

3.1 ClockWise Applicatie

Probleemstelling
Wat kunnen we doen om het de klanten van ClockWise nog makkelijker te maken om mobiel hun uren te schrijven.

Subvragen

- Welke SDK/API te gebruiken om een cross-platform oplossing aan te bieden
- Op welke mobiele platformen zal de focus moeten liggen
- Welke technologie en methodieken zijn noodzakelijk
- Hoe kunnen we de applicatie snel houden om zo het gebruikers gemak te vergroten
- Hoe gaat het interface er uit moeten zien om snel en intuïtief de uren te schrijven

Doelstelling
ClockWise heeft klanten die periodiek betalen voor een urenregistratie oplossing. Dit pakket is momenteel web-based. Om de dienst uit te breiden zodat iedereen nog makkelijker overal zijn uren kan schrijven is een mobiele applicatie nodig die uren informatie kan synchroniseren met de ClockWise servers. Het is mijn doel voor de klanten van ClockWise een applicatie te maken voor hun mobiele telefoon die hun de mogelijkheid biedt uren te schrijven waar en wanneer ze dan ook maar zouden willen.

3.2 Opdracht

De opdracht, zoals gegeven door ClockWise is te vinden in bijlage A.

3.3 Productbeschrijving

De applicatie zou een connectie moeten kunnen maken met de bestaande ClockWise SOAP server.
De gebruiker moet via de applicatie in kunnen loggen op de ClockWise server met zijn eigen inlog naam, wachtwoord en ClockWise URL. De ClockWise URL is om aan te geven het ClockWise account het gaat.

De klanten en projecten worden getoond aan de gebruiker waarna hij uren kan gaan schrijven, veranderen of verwijderen. Aan deze uren is ook meta data toe te voegen zoals opmerkingen, eenheden of reiskosten. Ook moet het mogelijk zijn om makkelijk uren te schrijven op verschillende dagen. Weken dienen ingeleverd te kunnen worden wanneer alle uren van de week er goed in staan. In grote databases is het belangrijk dat er een snelle vorm van navigatie tussen de projecten mogelijk is. Aangezien het via internet zijn informatie synchroniseert zou er een correcte afhandeling moeten zijn wanneer het internet weg valt. De gebruiker zou niet zijn net ingevulde uren kwijt moeten zijn.

De applicatie zal moeten gaan werken op zoveel mogelijk mobiele platformen zoals bijvoorbeeld Android, Blackberry en Windows mobile. iPhone is minder belangrijk aangezien hier al een applicatie voor bestaat.

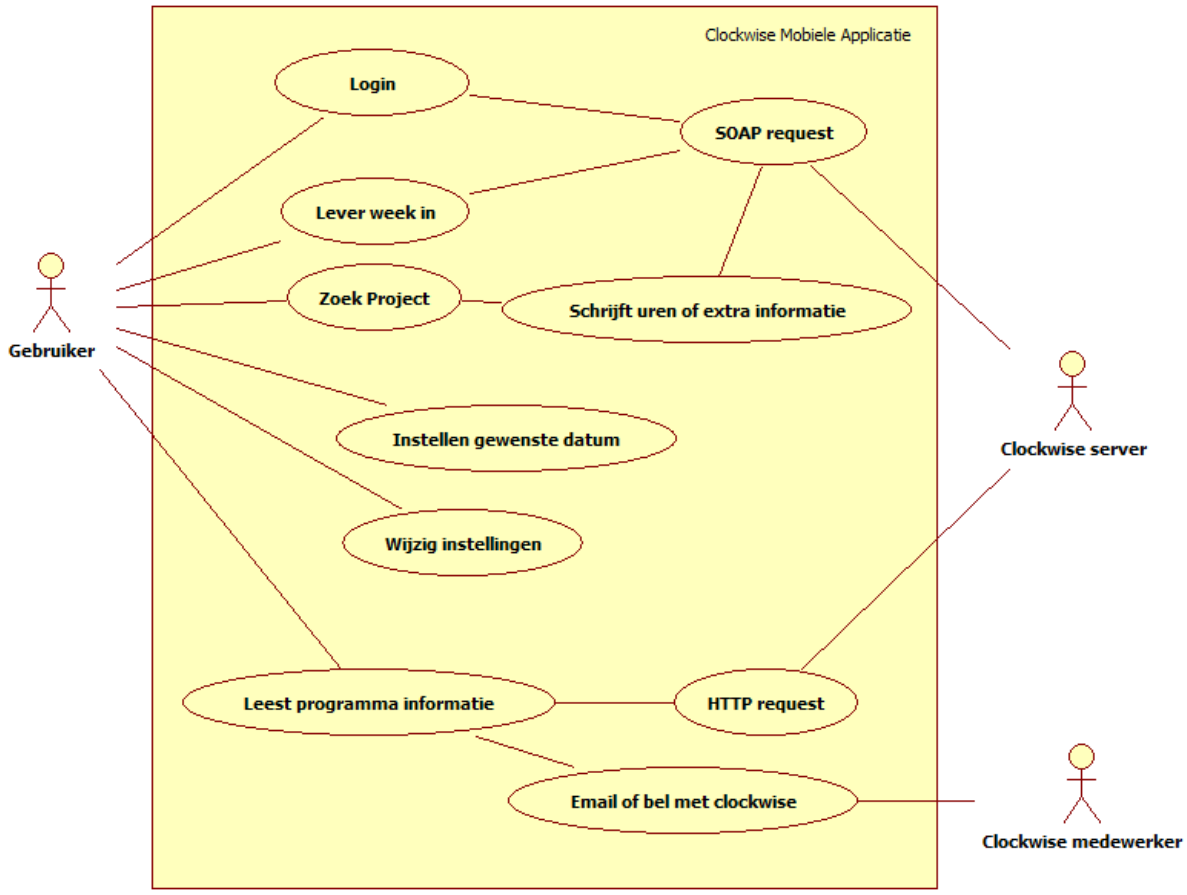
Om te zorgen de de applicatie daadwerkelijk gebruik kan gaan worden, moet worden onderzocht hoe de applicatie bij de klant komt. De applicatie zelf wordt gratis, aangezien het een onderdeel is van ClockWise waar klanten periodiek voor betalen.

Aangezien het een mobiele applicatie is zal er rekening gehouden dienen te worden met de eigenschappen van de mobiele platformen, zo zal bijvoorbeeld het invoeren van tekst tot een minimum beperkt moeten worden aangezien mobiele apparaten over het algemeen niet een heel handig toetsenbord hebben.

3.4 Eisen en randvoorwaarden

De eisen en randvoorwaarden zijn te zien via het MoSCoW document op bijlage B. Hierin worden verschillende mogelijkheden van de applicatie beschreven en ook hoe belangrijk de mogelijkheden zijn voor het functioneren van de applicatie.

3.5 Use case diagram



Afbeelding 2: Use case diagram

De gebruiker van de mobiele ClockWise applicatie kan verschillende dingen doen, deze taken zijn ze zien op afbeelding 2. Dit use case diagram geeft in grote lijnen aan acties een gebruiker kan nemen en welke externe factoren daarmee beïnvloedt worden.

Wanneer de gebruiker start zou hij eerst moeten inloggen, dit genereert een soap request dat word afgehandeld door de ClockWise server. Hierna heeft de gebruiker meerdere mogelijkheden. Zo kan hij een week inleveren, een project zoeken, de actieve datum instellen (via het kalender) of zijn instellingen zoals inlog gegevens wijzigingen. De Instellingen en datum wijzigen zijn veranderingen in de applicatie zelf. Zoals te zien is op het use case diagram heeft het geen verbinding met externe factoren. Inloggen, een week inleveren en een project zoeken genereren SOAP requests. Het “Leest programma informatie” doelt op de informatie pagina in de applicatie, de informatie daarin word gehaald van de ClockWise webserver met een HTTP request. Ook is het daar mogelijkheid om te bellen of emailen naar ClockWise.

3.5 Eindproducten

De gewenste eindproducten zijn te zien in het afstudeer opdracht Bijlage A

- De bron code van de applicatie (met commentaar)
- De applicatie
- Documentatie over de applicatie
- Handleiding over exploitatie, hoe is de applicatie bij de gebruikers van ClockWise te krijgen.
- Presentatie van de opdracht

4 – Analyse

4.1 Overzicht

Voor de communicatie met de ClockWise server is een SOAP webservice beschikbaar. Er is al een iPhone applicatie aanwezig die verbinding kan maken met de ClockWise servers. Ook is er een desktop applicatie in de maak die binnenkort gelanceerd wordt. Kennis van de SOAP service is ook aanwezig.

4.2 Analyse mobiele markt

Er zijn veel besturingssystemen voor de huidige generatie aan mobiele telefoons. Maar welke is het best geschikt voor de ClockWise applicatie, hoe ziet de markt er nu uit en wat zijn de voorspellingen van de toekomst. Voor de iPhone heeft ClockWise al een applicatie maar ik neem Apples besturingssysteem voor de compleetheid als nog mee in mijn zoektocht. Ik heb het internet afgezocht naar hoe de mobiele markt er momenteel uit ziet, hoe het is gegroeid en wat de voorspellingen voor de toekomst zijn. De meest dominante bronnen zijn terug te vinden in de bronvermelding. Ook heb ik onderzocht wat de huidige klanten van een mobiele ClockWise applicatie zouden willen zien en hun gedrag op de ClockWise.info website.

4.2.1 Lijst van mobiele besturingssystemen

Android van Google

- *Bada* van Samsung
- *BlackBerry* OS van RIM
- *iOS* van Apple
- *Maemo* van Nokia
- *Moblin* van Intel
- *Symbian* OS van Nokia
- *WebOS* van HP (Opvolger van Palm OS)
- MeeGo
- *Windows Mobile / Windows Phone 7* van Microsoft

Android is een snel groeiend systeem. Het loopt langzaam in op Apples iOS en er wordt nog een hoop van verwacht. Aangezien het gebaseerd is op het vrije lLinux systeem is het mogelijk veel dingen te doen op een Android apparaat, dit maakt het soms een iets complexer systeem maar met meer vrijheid bij het ontwikkelen van applicaties.

Bada, webOS, Maemo, Moblin en MeeGo zijn net als Android gebaseerd op Linux. Dit zijn de kleinere mobiele besturingssystemen. Maemo en Moblin zijn later samengevoegd onder de nieuwe naam MeeGo. Geen van deze besturingssystemen heeft een dominante markt positie.

iOS bekend van iPhone, iPad en iPod is momenteel de grote speler, het heeft een niveau gezet waar alle andere spelers naar proberen te streven. Het begint aan populariteit te verliezen door Android maar het blijft een sterke positie houden op de mobiele markt.

Blackberry is bekend van zijn fysieke toetsenbord op het mobiele apparaat. Het is erg geliefd in de commerciële wereld. Toch word deze creatie van RIM momenteel weggedrukt door het grote succes van iOS en Android.

Symbian bestaat langer dan de meeste alternatieve besturingssystemen. Vele oude telefoons gebruiken dit besturingssysteem maar er zijn ook nieuwe telefoons die het gebruiken. Symbian heeft ook veel verloren door het succes van iOS en Android. Het laatste nieuwe vertelt dat Nokia die momenteel het Symbian besturingssysteem ontwikkelt over gaat naar *Windows Phone 7.* Dit zou nog wel eens de dood van Symbian kunnen betekenen.

Windows Mobile / Windows Phone 7 is een erg kleine speler op de mobiele markt. Microsoft probeert markt aandeel te winnen met de nieuwe *Windows Phone 7.* Meningen over dit toekomstig succes zijn nog sterk verdeeld.

Het is erg lastig om vast te stellen voor welke mobiele platformen we het beste een applicatie kunnen maken. De markt is erg variabel, nieuwe systemen komen en gaan. IOS, Android en RIM zijn momenteel de grote sterke spelers. Het gebruik van Windows Phone 7 kan toenemen en de verwachting is dat het gebruik van Symbian gaat afnemen.

4.2.2 Bezoekers op ClockWise.info

Bezoek van laatste jaar op ClockWise.info via mobiele apparaten.	
iPhone	330
iPad	132
Android	33
iPod	20
Symbian	15
Windows	4
BlackBerry	2

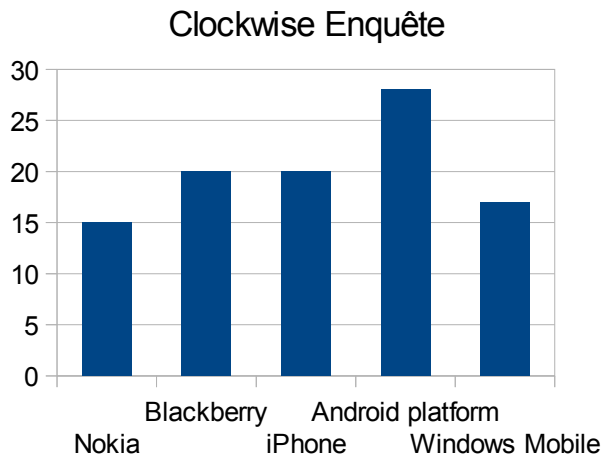
Tabel 1: Mobiele bezoekers op ClockWise.info

Om te beginnen ben ik gaan kijken naar de mobiele bezoekers op ClockWise.info (Zie Tabel 1) Niet erg veel mensen komen mobiel op de website van ClockWise zelf. De iPhone/iPad/iPod (iOS) komen het meeste langs op ClockWise.info. Dit is niet heel verbazend aangezien er al een ClockWise applicatie is voor iOS. Verder hebben ook de meeste ClockWise programmeurs een iPhone wat de resultaten beïnvloedt. We hebben een stuk minder Android bezoekers op de website. Wel worden er momenteel wereldwijd meer Androids verkocht dan van Apples telefoons. Toekomst gericht zou een applicatie voor de Android misschien een goed idee zijn. Symbian telefoons worden een hoop verkocht, maar daar zitten ook veel simpele apparaten tussen die niet effectief zullen zijn voor een ClockWise urenregistratie applicatie. Het lage bezoekers aantal van Blackberry apparaten verbaast mij. Blackberry van RIM staat bekend als grote speler in de commerciële markt, dezelfde markt waar ClockWise met haar product in staat.

4.2.3 Enquête

We hebben in de ClockWise nieuwsbrief een digitale enquête gedaan om uit te zoeken welke smartphones de klanten van ClockWise gebruiken. De enquête zelf is te zien op bijlage D. Deze enquête hebben we expres klein gehouden om niet teveel tijd van de klant te vragen. We hebben zelf een selectie gemaakt van de meest belangrijke platformen. Dit zijn Nokia's Symbian, Blackberry, Android, iPhone en Windows Mobile.

De mogelijke antwoorden; “niet”, “een enkele keer “, “redelijk vaak”, “vaak”, “uitsluitend” zijn respectievelijk de waardes 0, 1, 2, 3, 4 gegeven om uit te vinden hoe veel interesse er was voor de bepaalde platformen.



Afbeelding 3: Interesse van klanten van ClockWise in een mobiel platform

Uit de enquête kunnen we opmaken dat er in ieder geval veel vraag is naar een Android applicatie, aangezien de wens boven alle andere uit steekt. Ook kunnen we zien dat er interesse is voor alle mobiele platformen.

4.2.4 Conclusie

Android is een snelle groeier die waarschijnlijk gaat blijven. Dit blijkt zowel uit de enquête als de analyse van de mobiele markt. Ook stond Android op de tweede plaats na iOS wat betreft bezoek aan de ClockWise website. Hierdoor is Android een belangrijke focus punt is voor de volgende ClockWise mobiele applicatie. Apples iOS blijkt net zo belangrijk maar aangezien daar al een applicatie voor bestaat wordt iOS niet mee genomen in het onderzoek naar een geschikte SDK.
Ook blijkt dat de andere systemen veel gebruikt worden en dat de wens van ClockWise om een globaal cross-platform systeem overeen komt met de wens van de klanten, en overeen komt met de ontwikkeling van de markt.

4.3 Analyse SDK/API

Voor dat ik kan beginnen met het maken van de applicatie moet een cross-platform SDK gevonden worden. Ik ben opzoek gegaan naar zoveel mogelijk systemen die het mogelijk maken een cross-platform applicatie te schrijven. Ik heb uitgezocht hoe ze werken, wat de voor en nadelen zijn en of we de ClockWise applicatie ermee kunnen ontwikkelen. Voor de systemen die interessant waren heb ik een prototype gemaakt om zeker te zijn dat het mogelijk is de applicatie daarin te ontwikkelen.

Er zijn veel systemen die het mogelijk maken om een applicatie te draaien over meerdere smartphones. Ik heb hier groot aantal onderzocht.

4.3.1 Categorieën

De gevonden oplossingen zijn in te delen in de volgende categorieën.

- Webkit
- Cross Compiler
- Native Arm
- Server-based

Webkit
Webkit is een open source browser engine die het mogelijk mogelijk maakt om binnen applicaties websites te presenteren. Aangezien het open source is, is het zelf te compileren over een hoop verschillende platformen. Sommige cross-platform smartphone systemen maken hier gebruik om de presentatie van de applicatie over alle apparaten het zelfde te houden. Het grote nadeel is dat het gemaakte programma niet anders is dan een website. In HTML kan je erg makkelijk een grafisch interface ontwikkelen maar voor complexe logica is Javascript toch echt erg traag.

Cross Compiler
Cross compilers maken het mogelijk broncode te compileren voor andere systemen dan waar de compiler op staat. Hierdoor is het mogelijk om van 1 bron, meerdere programma's te compileren over verschillende systemen. Een groot nadeel is dat het niet zeker is dat elke verschillende bron de code op de zelfde manier uitvoert. Mocht de implementatie van een doel systeem anders zijn dan van een ander doelsysteem zouden er al weer uitzonderingen per systeem in de code gedaan moeten worden.

Native Arm
Bijna alle smartphones zijn gebaseerd op de ARM processor, door code hier voor te compileren kunnen we een systeem maken die over alle smartphones het zelfde werkt. Dit heeft wel het nadeel dat het onmogelijk word gebruik te maken van de bij de smartphone meegeleverde software API's. Alles moet opnieuw gemaakt worden. Vooral het GUI is hierin een moeilijk punt.

Server based
Door een applicatie te maken die zijn uiterlijk aanpast op gegevens die het krijgt via een server is het mogelijk aan RAD (Rapid Application Development) te doen. Met deze methode kan je relatief makkelijk software applicaties ontwikkelen met het nadeel dat je niet totale controle over de applicatie heeft. Er zullen situaties ontstaan dat ClockWise of haar klanten iets zouden willen wat in het systeem echter onmogelijk te doen is.

4.3.2 Onderzochte systemen

Ik heb de volgende systeem gekozen om te onderzoeken.
Er word onderzocht hoe ze werken, al het GUI de mogelijkheden bied die nodig zijn voor de applicatie, welke mobiele platformen de SDK ondersteund en als het mogelijk is SOAP requests te sturen naar de ClockWise server. Bijlage 4 geeft hier ook nog een samenvatting van.

- Native Applications
- Java ME
- Java met abstractielaag
- Mobile website
- Mobile website in webkit
- PhoneGap
- Rhodes
- Appcelerator
- MoSync
- WorkLight
- QuickConnectFamily
- MobileReflex
- iPFaces
- moppr
- Jmango
- Maobl
- MotherApp
- Tersus
- Airplay
- metismo
- Dragon RAD
- Smartface

Native applications

We kunnen voor elk apparaat afzonderlijk een programma schrijven. Dit geeft totale vrijheid om met het apparaat te doen wat we willen. Optimalisatie en oplossen van problemen kunnen dan worden gedaan op de verschillende ontwikkel platforms. Groot nadeel hiervan is de tijd, geld en moeite die het zou nemen om de verschillende versies te ontwikkelen en bij te houden.

Deze methode is niet te gebruiken voor de ClockWise applicatie omdat het te lang zou duren om voor alle verschillende platformen een eigen applicatie te schrijven. Ook zou het erg lastig zijn wanneer er een verandering door gevoerd zou moeten worden. Dit aangezien dan dan voor elke platform het opnieuw gemaakt zal moeten worden.

Voordelen: Totale controle over de smartphone en zijn mogelijkheden.
Nadelen: Duur, Tijdrovend.

Java ME
www.oracle.com/javame

Java 2 Platform Micro Edition (J2ME-platform) is een Java versie gemaakt voor kleine apparaten. De ondersteuning hiervoor is bij de grote smartphones afwezig. Dit voornamelijk omdat het J2ME-platform te globaal is maar ook te oud om de complexe functies van de smartphone te ondersteunen.

Er zijn Java ME wrappers speciaal voor de Android, dit maakt het met een omweg ook mogelijk voor de Android te ontwikkelen via Java ME.

	Native Ontwikkel Taal	Java ME compatibel
Android	Java	Nee *
iOS	Objective C	Nee
BlackBerry	Java	Ja
Symbian	Java / C++	Ja
Windows	Visual C	Ja
Palm webOS	Web / C++	Nee

* Niet out of the box

Tabel 2: Ondersteuning van java bij verschillende platformen

Verder is het niet mogelijk om gebruik te maken van de native widgets, wel zijn er oplossingen als lwuit en j2mepolish die een eigen, redelijk nette GUI bieden. Ook kan een low-level maar globale LCDUI worden gekozen, al zou er hier dan een hoop nieuwe widgets bij gemaakt moeten worden.

Java ME is niet geschikt om de ClockWise applicatie mee te maken, de technologie is oud en werkt als nog niet goed cross-platform door verschillende implementaties op de verschillende apparaten. Ook is de GUI niet goed genoeg om de applicatie mee te maken. Een eigen GUI maken zou te veel tijd innemen om op professioneel niveau te krijgen.

Voordelen: Een applicatie maken met J2ME zou over meerdere telefoons werken.
Nadelen: J2ME word eigenlijks voornamelijk ondersteund op de “net-niet-smartphones”. Het is moeilijk of onmogelijk native widgets te gebruiken.

Java met eigen abstractielaag

We kunnen een Java applicatie maken waarbij er word gecontroleerd voor welke telefoon het gecompileerd word. Door in deze applicatie een abstractie laag te gebruiken (Abstract Factory Design Pattern bijvoorbeeld) kunnen we de telefoon specifieke methodieken apart afhandelen. Maar een gedeelde kern houden in alle verschillende distributies.
We zouden zo een Android Java / JavaME programma kunnen maken die het grootste gedeelte van de smartphones zou moeten ondersteunen. Of Android Java / Blackberry Java / Symbian Java zodat we van alle apparaten de voordelen kunnen gebruiken die JavaME niet zou kunnen bieden.

Dit zou een mogelijkheid kunnen zijn voor de mobiele applicatie. Wel zou het complex zijn en veel tijd nemen om de verschillende API's te wrappen naar een CORE class. Ook de verschillende methodieken over de apparaten zijn niet makkelijk met elkaar te binden.

Voordelen: Gebruik telefoon specifieke eigenschappen in een enkel programma voor meerdere smartphones
Nadelen: Alleen ondersteuning voor alle smartphones met Java, Complex en tijdrovend om te implementeren.

Mobile website

Door een website te maken geoptimaliseerd voor mobiele apparaten geven we elk mogelijk apparaat toegang tot ClockWise. Webtechnologie is natuurlijk van huis uit goed in renderen op verschillende resoluties. Ook heeft ClockWise veel ervaring met webtechnologie. Op sommige apparaten is het mogelijk bookmarks maken die er uit zien als programma's. Maar dit zou de gebruiker dan zelf moeten doen.

Al is dit een goede oplossing om makkelijk elk mobiel apparaat te voorzien van een ClockWise interface is het ook erg gelimiteerd. Zo heeft een website geen toegang tot eigenschappen en informatie op het apparaat zelf. Ook heeft het niet de voordelen van een applicatie, zoals verder werken wanneer het internet even weg valt. En de snelheid van een native GUI.

Voordelen: Elk apparaat kan gebruik maken van de dienst, de dienst is in eigen beheer dus makkelijk up-to-date te houden. Resolutie onafhankelijk.
Nadelen: Niet een applicatie, meer een bookmark op de smartphone. Totaal geen mogelijkheid gebruik te maken van de eigenschappen van de smartphone.

Mobile website in een native shell

Door een mobiele website te maken zoals het in het vorige punt, en voor elke smartphone een eigen Shell te schrijven die niets anders doet dan de website openen kunnen we een applicatie aanbieden aan de gebruiker. Zelf wanneer het apparaat even geen internet heeft kunnen we de gebruiker hier zelfs netjes van op de hoogste stellen. Maar verder zou een dergelijke applicatie gewoon een mobiele website renderen. In dit programma zou software als webkit kunnen draaien of toegang gevraagd worden aan de op het apparaat geïnstalleerde browser.

Dit is een interessante methode maar er zijn vele pakketten die dit gebruiken en hier zelfs aan voorbijgaan. Denk hierbij bijvoorbeeld aan PhoneGap.

PhoneGap
www.phonegap.com

Met PhoneGap bouw je in HTML5, CSS3 en javascript een applicatie die dan intern via webkit gerenderd word als website. Dit werkt voor de meeste smartphones (iOS, Android, Blackberry, Palm, Windows en Symbian). PhoneGap is open source en kost niets. Ook geeft het via javascript de mogelijkheid dingen als de Accelerometer en GEO location aan te roepen. Het is momenteel bezig het het maken van een build service zodat zij jou applicatie compileren klaar voor elke mobiele markt. Deze methode ondersteund: Android, BlackBerry, iOS, Symbian en webOS.
Met toolkits als sencha touch is webbased mogelijk een interface te renderen die er uit zien als de native widgets van de smartphone. Met dit soort toolkits is het zelfs mogelijk speciale effecten zoals rotaties en slides als overgang tussen schermen te ondersteunen.

Voordelen: Html is van nature resolutie onafhankelijk, ClockWise heeft veel ervaring met webtechnologie.
Nadelen: Het is relatief traag. Er is geen ondersteuning voor native widgets. Problemen met SOAP service aan te roepen.

Rhodes
www.rhomobile.com

Rhodos is een open source systeem waarmee je met Ruby code native applicaties voor smartphones kunt ontwikkelen. Het komt commercieel over en heeft een prijskaartje wanneer de gebouwde applicatie niet open source is. (500 dollar per applicatie) Het heeft ook systemen voor het compileren en verspreiden van de applicatie.

Rhodes ziet er veel belovend uit. Het is complexe software met vele mogelijkheden. Toch is het niet handig deze software te gebruiken. Het was erg moeilijk om de SDK te installeren en de learning-curve van het creëren van software ervaar ik als erg steil. Als er geen betere alternatieven zijn zou ik hier op terug kunnen komen maar voor nu is het te complex om teveel tijd aan te besteden.

Voordelen: Werkt over alle systemen
Nadelen: Duur, console based development

Appcelerator
www.appcelerator.com

Dit gebruikt webtechnologie als HTML5 en CSS3 om applicaties te maken. Appcelerator bestaat niet alleen voor de mobiele markt. Ook bied het de mogelijkheid om applicaties te maken voor de PC die crossplatform zijn op verschillende besturingssystemen. Deze mogelijkheid is gebaseerd op de zelfde webtechnologie. Appcelerator word gebruikt door grote namen wat naar mijn mening wel spreekt voor de betrouwbaarheid. Nadeel is dat het van de smartphones alleen de iPhone en Android ondersteund en niet de vele kleinere alternatieven.
Een voordeel is dat Appcelerator niet meteen de binary maakt maar de gebruikte code genereert zodat er desnoods per platform nog mutaties zijn door te voeren.

Toch kan dit niet gebruikt worden voor de ClockWise applicatie. Een iPhone applicatie bestaat al en om dit te gebruiken voor alleen een Android applicatie zo zinloos zijn. Dan is namelijk beter de Android java API te gebruiken om totale controle over het apparaat te hebben.

Voordelen: Veel gebruikt. Desktop applicatie mogelijk.
Nadelen: Geen ondersteuning voor alle platformen.

MoSync
www.mosync.com

De code word geschreven in C++. Dit word door een MoSync GCC backend vertaalt naar bytecode wat later door een MoSync word vertaalt naar een platform afhankelijk programma. In andere woorden, het is mogelijk een programma te schrijven in één taal, en dat te vertalen naar alle mogelijke andere platformen. Het gebruikt alleen niet de native widgets van de smartphone zelf. De bijgeleverde GUI is zeer primitief en niet geschikt voor een professionele applicatie.

Voordelen: Interessant systeem, effectief en snel.
Nadelen: Geen native widgets. Slechte alternatieve GUI.

WorkLight
www.worklight.com

Om de SDK te downloaden moest ik best lang wachten omdat er wordt gecontroleerd door een persoon als de aanmelding wel van een echte software ontwikkelaar is. Na een paar dagen heb ik de SDK kunnen downloaden en de documentatie in kunnen kijken. Ook WorkLight gebruikt weer webtechnologieën om applicaties over verschillende apparaten te verspreiden. Een voordeel ten opzichte van PhoneGap is dat zelf ondersteuning bieden met javascript SOAP-services te gebruiken.

Toch kan ik WorkLight niet aanraden voor de ClockWise applicatie. WorkLight ondersteund maar een paar smartphones, er zijn betere gratis oplossingen. Mocht er geen betere alternatief gevonden worden is het natuurlijk altijd mogelijk uit te zoeken hoe zij SOAP aan de praat hebben gekregen en die technologie dan toepassen binnen PhoneGap.

Voordelen: Op javascript gebaseerde SOAP client.
Nadelen: Duur, Ondersteuning voor weinig apparaten.

QuickConnectFamily
www.quickconnectfamily.org

QuickConnectFamily heeft een zeer onprofessionele site maar gelukkig is het systeem zelf beter. Zoals PhoneGap werkt het door een HTML5 applicatie te maken in een webkit wrapper. Een iets tragere techniek maar wel goed over te dragen binnen de bestaande smartphones. QuickConnectFamily lijd wel aan een magere documentatie en komt op veel punten erg onprofessioneel over. Officieel is er een sterke ondersteuning voor Xcode op mac, een ontwikkelplatform waar ik geen toegang tot heb. Het is mogelijk ook gebruik te maken van Eclipse maar dit gaat net allemaal wat moeilijker.

Jammer genoeg heeft QuickConnectFamily weer geen ondersteuning voor alle beschikbare smartphones maar in ieder geval een aantal. Ik heb er geen vertrouwen in dat de applicatie maken met QuickConnectFamily een goede optie is, goede documentatie is belangrijk, niet alleen om problemen die gaan komen op te lossen maar ook om het mogelijk te maken dat andere mensen aan de applicatie kunnen werken.

Voordelen: Webtechnologie, veel verschillende javascript API's te gebruiken.
Nadelen: Ondersteuning voor weinig apparaten, slechte documentatie.

MobileReflex
www.mobilereflex.com

MobileReflex is een RAD toolkit je kan de GUI maken via een XML bestand. Alle data aanvragen van de applicatie verlopen via de MobileReflex server en hebben wij bij ClockWise dan geen controle over. De cliënt zelf heeft hierdoor geen applicaties logica en daardoor makkelijk beschikbaar over alle smartphones. De controle die we kunnen uit oefenen over de loop van het programma word hier wel door gelimiteerd. Dit is niet geschikt voor de ClockWise applicatie. We hebben zelf servers en willen onze eigen data kunnen beheren. Controle over de werking van de applicatie is ook een belangrijke eigenschap.

Voordelen: Goede ondersteuning van de verschillende smartphones.
Nadelen: Afhankelijk van de MobileReflex server.

iPFaces
www.ipfaces.org

iPFaces werk op ongeveer de zelfde methode als MobileReflex. Met iPFaces geef je de al bestaande iPFaces applicatie de locatie van je eigen server en daar vandaan haalt hij de beschrijving hoe de applicatie er uit moet zien. De intelligentie is zodoende geheel in PHP te maken waarbij de iPFaces applicatie niets meer is dan een simpele interface met de klant. Het is een interessant systeem maar wil je een iPFaces applicatie met een ClockWise logo betaal je daar al snel vele honderden euro's voor per platform.

Aangezien de applicatie al bestaat en data van buiten de applicatie interpreteert is snelle ontwikkeling mogelijk, maar ook hier raken we controle kwijt van wat de applicatie kan doen. De applicatie gaat zodoende heel erg lijken op web formulieren. En kan onze klanten niet een dienst geven die intuïtief en prettig is.

Voordelen: Snel applicaties maken.
Nadelen: Overdreven duur.

m|oppr
www.moppr.com

Van m|opper is momenteel alleen nog maar een beta versie te downloaden. Er zijn aankondigingen geweest voor ondersteuning van meerdere platformen maar momenteel is het alleen beschikbaar voor Symbian en iOS. Aangezien het nog een nieuwe opkomende techniek is en nauwelijks ondersteuning heeft voor de apparaten die wij willen gebruiken zullen we dit niet gebruiken voor de mobiele ClockWise applicatie.

Nadelen: Alleen ondersteuning voor de iOS en Android, m|oppr zit nog in de beta fase.

Jmango
www.jmango.net

Met Jmango kan je via de browser een applicatie bouwen. Dit is naar mijn mening bijzonder onhandig. Web pagina's laden langzaam. De applicaties zijn redelijk simpel en mist de rijkheid die we nodig hebben voor een ClockWise applicatie. De applicatie zou hierdoor de zelfde mogelijkheden krijgen als een HTML Form. De voorbeelden gebruiken allemaal erg oude technologie, aangezien wij ons willen richten op de nieuwere smartphones zouden we met deze techniek onze doelgroep komen te missen missen.

Nadelen: Web-based bouwen van applicatie, complexe programma's niet mogelijk.

mobl
www.mobl-lang.org

Genereert een HTML5 applicatie die draait binnen webkit. Het gebruikt hiervoor een eigen taal genaamd mobl. Aangezien het een nieuwe taal is zijn er weinig voorbeelden en library's te vinden om een complexe urenregistratie applicatie mogelijk te maken. De bijgeleverde GUI ziet er netjes uit maar is ook redelijk simpel. Het systeem heeft mogelijkheden, maar is nu nog te klein om te gebruiken voor een grote applicatie.

Voordelen: Gebruikt een taal ontworpen voor crossplatform mobiele applicaties.
Nadelen: Alleen ondersteuning voor de iPhone, Android en Palm.

MotherApp
www.motherapp.com

Definieer een programma in een variant van HTML en MotherApp genereert via internet een native programma op maat. Technologie werkt, ik heb zelf al veel gebruik gemaakt van een TED cliënt die gemaakt is met MotherApp. Groot nadeel hier is dat hij alleen maar bestaat voor iPhone, Android en Blackberry. Wij willen met onze mobiele ClockWise applicatie meer smartphones kunnen gebruiken dat deze 3.

Voordelen: Bewezen technologie, neemt compileren uit handen.
Nadelen: Alleen ondersteuning voor de iPhone, Android en Blackberry.

Tersus
www.tersus.com

Tersus is een visuele programmeer taal om websites te maken, Het heeft ook de mogelijkheid tot het exporteren naar een iPhone programma. Jammer genoeg zijn andere apparaten nog niet ondersteund. Visuele programmeer word steeds sterker maar kan nog niet op tegen een professionele programmeertaal. Er is minder mogelijk en word snel onoverzichtelijk.

Voordelen: Grafisch programmeren is in theorie sneller.
Nadelen: Alleen ondersteuning voor de iPhone.

AirPlay
http://www.airplaysdk.com/

Met AirPlay word via een cross compiler rechtstreeks ARM machine code geproduceerd voor het mobiele platform naar keuze. De taal waar in geprogrammeerd word en C++ De opvolger van de taal C die nog wel eens de moeder taal van alle programmeer talen word genoemd. Het ziet er veel belovend uit. Vooral als het daadwerkelijk crossplatform gaat werken. Hiervoor zou ik nog een paar testen moeten doen over verschillende apparaten. Het kan geen gebruik maken van de native GUI maar heeft zelf een GUI die er redelijk goed uit zien en snel te ontwikkelen is.

Voordelen: Prima programmeer taal, HTTP SOAP aanvragen te doen zonder problemen. Ondersteuning voor alle smartphones.
Nadelen: Niet mogelijk de native GUI van de smartphone zelf te gebruiken

Metismo
http://www.metismo.com/

Metismo maakt gebruik van een cross compiler om snelle native code te produceren voor alle apparaten. Het heeft een interface die lijkt op Java ME. Aangezien Java ME toch redelijk oud is zou het kunnen zijn dat de GUI en dergelijke niet erg professioneel er uit gaan zien. Aangezien ik nergens op de site een mogelijkheid heb kunnen vinden een SDK te downloaden, of een plek waar ik zou kunnen inloggen kan ik dit niet met zekerheid zeggen. Wel vind ik het zeer slordig om geen SDK aan te bieden zonder contact met het bedrijf op te nemen. Hierdoor kan er nooit een sterke actieve community opgroeien waar je altijd voor de meeste problemen antwoorden kan vinden.

Nadelen: Geen SDK te vinden

Dragon RAD
http://www.dragonrad.com/

Dragon RAD applicaties worden gemaakt in een WYSIWYG style editor. Je maakt een connectie aan een databron, en verteld hoe de velden op het scherm gepresenteerd dienen te worden. Simpel maar effectief systeem en nog best te gebruiken voor ClockWise. De specialisatie zit vooral in Blackberry's maar het ondersteunt ook de andere grote mobiele besturingssystemen. Het is duur wanneer er meer dan twee apparaten zou moeten ondersteund worden. Als het werkt zo als geadverteerd zou een applicatie hierin heel snel gemaakt kunnen worden. Wel gaat dit weer ten koste van de controle die we hebben over de ontwikkeling. De applicatie zou ook erg veel gaan lijken op een webform of een spreadsheet.

Het zou een optie kunnen zijn, ik hoop alleen dat er betere opties gevonden kunnen worden. Hoewel RAD ontwikkeling snel is, geeft het niet veel vrijheid. En als een klant een nieuwe functie wil hebben. Wil ik hem niet verkopen “nee, dat laat onze ontwikkel omgeving niet toe”

Voordelen: RAD ontwikkeling
Nadelen: Duur (4,900 dollar per ontwikkelaar per jaar), Mindere mate van controle over de applicatie.

Smartface
http://developer.smartface.biz/

Net als Dragon RAD is Smartface een RAD toolkit gemaakt om snel een applicatie te ontwikkelen. Wat me tegenviel is dat het gebruik maakt van Java ME het interface is weer simpel en gelimiteerd tot de mogelijkheden van Java ME. De applicatie moet aan de gebruiker vragen voor toegang tot het internet (omdat het nog wel eens geld zou kunnen kosten). Dit is netjes maar oud aangezien de meeste smartphones altijd een internet connectie hebben en open houden. Het interface is duidelijk, het programma is goed maar persoonlijk vertrouw ik niet in het ontwikkelen van RAD applicaties. De controle over het eindproduct die je in levert om snel een oplossing te krijgen is naar mijn mening een te hoge prijs. Daarbij het het financieel prijskaartje van RAD omgevingen ook altijd sterk aanwezig.

Voordelen: RAD ontwikkeling.
Nadelen: Duur, (1,490 dollar per applicatie) Mindere mate van controle over de applicatie.

Conclusie

PhoneGap, Rhodes, Appcelerator en MoSync zijn allemaal interessante systemen die beloven in één ontwikkel systeem een applicatie te kunnen maken voor alle smartphones. Onderling is hierin natuurlijk best wat verschil. MoSync ziet er betrouwbaar uit en is snel genoeg om een professioneel product te maken. Het is alleen jammer dat het geen native widgets heeft, wat de ontwikkeltijd zou verlengen. Appcelerator is interessant omdat hij native code genereert die achteraf nog bij te werken is per platform. Bij de andere platformen is dat niet mogelijk. Dit betekent dat op het moment dat er een modificatie moet komen voor 1 van de platformen dit altijd moet worden gedaan in de gedeelde broncode waardoor het lastiger is.
Nadeel van Appcelerator is dat het alleen maar iPhone en Android ondersteunt. PhoneGap is ook een goede oplossing aangezien het open source is en zo de programmeur meer controle en vrijheid geeft. Dit kan handig zijn als er iets mis gaat of als er een bug in het pakket zelf zit.

Native code voor elke platform geeft wel de vrijheid om elk apparaat optimaal te gebruiken, maar geeft veel overhead. Java ME geeft wel de vrijheid één oplossing te maken voor elk platform maar is weer zo globaal dat je het apparaat niet compleet kan gebruiken. Een ideale oplossing zou hiertussen moeten zitten. De verschillende RAD tools en server based oplossingen zijn te gelimiteerd om een applicatie in te maken. De kans is groter dat je met deze oplossingen in een situatie terecht komt waarbij je nieuwe mogelijkheden niet kunt implementeren.

Airplay SDK ziet er veelbelovend uit: het is snel en uitgebreid, werkt op bijna alle apparaten en er zijn al vele cross-platform applicaties meer gemaakt. Het is oorspronkelijk bedoelt voor games, dit zorgt er voor dat er veel is gedaan aan optimalisatie. Een nadeel is dat het niet de Native GUI van de smartphone kan raadplegen maar de bijgeleverde GUI is goed in elkaar gezet.

Om meer overzicht te krijgen heb ik alle pakketten in een tabel uit gezet met punten die belangrijk zijn voor de applicatie. Dit is te zien op bijlage C. Kandidaten voor de applicatie zijn vanuit hier gezien:

- PhoneGap
- Airplay SDK
- MoSync
- Rhodes
- Java met abstractielaag

Het Java met abstractielaag idee leek mij erg interessant, maar geen bewezen technologie en als nog erg complex en tijdrovend. Ik heb daarom gekozen het niet te gebruiken. Rhodes was een ander veel belovend pakket maar hiervan kreeg ik de ontwikkelomgeving niet goed geinstaleed. Verder zou het ontwikkelen via een console. Ik heb liever een IDE aangezien dat het schrijven van een applicatie erg versnelt.

Overgebleven zijn PhoneGap, Airplay SDK, MoSync waarmee ik een prototype gaat ontwikkelen.

4.3.3 Prototypen

PhoneGap

Het prototype maken met PhoneGap was niet eenvoudig. Het installeren van het pakket is redelijk te doen al moest er wel hier en daar wat aan configuratie files veranderd worden. Om het SDK werkend te krijgen met een IDE moeten ook alle SDK's van de smartphones die je zou willen gebruiken geïnstalleerd worden. Om dit allemaal met elkaar werkend te krijgen is een kleine uitdaging. Het was daarna wel erg eenvoudig een applicatie te maken. Een simpele HTML bestand in de goede directory plaatsen en op compileer drukken was genoeg.
Ik kwam er snel achter hoe ik met ' Sencha touch' een GUI kon maken die lijkt op de native GUI van de smartphone. Dit is natuurlijk wel minder snel dan een native GUI maar ziet er prettig uit.

PhoneGap leek me een goede mogelijkheid voor de cross-platform ClockWise applicatie. Maar ik ben hier op terug gekomen. Het GUI systeem werkt prima en is professioneel. De testen om een connectie te maken met de SOAP service zijn mislukt. Javascript heeft niet veel goede API's die dit mogelijk maken. Ook zelf een SOAP pakket maken en versturen over het HTTP protocol is mislukt. In javascript wordt het moeilijk gemaakt om cross-domein scripts te starten. Dit omdat het niet veilig is dat scripts bij informatie kunnen van andere domeinen. Er zijn mogelijkheden om hier omheen te gaan en ik heb verscheidene onsuccesvol geïmplementeerd, dit door onder andere bugs en incompatibiliteit van verschillende versies en verschillende gebruikte protocollen.

MoSync

De installatie van de MoSync SDK was heel simpel en het werkte meteen out of the box. Een simpele applicatie schrijven was niet heel moeilijk en de bijgeleverde voorbeelden kon ik met 1 druk op de knop laten compileren over meerdere platformen. Eigenlijks was het perfect. Totdat ik doorkreeg dat de GUI die er bij zat erg slecht was. Het zag er slecht uit en was niet doorontwikkeld. Eigenlijk zou de GUI van de ClockWise applicatie, de widgets, totaal opnieuw gemaakt moeten worden. Dit kost veel tijd. Ik heb niet het vertrouwen dat ik een GUI en een applicatie kan maken binnen de gestelde termijn.

AirPlay SDK

AirPlay was redelijk verbazend. Ik heb binnen een kwartier een succesvolle soap request naar ClockWise kunnen sturen met dit systeem. De installatie vermengt zich met Microsoft visual studio, een goede IDE. Het ontwikkelen ging erg snel en er was veel documentatie en voorbeeldmateriaal aanwezig om mij bekend te maken met de API's. Het GUI systeem werkt met een eigen formaat menu configuratie bestand. Dit bestand kan redelijk groot en onoverzichtelijk worden. Om een goed menu te maken gaat in het goed plaatsen van de informatie in dit bestand wel tijd zitten. Het menu is meteen schaalbaar over meerdere resoluties en reageert erg snel. Ik heb snel een prototype kunnen maken en heb besloten dit systeem te gebruiken. Het is het enige systeem tot nu toe waarvan ik zeker was dat ik de ClockWise applicatie zou kunnen maken. Het is een snel systeem, werkt over meerdere platformen, heeft goede en veel documentatie en voorbeelden, heeft naam gemaakt op de mobiele games markt wat spreekt voor de betrouwbaarheid en continuïteit en ik had het snelste een kleine prototype draaiend dat werkte zonder problemen.

4.4 Analyse Applicatie

Er zijn verschillend onderdelen en problemen die opgelost of uitgezocht moeten worden. Hieronder is het project hiërarchisch opgesteld. Er is hier snel te zien wat gemaakt moet worden en waar rekeningen mee gehouden dient te worden. De meeste onderdelen ga ik afzonderlijk behandelen. De vraag over de mobile markt en welke SDK we gaan gebruiken zijn beantwoord, ook zijn de vereiste functionaliteiten van de applicatie bekend. Nu nog de vraag hoe de applicatie technisch gerealiseerd kan worden. De ClockWise applicatie gaan we schrijven met de Airplay SDK in de programmeertaal C++, C++ is een complexe maar zeer sterke taal die veel gebruikt word in belangrijke realtime programma's.

- Informatie vergaren: Hoe ziet de mobiele markt er uit
- Onderzoek: Welke SDK/API te gebruiken
- Programmeren: De ClockWise applicatie
    - Soap connectie met de database
        - Encryptie van de gegevens
        - Compressie van de gegevens
        - Wat te doen bij verlies van internet verbinding
    - Weergave van de GUI
        - Dag, Week of Maand overzicht
            - Simpele en snelle manier van uren invoeren
            - Meta informatie invoeren (eenheden, reiskosten en dergelijke)
            - Ondersteuning grote database, met snelle navigatie tussen de projecten
        - Kalender
        - Pagina om instellingen te maken
            - Inlog gegevens
        - Informatie pagina
        - mogelijkheid om Weken in te leveren
    - Mogelijke versnellingen van
        - Caching
        - Slimme SOAP requests
- Gebruikers testen
- Documentatie
    - Broncode
    - Scriptie
    - Gebruikers Handleiding
    - Technische Handleiding voor programmeurs
        - Hoe werkt Exploitatie over de verschillende markten
        - UML
        - Specialisaties
    - Informatie scherm

4.3.1 SOAP connectie met de database

Er zijn verschillende aanpakken mogelijk om een connectie te krijgen met de bestaande SOAP server van ClockWise. Het is mogelijk een bestaande API tezoeken die SOAP kan praten om een snelle oplossing te maken, of ik zou zelf de SOAP XML data kunnen genereren en over de netwerk lijn kunnen sturen. Ik heb er voor gekozen om zelf de connectie te beheren en geen API hiervoor te gebruiken. Hier heb ik twee redenen voor. Tijdens het ontwikkelen van een prototype voor PhoneGap heb ik drie verschillende SOAP API's gebruikt. Allemaal heb ik ze niet aan de praat gekregen. Dit vanwege hele kleine verschillen in het SOAP protocol die ik in de API niet kon instellen. Ik ben toen veel tijd kwijt geraakt met een goede API zoeken, terwijl de test toen ik de connectie zelf beheerde in 15 minuten succes was uitgevoerd. Verder zit er bij de AirPlay SDK geen SOAP client. Terwijl ik wel specifiek gedrag wil van de applicatie, bijvoorbeeld, hoe handelen we de request af als het internet op de mobile applicatie weg valt. Om controle te houden over dit soort zaken heb ik er voor gekozen zelf de connectie te beheren. Het SOAP protocol is erg simpel dus heel veel meerwerk gaat dit niet opleveren.

4.4.2 Encryptie van de gegevens

De gegevens van de klanten van ClockWise kunnen gevoelig zijn. Aan welke projecten momenteel wordt gewerkt en hoeveel tijd en geld hier in zit mag niet op straat komen te liggen. Om de gegevens af te schermen zit er een wachtwoord op. Om in te loggen word er gebruik gemaakt van Basic HTTP authentication. De gebruikersnaam en wachtwoord word samen gecodeerd met een BASE64 algoritme. Dit is overigens makkelijk terug te coderen dus technisch gezien gaat het wachtwoord “plain text” over het netwerk. De informatie gaat dan nog steeds zonder Encryptie over het netwerk. Daarom wordt ook SSL encryptie gebruikt via HTTPS communicatie. De server ondersteund dit en het wordt ook gebruikt in de desktop applicatie.

4.4.3 Compressie van de gegevens

Het SOAP protocol gebruikt XML als data drager. XML is een goede en duidelijk data houdende bron maar niet heel erg efficiënt in zijn omvang. Om de meeste simpele data over te sturen kunnen XML bestanden erg groot worden. Voor complexe data wordt het bijna onmogelijk het naar een mobiele applicatie te sturen. Een oplossing hiervoor zou zijn om de XML data op de server te comprimeren en op de cliënt weer te decomprimeren. Zo kan er een grote hoeveelheid XML worden verstuurd zonder veel overhead. XML data is namelijk erg goed te comprimeren.

SOAP gebruikt HTTP om zijn informatie over te sturen. HTTP heeft op zijn beurt de mogelijkheid compressie te gebruiken tijdens de data overdacht. Als de ClockWise cliënt dit ondersteunt hoeft een grote hoeveelheid data niet zinloos te worden verstuurd. Het maakt de applicatie sneller en potentieel goedkoper voor de klant die betaalt voor zijn bandbreedte.

4.4.4 Wat te doen bij verlies van internet verbinding

Apple's App-Store accepteert geen applicaties die geen oplossing hebben voor dit probleem. Zij forceren een waarschuwing naar de gebruiker wanneer het internet weg valt en eisen dat de applicatie weer verder kan gaan wanneer de verbinding terug is. En dit is niet zo vreemd, tijdens lange reizen kan het internet vaak wegvallen wanneer het mobiele apparaat zijn best mogelijke connectie kiest over de verschillende protocollen. Bij de ClockWise Applicatie is een werkende internet verbinding erg belangrijk. Wanneer de gebruiker uren informatie invoert en het word niet opgeslagen is hij zinloos bezig geweest. Dus we moeten alles op alles zetten om dit goed te laten gaan. Informatie dient gecontroleerd te worden en opnieuw verzonden mocht er iets verkeerds gaan. En de gebruiker dient hier van op de hoogte gesteld te worden zodat ook hij belangrijke data kan controleren.

4.4.5 Weergave van de GUI

De applicatie gaat gebruikt worden door een hoop mensen met een verschillende achtergrond. Sommige mensen zullen een technische achtergrond hebben en anderen een totaal A-technische achtergrond. De applicatie zal daarom zo intuïtief mogelijk moeten zijn om te zorgen dat iedereen met de applicatie overweg kan. Geprobeerd zal worden veel ideeën en methodieken van de ClockWise website te gebruiken aangezien mensen daar al bekend mee zijn. Verder is een mobiel een gelimiteerd apparaat, zolang het niet nodig is dat een gebruiker teksten invoert moet hij dit ook niet hoeven doen. Teksten invoeren op mobiele apparaten werkt namelijk erg vertragend.

De huis style van ClockWise is Oranje en Wit met misschien een tintje groen.

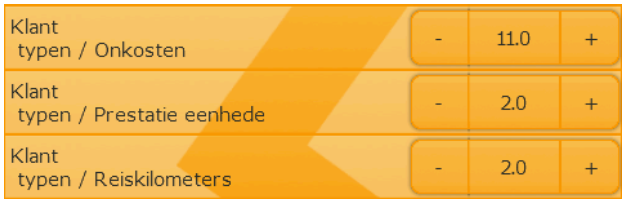
4.4.6 Dag, Week of Maand overzicht

Het is mijn doel niet te ver af te wijken van de ClockWise website. Als mensen dat gewend zijn is het niet netjes om ze compleet een ander interface te geven. ClockWise begint met een weekoverzicht. Aangezien een gehele week niet te presenteren is op een klein scherm als dat van een smartphone heb ik er voor gekozen een dag overzicht te maken. Dit dag overzicht laat de dag zien en pijltjes om snel naar andere dagen te gaan. Een speciale kalender pagina maakt het mogelijk om ook snel naar andere verder liggende dage te springen.

4.4.7 Simpele en snelle manier van uren invoeren

Uren schrijven zal een hoop gebeuren in de mobiele ClockWise uren registratie applicatie. Maar hoe maken we dit zo makkelijk mogelijk?

Om het mogelijk te maken om snel uren in te voeren zonder daarvoor een toetsenbord te gebruiken stel ik voor om naast op de uren te kunnen klinken om ze in te voeren met een toetsenbord ook plus en min knoppen toe te voegen om snel kleine mutaties te maken. Zie hiervan een prototype op Afbeelding 4.



Afbeelding 4: Prototype plus en min knoppen voor uren registratie

4.4.8 Meta informatie invoeren (eenheden, reiskosten en dergelijke)

Bij uren zitten ook meta data, zoals eenheden, reiskosten, reiskilometers en dergelijke. Ook deze data moet zo eenvoudig mogelijk kunnen worden ingevoerd. De plus/min knoppen van de uren registratie zijn misschien hier ook weer te gebruiken

4.4.9 Ondersteuning grote database, met snelle navigatie tussen de projecten

Sommige klanten van ClockWise hebben een enorm grote en complexe project structuur. De ClockWise applicatie zal elke structuur moeten kunnen ondersteunen. Ervaring met de ClockWise website en iPhone applicatie vertelt echter dat hier goed naar gekeken dient te worden.
Zo is er een database waar meer dan 1000 projecten in zitten waar uren op te schrijven zijn. De meest simpele SOAP request die ik kan doen, die alleen maar de project namen ophaalt geeft dan al een antwoord van 300kB. Dit is te groot om snel naar een mobiel apparaat te sturen. Door compressie over de SOAP XML data te gebruiken wordt de data al snel 15kB. Een grootte die zonder problemen naar een mobiel apparaat te sturen is. Het mobiele apparaat zal deze data dan nog wel moet uitpakken, en daar voor heeft het een groot stuk geheugen nodig.
Compressie van de data is een vereiste wil de applicatie ondersteuning kunnen bieden voor grote projecten. Als de 1000 projecten in 1 lijst getoond worden op een relatief klein mobiel scherm kan het erg onpraktisch zijn voor de gebruiker om het gewenste project te vinden om uren op te schrijven. Hier hebben we verschillende oplossingen voor bedacht.

- Een zoek veld om selecties te kunnen maken in de data. (gebruikt in desktop applicatie)
- een “recente projecten” knop die projecten laat zien waar eerder in de week aan gewerkt is. (gebruikt in desktop applicatie)
- Een 'voeg nieuw uur toe' wizard. Deze mogelijkheid laat de gebruiker eerst kiezen uit alle klanten, dan uit alle projecten, dan uit alle sub projecten van dat project. (gebruikt in iPhone applicatie)
- Een boom structuur waarbij klanten en projecten open te klappen zijn om subprojecten te laten zien.
- De gehele boom structuur waarmee je de huidige selectie niveau kan instellen. (gebruikt in de ClockWise website)

Het zoekveld is niet ideaal voor de mobiele applicatie, toetsenborden op mobiele applicaties zijn niet heel efficiënt voor gebruik, als het zonder kan moet het zonder gedaan worden.
De “recente projecten” knop is een goede optie voor het begin scherm. Zo zou de gebruiker meteen uren kunnen schrijven op projecten waar hij of zij vaak aan werkt. Misschien zijn er ook favorieten aan toe te voegen om dit scherm nog handiger te maken.
De 'voeg nieuwe uur toe' wizard is een goede methode om snel een uur voor een project in te voeren. Het zou niet de enige methode moeten zijn omdat het meer clicks verijst naar mate de project structuur complexer word.
Een boomstructuur weergeven die in en uit te klappen is, maakt het mogelijk in een paar klikken bij de gewenste project te komen net als de eerdere wizard mogelijkheid. Verschil is dat de structuur altijd zichtbaar blijft.
Een presentatie van projecten laten zien naar de geselecteerde project niveau vereist eerst een keuze uit een groot menu, iets wat de ClockWise website zich makkelijk kan veroorloven omdat het een groot scherm heeft. Voor de ClockWise mobiele applicatie zou dit niet werken.

In de mobiele ClockWise applicatie kies ik er voor om de recente projecten knop te gebruiken als handig 'home' scherm en bij grote projecten de mogelijkheid te bieden een projectstructuur open en dicht te klappen om snel uren te kunnen schrijven voor andere projecten.

4.4.10 Kalender

In een ideale situatie zullen de uren op een project geschreven worden op de dag dat de uren gemaakt worden. Toch gebeurt dit niet altijd op deze manier. De mogelijkheid uren bij te werken op andere dagen is daarom belangrijk. Vandaar dat er een snelle en simpele kalender moet komen met de mogelijkheid de datum waarop uren te schrijven te selecteren. Aangezien kalenders veel extra eigenschappen hebben, zoals alternatieve presentatie in schrikkel jaren en dagen, verschillende filosofieën bij het nummeren van weken, verschillende hoeveelheid weken in een maand en dergelijke is het handig als er een kant en klare kalender gevonden kan worden om te gebruiken die getest is en betrouwbaar. Deze kalender moet naast de mogelijkheid om dagen, maanden en jaren te kiezen ook weeknummers laten zien aangezien dat de belangrijke eenheid is in de ClockWise website en ClockWise desktop applicatie. Misschien is het meteen mogelijk op die locatie weken te kunnen inleveren zodat dingen niet dubbel zijn te vinden in de applicatie.

4.4.11 Pagina om instellingen te maken

Wanneer de applicatie voor het eerst opstart zal er gevraagd dienen te worden naar de inlog gegevens. De applicatie kan niets doen als deze gegevens niet correct zijn. Maar ook moet het mogelijk zijn natijd deze gegevens te veranderen. Een configuratie pagina waar deze verschillende instellingen kunnen worden gedaan is nodig. Er zijn twee methoden die we hiervoor kunnen gebruiken. De form methoden die bekend is van het internet; maak de gewenste veranderingen en druk op opslaan. Of de Apple manier; sla de wijzigingen op wanneer ze gemaakt worden. Ik zal voor deze applicatie de Apple manier gebruiken. Dit voornamelijk omdat het een intuïtieve methode is voor mensen die minder bekend zijn met webapplicaties.

4.4.12 Informatie pagina

ClockWise heeft aangegeven een informatie pagina in de applicatie te willen. Deze heeft ze ook voor de iPhone applicatie en is goed bevallen. Net als de iPhone applicatie zal deze pagina gevuld moeten worden met tekst via het internet zodat er op afstand nieuwe dingen bij toe te voegen zijn. De iPhone applicatie gebruikt hier een HTML renderer voor. Aangezien de SDK die ik gebruik geen HTML kan renderen zal ik een alternatieve methode moeten gebruiken. DE applicatie kan “plain text” presenteren of een klein gedeelte van de HTML tags implementeren.

4.2.18 Mogelijke versnellingen van applicatie

Mobiele apparaten zijn kleine computers, maar ze zijn niet zeer krachtig in vergelijking met desktop computers uit de zelfde tijdsperiode. Om toch te zorgen dat we het meeste uit de applicatie en uit het apparaat halen moeten we nadenken over het versnellen van de applicatie.

Caching

Als een gebruiker een project aanvraagt via de applicatie, zal de gebruiker waarschijnlijk een tijdje moeten wachten op antwoord van de server voordat hij verder kan gaan met bijvoorbeeld uren op dat project schrijven. Door de projecten te cachen kunnen we hier misschien winst behalen. Is de project informatie eenmaal is opgehaald, hoeft de applicatie dat niet nog eens te doen. Toch moet hier voorzichtig mee omgegaan worden. Het is mogelijk dat project informatie wordt veranderd via een externe bron, misschien de ClockWise website, waarna de mobile ClockWise applicatie verouderde informatie weergeeft.

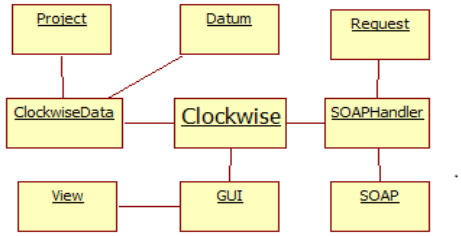
PreCaching

Wat ook een mogelijkheid is, is om gegevens die de gebruiker zou kunnen gaan aanvragen alvast op te halen. Zo kunnen we de gedetailleerde informatie van het huidige project al ophalen voordat de gebruiker er om vraagt. Of kunnen we alle projecten voor de volgende en vorige dag alvast inladen als de applicatie niets te doen heeft zodat deze meteen kunnen worden getoond na een druk op de knop zonder dat er eerst gewacht hoeft te worden op een soap request. Een nadeel is wel dat er veel data word verstuurd die niet zinnig of gewenst is. De ervaring van snelheid van de applicatie zal wel erg goed zijn.

5 – Ontwerp

5.1 Het Class Diagram

Op Bijlage E is het Class Diagram te zien van de ClockWise applicatie, hierin is de structuur en werking van de applicatie te herleiden. Een heel erg versimpelde versie hiervan is te zien op afbeelding 5. De belangrijke objecten zijn “ClockWise”, “ClockWiseData”, “GUI” en “SOAPHandler” Door te begrijpen hoe deze objecten samen spelen is te zien hoe de applicatie werkt.



Afbeelding 5: Versimpelde applicatiestructuur

ClockWise
De applicatie draait om de Class “Clockwise” deze class houdt de data bij, zet een GUI op het scherm en houdt zich bezig met het regelen van een goede loop van de ClockWise applicatie. De Class “Clockwise” is een eigen variant op de singelton design pattern. Dit maakt de Class bereikbaar in alle gedeeltes van het programma door middel van de “ClockWise::get()” methode.

ClockWiseData
De “ClockwiseData” class die onderdeel is van de “Clockwise” class functioneert als cache en globale opslag van alle project data, dit doet hij met behulp van de classes “Project” en “Datum”. Het behandelt ook de aanvragen van nieuwe data die de gebruiker zou willen hebben door de methode “loadDay()”.

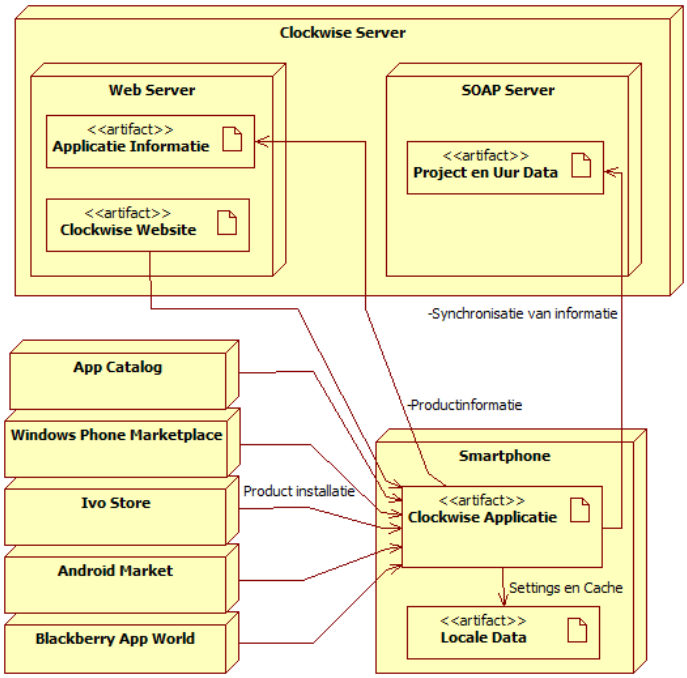
GUI
De “GUI” class is ook onderdeel is van de “ClockWise” Class. Deze is verantwoordelijk voor het tonen van de grafische interface. Hij bouwt dit op aan de hand van de huidige geselecteerde tijd “ClockWise::selectedTime” en de informatie die te vinden is in het “ClockwiseData” object. De “GUI” class heeft een “View” class, die op zijn beurt verantwoordelijk is voor het renderen van de huidige zichtbare pagina waar de gebruiker van de software zich bevindt. Het maakt gebruik van polymorfisme om het mogelijk maken alle objecten te updaten en aan te roepen alsof ze een “View” object zijn. Dit terwijl ze toch eigen mogelijkheden hebben die de base class “View” niet heeft.

SOAPHandeler
De “SoapHandeler” is onderdeel is van de “ClockWise” class. Hier zijn nieuwe “SOAP” objecten aan te vragen via de “getSOAP()” methode. Hier is gebruikt gemaakt van een factory design pattern.
Dit object geeft een nieuw SOAP object en houdt de status van de objecten bij om te zorgen voor een goede actieve connectie met de ClockWise Database. De “SOAPHandeler” class heeft ook de mogelijkheid om SOAP requests parallel of sequentieel uit te voeren. Sequentiële uitvoering van SOAP aanvragen is langzamer maar daarmee komt de informatie altijd op dezelfde volgorde terug in de applicatie. Parallelle uitvoer is sneller maar ook fout gevoeliger. Om een SOAP object aan te kunnen maken is er een “Request” Object nodig, dit object maakt ook weer gebruik van polymorfisme om een globale interface te geven tot een grote verscheidenheid aan mogelijke aanvragen aan de SOAP server. Een aanvraag zou er zo uit kunnen zien: “soapHandeler.getSOAP(Request_ID(userInfo.name, userInfo.passwd));”
Waarbij het “Request_ID” object een Child Class is van de“Request” base Class maar de mogelijkheid heeft om een gebruikersnaam en wachtwoord op te geven.

Gzip en Utils
De “Gzip” class geeft de mogelijkheid compressie te decoderen voordat de gegevens worden ingeladen in het “ClockWiseData” object. Er wordt hier gebruik gemaakt van HTTP 1.1 Gzip encoding. De “Utils” Class geeft een verzameling kleine en vaak gebruikte methoden en waarden die soms handig zijn.

5.2 Deployment diagram

Op Afbeelding 6 is het Deployment diagram te zien hier in is beschreven wat nodig is om de applicatie te kunnen gebruiken en waar de applicatie zich bevind.



Afbeelding 6: Deployment diagram

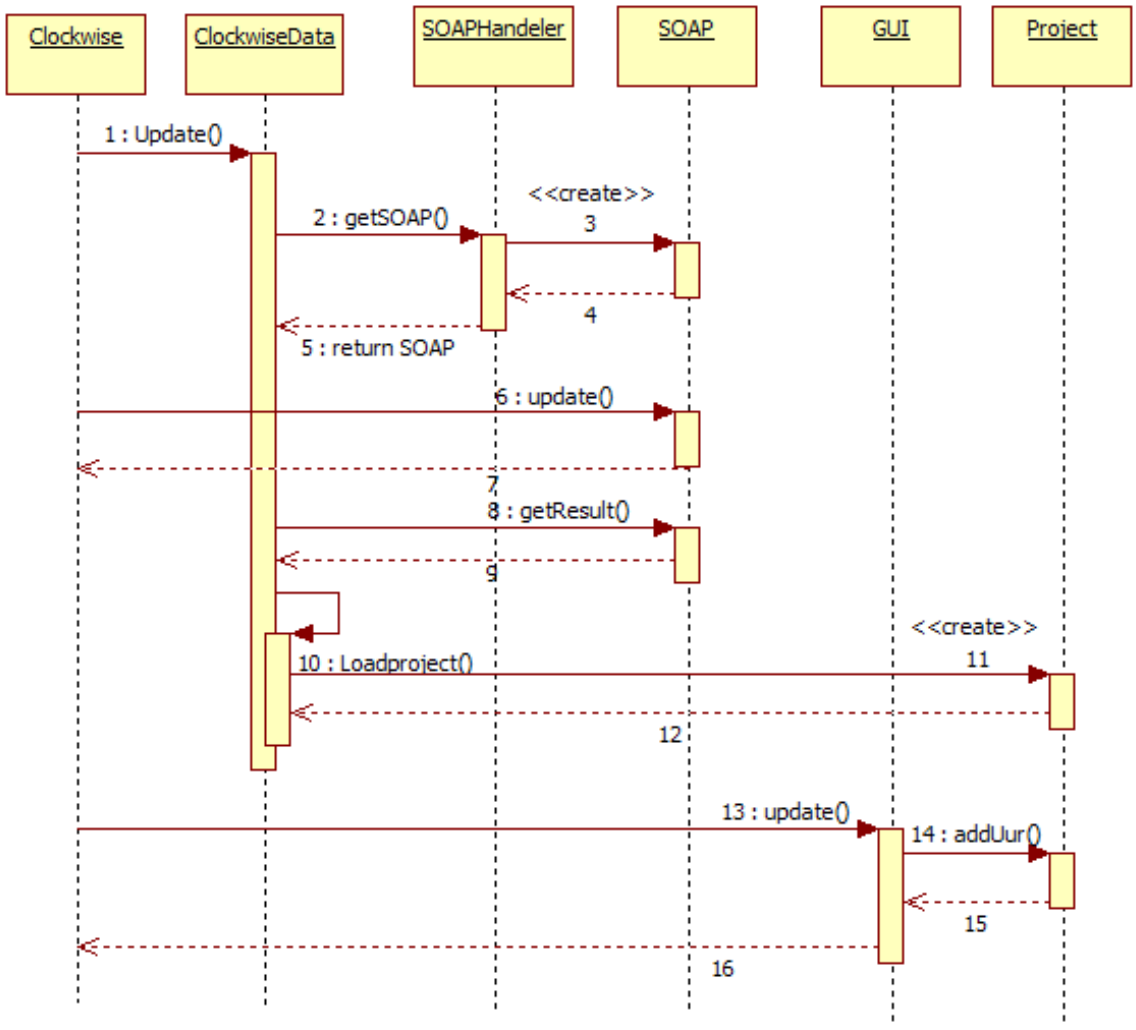
Installatie
De generieke ClockWise applicatie zou voor elk platform in zijn eigen applicatie market bevinden. (op afbeelding 6, links onder) Voor sommige mobiele platformen is het ook mogelijk de installatie te doen volgens de ClockWise website. Ook is het mogelijk op de ClockWise website links te maken die rechtstreeks naar de markt installatie van de ClockWise applicatie linkt. Deze link wordt natuurlijk automatisch aangepast aan de smartphone die de website bezoekt.

Smartphone
Wanneer de applicatie is geïnstalleerd van de markt of van de ClockWise website, is het te gebruiken. Maar tijdens het gebruik is een internet connectie nodig om informatie van de ClockWise servers te halen. Zelf kan de applicatie wel locale data op het apparaat opslaan. Denk bijvoorbeeld aan inloggegevens.

ClockWise Server
De smartphone applicatie gebruikt bepaalde diensten op de ClockWise server, de type smartphone die verbinding zoekt met de server maakt hier niet meer uit. De ClockWise applicatie gebruikt standaard protocollen om met de ClockWise server te communiceren, dit is hetzelfde voor alle mobiele apparaten. Zo zou de applicatie informatie kunnen opvragen om de informatie pagina te voorzien van tekst. Maar ook kan het uur een project informatie aanvraag en versturen.

5.3 Sequence diagram

Afbeelding 7 geeft een sequence diagram weer van de ClockWise applicatie. Hier word weergegeven hoe een aanvraag aan de SOAP server voor informatie over een project word afgehandeld.
Het Clockwise object houdt zich bezig met de verschillende onderdelen bij te houden. Dit doet het door periodiek update() methoden aan te roepen. Wanneer het ClockwiseData Object besluit dat er nieuwe informatie nodig is vraagt het bij punt 2 een nieuw SOAP object aan, die het bij punt 5 van de SOAPHandeler ontvangt. Dit SOAP object word in zijn bestaan onderhouden door periodieke update() aanroepen van het Clockwise Object. Wanneer het SOAP object klaar is met de informatie van het internet ophalen kan de ClockwiseData class de informatie opvragen, dat gebeurd hier bij punt 8. Het ClockwiseData object interpreteert de gegevens en maakt in het geheugen een Project object aan. Later kan hier door bijvoorbeeld het GUI object uren aan toegevoegd worden, dit gebeurt bij punt 14.



Afbeelding 7: Sequence diagram van SOAP request en Project update

6 – Realisatie

6.1 Globale fasering

Het project heeft een aantal verschillende fases. In de volgende tabel zijn ze weer gegeven. Verder in dit hoofdstuk ga ik deze fases toelichten.

Week nummer	Mijl Paal	Document / Product
5 - 6	Onderzoek: smartphones	Onderdeel voor scriptie
7 - 11	Onderzoek: Welke SDK / API te gebruiken	Onderdeel voor scriptie, UML
9 - 11	Succesvolle connectie met SOAP server	-
16 - 21	Professionele GUI	Ontwerp
11 - 23	ClockWise applicatie Versie 1.0	Gedocumenteerde broncode

6.2 Realisatie per fase

6.2.1 Onderzoek: Smartphones

In dit project heb ik voor het eerst een complexe smartphone applicatie gemaakt. Ik heb vele programma's geschreven dus weet dat ik de mogelijkheden heb om een applicatie te schrijven maar ik had totaal geen kennis van hoe de smartphone wereld er uit zag. Om te begrijpen waar ik mee bezig zou gaan, waar de valkuilen zouden kunnen zitten en wie de grote spelers zijn heb ik mij eerst ingelezen. Aangezien het een snelle markt is, waar in 2 maanden tijd alles anders kan zijn ben ik me gaan inlezen op het internet. Boeken zouden in deze situatie snel oude informatie bevatten.

Stappenplan
Om te beginnen heb ik me ingelezen in de smartphone wereld, welke spelers zijn er, hoe groot zijn ze. Wat zijn de nieuwe spelers, de oude spelers. Hoe snel groeien de verschillende bedrijven.
Maar ook hoe werken de mobiele besturingssystemen, wat voor SDK's hebben ze, welke mogelijkheden ondersteunen de huidige generatie aan smartphones.

Van hieruit ben ik gaan kijken naar welke smartphones de bezoekers op ClockWise.info gebruiken. iPhone kwam hier als grote winnaar uit. Maar dit is niet verbazend aangezien er al een iPhone applicatie is gemaakt voor ClockWise en de ClockWise programmeurs zelf allemaal een iPhone hebben. Android kwam direct daarna aanbod als we kijken naar de hoeveelheid bezoekers. De bezoekers informatie is gehaald uit google's analytics software die gegevens verzameld over het gebruik van het ClockWise domein. Deze software geeft betrouwbare informatie.

Maar wat ook belangrijk is: wat willen de gebruikers van ClockWise zelf zien? Een enquête vertelde dat de wensen van de klanten van ClockWise wezen naar interesse in een Android applicatie, en voor de andere grote namen was ook veel interesse. Deze enquête is te zien op bijlage D. De resultaten zijn te zien op Afbeelding 1. Dit was een kleine enquête gedaan onder de klanten die de nieuwsbrief ontvangen. We hebben zelf een selectie gemaakt van de voor ons belangrijke platforms. Dit zijn Nokia's Symbian, Blackberry, Android, iPhone en Windows Mobile.
De mogelijke antwoorden; “niet”, “een enkele keer “, “redelijk vaak”, “vaak”, “uitsluitend” zijn respectievelijk de waardes 0, 1, 2, 3, 4 gegeven om uit te vinden hoeveel interesse er was voor de bepaalde platformen.

Risico's

Bij het inlezen is er de mogelijkheid dat de informatie verouderd of verkeerd was. Op het internet is veel te vinden maar niet alles is juist. Ik heb geprobeerd kennis te vergaren van verschillende bronnen om een goed globaal idee van de marktwerking te krijgen.

Bij het kijken naar de bezoekers van ClockWise.info zijn weinig aanvragen geweest van de site door mobiele apparaten, je zou je kunnen afvragen hoe betrouwbaar de relatief kleine dataset is. Er zijn namelijk 536 aanvragen geweest voor de site door smartphones. Hiervan waren 482 afkomstig van iOS. In de zelfde tijds periode zijn ongeveer 30.000 bezoekers via een personal computer op de website geweest. 30.000 bezoekers geeft redelijk indicatie van gedrag op internet. De 54 mobiele bezoekers die niet gebruik maakte van iOS (Apple) zijn waarschijnlijk niet representatief, maar geven wel een indicatie.

Ook bij de enquête was er een kleine dataset. Er hebben 16 klanten de enquête ingevuld, zij representeren ongeveer 500 a 600 gebruikers. Ook zijn de waardes die ik de antwoorden heb gegeven niet sluitend. Hier is weer een indicatie te zien, maar niet een representatieve analyse van de wensen van de klant.

6.2.2 Onderzoek: Welke SDK / API te gebruiken

De vraag welke SDK te gebruiken was zeer belangrijk voor dit project. ClockWise wil graag een oplossing die werkt voor zoveel mogelijk klanten en daardoor voor zoveel mogelijk mobiele apparaten. Dit is niet iets wat standaard mogelijk is, zoeken naar de beste oplossing hierin mocht zeker wat tijd kosten. Ik heb het internet af gezocht om mogelijke kandidaten te vinden. Veel heb ik gevonden en kunnen verifiëren op een toegewijde wikipedia pagina met links naar crossplatform mobiele SDK's namelijk:

http://en.wikipedia.org/wiki/Multiple_phone_web-based_application_framework
http://en.wikipedia.org/wiki/Mobile_application_development

Stappenplan

Over dit onderzoeksproces zijn 2 iteraties geweest. Ik heb een aantal pakketten gezocht via het internet, ben ze gaan testen en heb de resultaten opgeschreven met het doel het best mogelijke pakket te vinden.
Toen ik mijn resultaten deelde met ClockWise werd mij gevraagd nog verder te gaan hierin. In de 2e iteratie kwam ik de wikipedia pagina's tegen die een erg complete lijst gaf. Ik heb de pakketten die ik miste van die lijst ook nog onderzocht en daarmee een redelijke indruk gekregen in de aanwezige mogelijkheden.

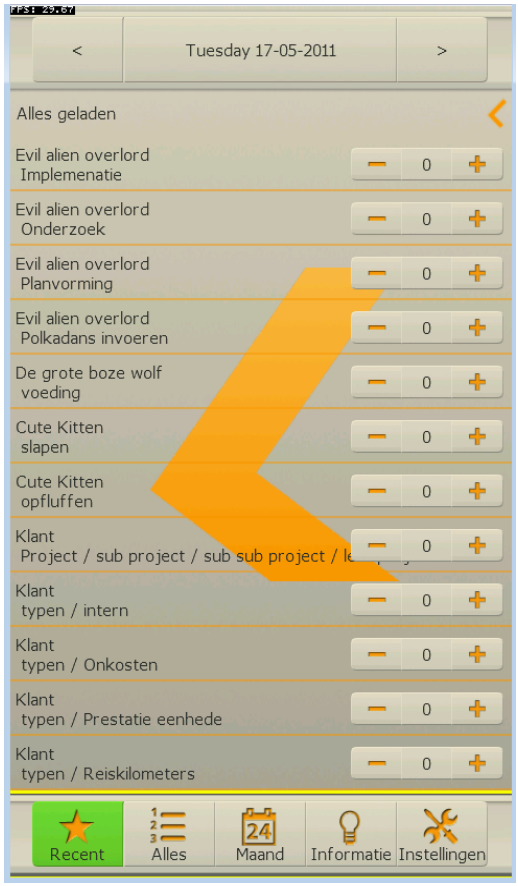
Afbeelding 8: Urenscherm GUI versie 1

Testplan / Testopstelling

Ik heb alle pakketten op verschillende eigenschappen bekeken. Hoe betrouwbaar is het, zijn er al veel succesvolle producten mee gemaakt, welke methode gebruikt het om een applicatie crossplatform te maken, heeft het een goed GUI, heeft het goede documentatie en voorbeelden, is het een actief onderhouden pakket, is het mogelijk SOAP of HTTP requests te doen, hoe duur is het pakket in aanschaf en vele andere factoren. Van alle pakketten heb ik een klein stukje tekst geschreven en de voor en nadelen genoteerd. Een aantal belangrijke eigenschappen heb ik per pakket in een tabel genoteerd (zie bijlage C).
Van de pakketten die mogelijk gebruikt zouden kunnen worden heb ik een test of zelfs een prototype applicatie gemaakt om dit te bewijzen.

Risico's

Er is zeker een risico geweest in dit proces, ik was op ten duur overtuigd dat PhoneGap het pakket zou worden waarmee ik de ClockWise applicatie ging maken. Maar het creëren van een prototype duurde heel lang. Ik ben veel problemen tegen gekomen die het onmogelijk maakten een SOAP request te doen. Het was een op webtechnologie gebaseerd pakket. Het zou via Javascript SOAP requests moeten kunnen maken. Maar de API's die ik gebruikte werkten allemaal niet of niet naar wens. Ook toen ik zelf begon SOAP pakketten te sturen door XML te genereren en over een HTTP lijn naar de server te sturen ging het mis. Ik heb weken besteed om uit te vinden waar alle problemen zaten en ik heb ze niet allemaal kunnen oplossen. Het grootste probleem was dat javascript niet de rechten heeft om contact te leggen met een andere domein dan waar het script staat. Aangezien de javascript zich bevond op “localhost” (de smartphone zelf) en een connectie zal willen maken naar ClockWise.info ging het mis. Er zijn methodes om hierom heen te gaan, ik heb daarvan meerdere geïmplementeerd maar deze werkten niet in alle situaties. Ook heb ik de SOAP server hier en daar moeten aanpassen om een connectie te kunnen maken. Dit werd zo omslachtig dat ik mijn verlies heb moeten toegeven en een ander pakket ben moeten gaan zoeken. Mijn eigen ego en wilskracht waren in dit geval het risico. Ik moest en zou het werkend krijgen, misschien was een beetje bescheidenheid hier een goede eigenschap geweest.



Afbeelding 9: Urenscherm GUI versie 2

6.2.3 Succesvolle connectie met SOAP server

Een connectie maken met de SOAP server is meer een persoonlijke mijlpaal. Hieraan kan ik zien dat de applicatie voldoet voor de meest belangrijke functie. Als dit goed gaat, en er zit een goed werkende GUI in hebben we al snel de gewenste ClockWise applicatie. In alle prototypen heb ik deze connectie geprobeerd te maken.

6.2.4 Professionele GUI

Stappenplan

de GUI van de applicatie wordt gemaakt met de CiwUI class van het AirPlay SDK. Deze GUI neemt het grootste gedeelte van het werk op zich. Het heeft knoppen, menu's en schaalt automatisch over verschillende resoluties. Ik heb eerst een simpele layout ontworpen (zie afbeelding 8) waarin de applicatie gemaakt is. Hoe deze er uitzag was niet heel erg belangrijk. Belangrijker was dat de flow van het programma goed was en dat alle gewenste operaties te doen zijn met zo min mogelijk gebruikers acties. Aan het eind van het project is een grafisch mooier GUI ontwikkeld die er beter uit ziet en dezelfde workflow houdt. Een werkende versie hiervan is te zien op afbeelding 10.
Ideeën die nog mogelijk geïmplementeerd kunnen worden zijn: Het GUI themeable maken, instelbare kleuren en een hoge contrast mogelijkheid voor mensen die slecht zien.

Risico's

Aangezien het GUI wordt ontwikkeld door de zelfde persoon als die de broncode schrijft bestaat de mogelijkheid dat er onhandige beslissingen worden genomen. Zo zou er een situatie kunnen ontstaan waarin het programma makkelijker te schrijven is als ik iets zou laten liggen wat het gebruikers gemak vergroot.

Een ander risico is de resolutie, de GUI kan zich aanpassen aan elke resolutie, dat is ingebouwd in de toolkit. Maar dit belooft niets. Er zijn zoveel verschillende apparaten die ik probeer te ondersteunen dat ik nooit met zekerheid kan zeggen dat een tekst die op de meeste apparaten leesbaar dat op alle apparaten is. Ik doe mijn best om de theme's die ik maak zo schaalbaar mogelijk te maken, maar bij de gebruikers testen zullen we er pas achter komen of dit ook in de grote verscheidenheid aan apparaten goed gaat.

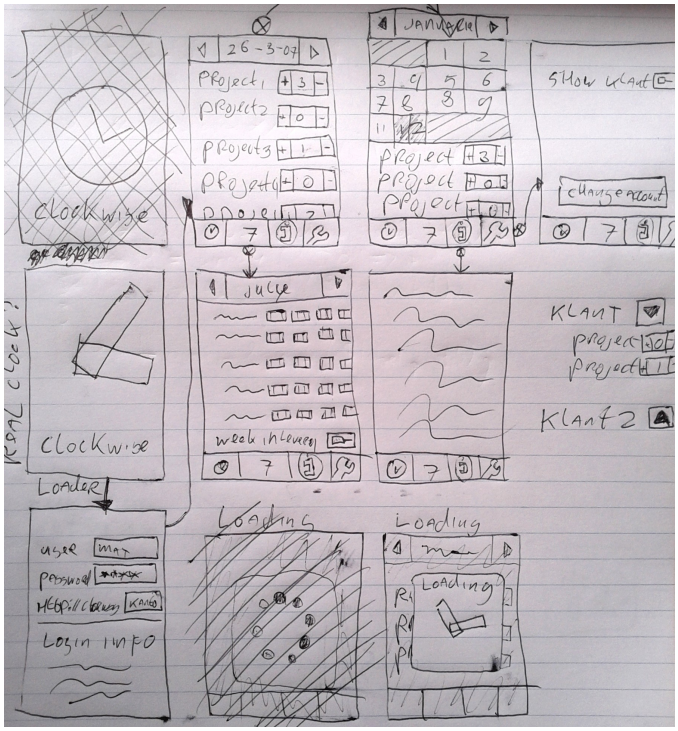
Ook is er een verschil in renderen van portret en landschap mode, de software kan herkennen hoe de gebruiker het apparaat vasthoudt en zijn layout hierop aanpassen. Wel dient dit ook op dezelfde manier goed te gaan over de verschillende apparaten.

Afbeelding 10: Maandscherm GUI versie 3

6.2.5 ClockWise Applicatie Versie 1.0

Stappenplan
De ClockWise applicatie is begonnen als een prototype waarin een SOAP request werd gedaan. Hierna is volgens een iteratief proces elke versie voorzien van nieuwe mogelijkheden en betere oplossingen dan de vorige versie. De wensen van de MoSCoW zijn langzamerhand geïmplementeerd zodat elke versie dichter bij het gewenste eindresultaat kwam.

Vele onderdelen hebben onderling ook verschillende versies gehad naarmate de applicatie groter groeide. Zo was het eerst noodzakelijk dat voor een SOAP request de XML data werd mee gegeven. Een versie later werden gedeeltes van de XML intern gegenereerd en in de laatste versie waarin alle informatie beschikbaar is wordt nu alle XML gegenereerd en is de noodzaak van XML bijna geheel uit de eigen API gehaald. Op de UML van de applicatie (zie bijlage E) is in de Class “Request” nog een methode “getXML()” te zien, deze word intern gebruikt om de gegenereerde XML door te geven aan een SOAP object. Ook heeft het SOAP object structurele veranderingen ondergaan. Aangezien de SOAP connectie een belangrijke eigenschap is van de applicatie werd hij bij elke nieuwe functionaliteit steeds groter. Om de code overzichtelijk te houden en om mijzelf te houden aan het object georiënteerde denksysteem heb ik besloten het object op te splitsen in verschillende kleinere objecten. Hier komen de “Request” objecten vandaan.



Afbeelding 11: Een prototype GUI is ook op papier te doen

Een applicatie is, globaal gezien een systeem waarbij data word opgevraagd, verwerkt en terug gezet. De ClockWise applicatie haalt het grootste gedeelte van de data van het internet en moet het daarna verwerken om op het scherm neer te kunnen zetten. Ik wou graag caching van de data toepassen om de applicatie te versnellen. Ik heb daarom besloten een complexe intern datatype te maken dat ik dan later voor caching kan gaan gebruiken als dat nodig blijkt. Dit naarmate duidelijk zou worden wat de voordelen en nadelen hiervan zouden zijn. Dit datatype wordt bewaakt door de “ClockWiseData” class die de data opslaat in “Project” en “Datum” objecten. Caching van de data kan aan en uit gezet worden en is mee te nemen in het gebruikers onderzoek.

Risico's

Een goede connectie met de SOAP server is altijd een risico geweest, de applicatie staat en valt bij zijn mogelijkheid de gewenste data snel van de server te krijgen en te verwerken voor de gebruiker. Het PhoneGap prototype werd onmogelijk gemaakt door dit risico. Het is maar goed dat ik een vooronderzoek gedaan heb om dit soort risico's te vermijden. De AirPlay SDK heeft geen ingebouwde ondersteuning voor het SOAP protocol. Wel heeft het een HTTP class waarmee over het internet gecommuniceerd kan worden. Het implementeren van de SOAP protocol was relatief gemakkelijk te doen in de AirPlay SDK. De mogelijkheid om data gecomprimeerd op te sturen was moeilijker maar maakte de applicatie aanzienlijk sneller. Encryptie van de data moet nog geïmplementeerd worden.

De GUI is een ander belangrijk punt geweest waarbij de applicatie kan slagen of falen. Ook op dit punt heb ik een hoop aandacht besteed in het vooronderzoek wat goed van pas is gekomen. De AirPlay SDK heeft geen Native GUI, hiermee bedoel ik de GUI toolkit die bij de telefoon zelf hoort. Maar het gemis werd goed gemaakt door een eigen GUI systeem dat bij AirPlay bijgevoegd was. Deze GUI voldoet voor de applicatie, al vind ik het jammer dat het onmogelijk is om teksten rechtstreeks in een widget toe te voegen met een on-screen toetsenbord. Inplaats daarvan springt er een pop-up open die vraagt om de tekst invoer. Ik heb geen problemen gehad met het GUI al nam het wel meer tijd dan verwacht om te wennen aan de configuratie bestanden.

De GUI van de AirPlay SDK heeft zelf geen kalender widget, dit was een moeilijk punt. Nu is een kalender widget prima zelf te schrijven maar dat neemt wel een hoop tijd in beslag. De kalender is een belangrijk punt om te navigeren naar de te schrijven uren. Het mag niet zo zijn dat over een jaar verkeerde dagen worden aangegeven,o f de maand een verkeerd aantal dagen heeft. Gelukkig zat er bij de voorbeelden een implementatie van een kalender die ik heb kunnen aanpassen voor gebruik in de ClockWise applicatie. Aan deze kalender heb ik weeknummers toegevoegd aangezien dat een belangrijke tijds eenheid is binnen het ClockWise pakket. Hier waren nog wel problemen. Er zijn verschillende standaarden die bepalen hoe de weeknummers lopen. Vooral in de definitie van wanneer de eerste week van het jaar begint. Ik heb een paar standaard formules geïmplementeerd maar ik was niet tevreden over de resultaten, ze liepen niet synchroon met de web-based ClockWise toepassing. Een Student die ook een afstudeer stage volgt bij ClockWise heeft dit probleem op zich genomen en een stuk C++ code geschreven die wij nu beide in onze applicatie gebruiken. Hij in de desktop applicatie en ik in de mobiele applicatie.

7 – Eindproduct

7.1 Resultaat

Het product is nog niet geheel ontwikkeld, er zullen nog een paar ontwikkel iteraties volgen en er zal nog functionaliteit bij komen. Er is op het moment van het schrijven van deze scriptie wel een geheel werkend product. Op bijlage B is op het MoSCoW document te zien wat op dit moment geïmplementeerd is. Groene punten zijn geïmplementeerd en werkend. Zwarte punten moeten nog gedaan worden en rode punten gaan zeker niet meer gedaan worden.
Het urenscherm is te zien op afbeelding 12.

Afbeelding 12: Urenscherm GUI versie 3

Menu
De menu knopen onderin het urenscherm verwijzen naar de verschillende pagina's. De eerste twee knoppen laten het dagscherm zien. De knop “Recent” laat daarin alleen de projecten zien waar recentelijk uren op geschreven zijn. De knop “Alles” laat alle projecten waar mogelijk uren op te schrijven zijn op die dag zien. De knop “Maand” laat de kalender zien (te zien op afbeelding 10) De informatie knop is nog niet geïmplementeerd, maar dit is een redelijk makkelijke pagina die alleen een tekst zal moeten laten zien. De instellingen pagina is te zien op afbeelding 13.

Afbeelding 13: Instellingen GUI versie 1

Projecten
Op het urenscherm worden de projecten laten zien. Op deze projecten zijn uren te schrijven door op de plus of min knop te drukken. Ook kan er op het uur zelf gedrukt worden om het met het toetsenbord in te voeren. (ook handig voor halve uren en dergelijke.) Wanneer op het project zelf word geklikt word er een project informatie dialoog geopend. (zie afbeelding 1 op bladzijde 5) Hier is extra informatie aan een project toe te voegen. Door op de datum te drukken word er ook naar de kalender gegaan.

Kalender
Het kalender is ook volledig geïmplementeerd. (zie afbeelding 10 op bladzijde 45) Door op een dag in de kalender te drukken kom je op de urenscherm op de gekozen datum. Ook kan je op de kalender zien of weken zijn ingeleverd en goedgekeurd. Op afbeelding 10 is te zien dat week 17 is afgekeurd. Door op de week te drukken kan je de week inleveren. Ook kan je hier op de datum klikken om in het maand selectie dialoog te komen.

7.2 Evaluatie

Het grootste gedeelte van de wensen van ClockWise is geïmplementeerd, er missen nog wel een paar dingen maar die gaan zeker nog komen.

De applicatie is nog niet helemaal stabiel, er zijn momenteel nog 2 bugs die de applicatie kunnen laten vastlopen. Als er heel snel meerdere connecties met de SOAP server gedaan word, kan het voorkomen dat er geen network sockets meer beschikbaar zijn op het systeem. Dit laat het communicatie gedeelte vastlopen waarbij het mogelijk is dat er oneindig lang op antwoord van de server gewacht wordt. Dit probleem is te verhelpen door een extra controle in te bouwen in het systeem. Ook is er een andere maar onbekende situatie die voor het zelfde gedrag zorgt. Hopelijk verdwijnt de 2e bug automatisch wanneer de eerste is verholpen. Gelukkig komen deze Bugs niet vaak voor in normaal gebruik van de applicatie.

Verder is er theoretisch nog het probleem dat er een gelimiteerd stuk geheugen beschikbaar is in de applicatie. Wanneer deze vol zou lopen zou er onstabiel gedrag voor kunnen komen. Ik heb het nog niet mee gemaakt maar ik zou deze situatie nog wel moeten simuleren om zeker te zijn dat er niets mis gaat.

7.3 Conclusie

De applicatie voldoet nu aan de belangrijke eigenschappen, de Must's van de MoSCoW zijn geïmplementeerd. De applicatie werkt goed, de vraag van ClockWise is opgelost en ik ben er zelf tevreden over. Er moeten nog wat kleine dingen gebeuren, maar daarin verwacht ik geen grote problemen. Het is nu nog voornamelijk de scherpe randjes weg halen, een proces dat na de creatie van elke applicatie plaatsvindt.

De echte test wordt de gebruikers evaluatie die nog moet plaatsvinden. Ik ben benieuwd hoe de applicatie door de gebruikers wordt ontvangen. En ook of naar aanleiding daarvan nog een hoop moet worden veranderd of dat de applicatie dan al geheel naar de wens van de klant is.

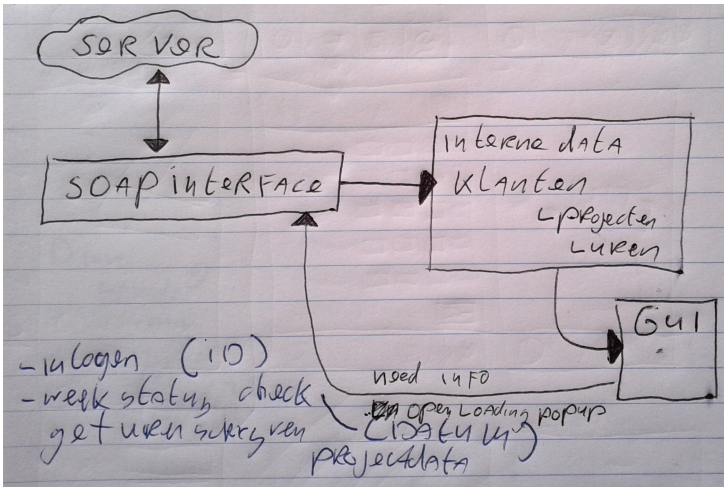
Een ander aspect van de applicatie is de continuïteit. De AirPlay SDK staat sterk en gaat nog lang blijven bestaan en nieuwe versies uitbrengen. Dit omdat de AirPlay SDK gebruikt wordt door een groot aantal bedrijven voor een groot aantal applicaties. De broncode van de mobiele ClockWise applicatie is voor een groot gedeelte voorzien van commentaar die de werking van de code uitlegt. Ook de UML diagrammen en documentatie helpt mee met zorgen dat zelfs zonder mij de applicatie door ontwikkeld kan worden.

8 – Proces en planning

8.1 Project aanpak

Projectmethodiek
Voor de ClockWise applicatie heb ik geen standaard softwareontwikkelmethode gebruik. Mijn reden hiervoor is dat de meeste methoden ontwikkeld zijn voor groepen die werken aan een groot project. Extreem programming, Scrum, Agile en Prince2 zijn goede voorbeelden hiervan. Deze methodieken zijn ontwikkeld om problemen op te lossen die voorkomen in het ontwikkelen van software. Dit vaak door vooruit te plannen zodat problemen opgelost kunnen worden voor dat ze ontstaan. Een groot probleem van het ontwikkelen van software is de complexiteitscurve, hoe complexer een programma wordt hoe moeilijker het is om nieuwe mogelijkheden toe te voegen. Vaak word deze curve gezien als exponentieel. De voorgenoemde methodieken geven methoden de complexiteitscurve aan te passen zodat grote projecten mogelijk worden. Om te zorgen dat ik zonder gebruik van een standaard softwareontwikkelmethode toch een succesvol product kan creëren gebruik ik aspecten uit verschillende softwareontwikkelmethodes.

Voordat ik ben gaan programmeren heb ik tekeningen gemaakt van een interne structuur van het programma. Ook heb ik tekeningen gemaakt van een potentiële GUI en heb ik datatypen uitgedacht en uitgetekend. Om later in het ontwikkelproces nog de mogelijkheid te hebben nieuwe functionaliteit toe te voegen heb ik het programma opgedeeld in verschillende onderdelen die elk haar eigen problemen oplossen, dit volgens het object georiënteerde gedachtegoed. Tijdens het programmeren heb ik altijd pen en papier bij te hand gehouden om problemen of nieuwe ideeën aan een todo lijst toe te voegen, dit ter vergelijking met een scrumm backlog. Na een software iteratie heb ik een UML gegenereerd van de applicatie wat mij heeft geholpen de structuur te bewaken die beoogt was voor dit project. Ik heb de flexibiliteit van “Cowboy Coding” gebruikt om het mogelijk te maken tot innovatieve oplossingen te komen. De term “Cowboy Coding” wordt als controversieel gezien. Er zijn nadelen aan deze methode die voornamelijk aan het licht komen bij onervaren programmeurs, dit omdat het niet verplicht om software van te voren te plannen. Het plannen van structuur is voor grotere projecten wel een must. In andere woorden, ik heb plannings aspecten genomen van de grote softwareontwikkelmethode en de flexibiliteit van “Cowboy Coding” om tot een succesvol project te komen.



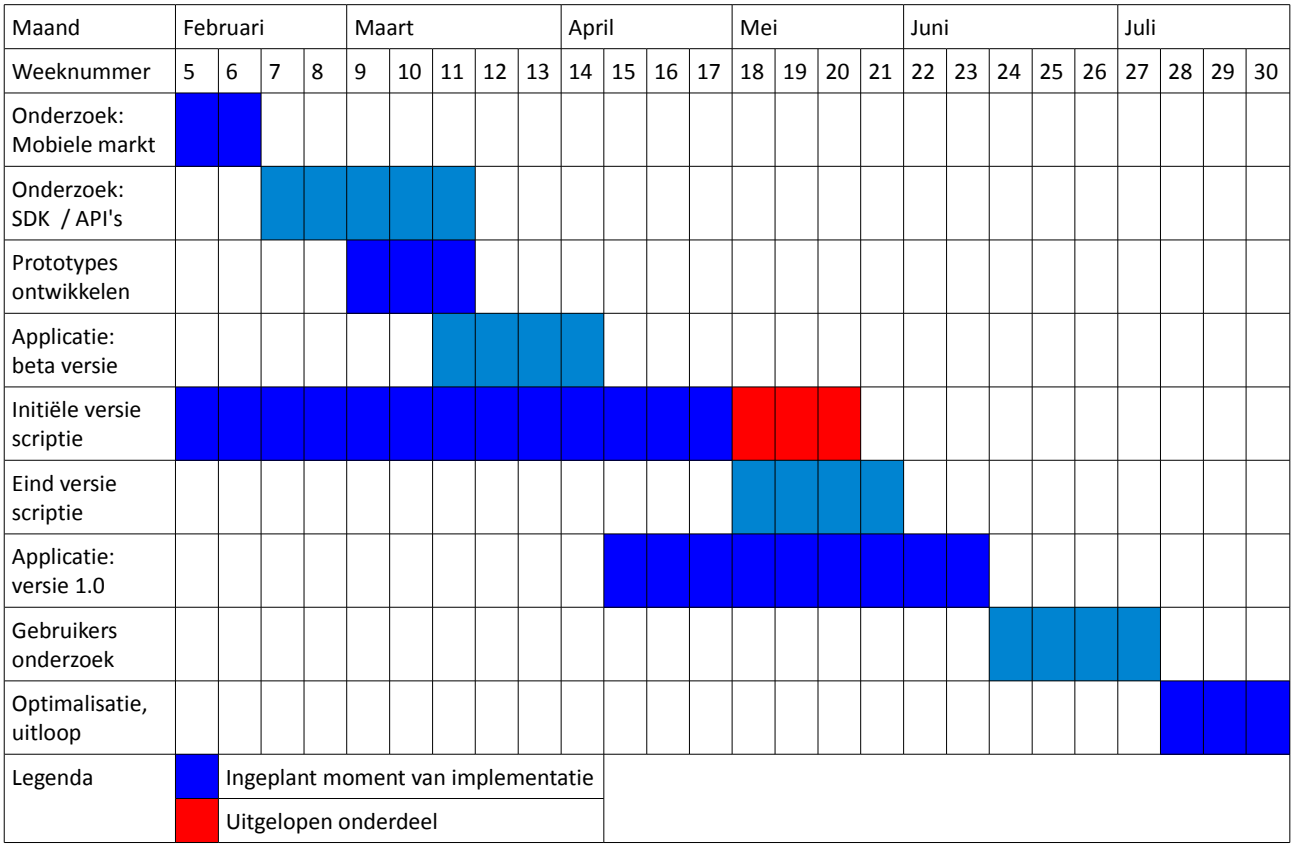
Afbeelding 14: Soms moet je gewoon ouderwets op papier ideeën uittekenen

Iteratief proces
Tijdens het ontwikkelen van software is mij opgevallen dat een iteratief proces prettig werkt. Elke iteratie genereert een nieuwe versie van de software. Bij elke iteratie los ik de problemen op die zijn voorgekomen uit de vorige versie waarna ik nieuwe functionaliteit toevoeg aan de applicatie. Een standaard iteratie heeft gemiddeld 3 dagen gekost. Bij elke iteratie word er dichter bij een eindproduct gekomen, ook goed voor de mentale drive om een applicatie af te maken.

strokenplanning
De stroken planning is gemaakt en bijgehouden in Open Office. De stroken planning is niet regelmatig bijgehouden maar gebruikt als een leidraad van de dingen die gedaan moeten worden. Tot nu toe is er alleen uitloop geweest in de initiale versie van de scriptie.

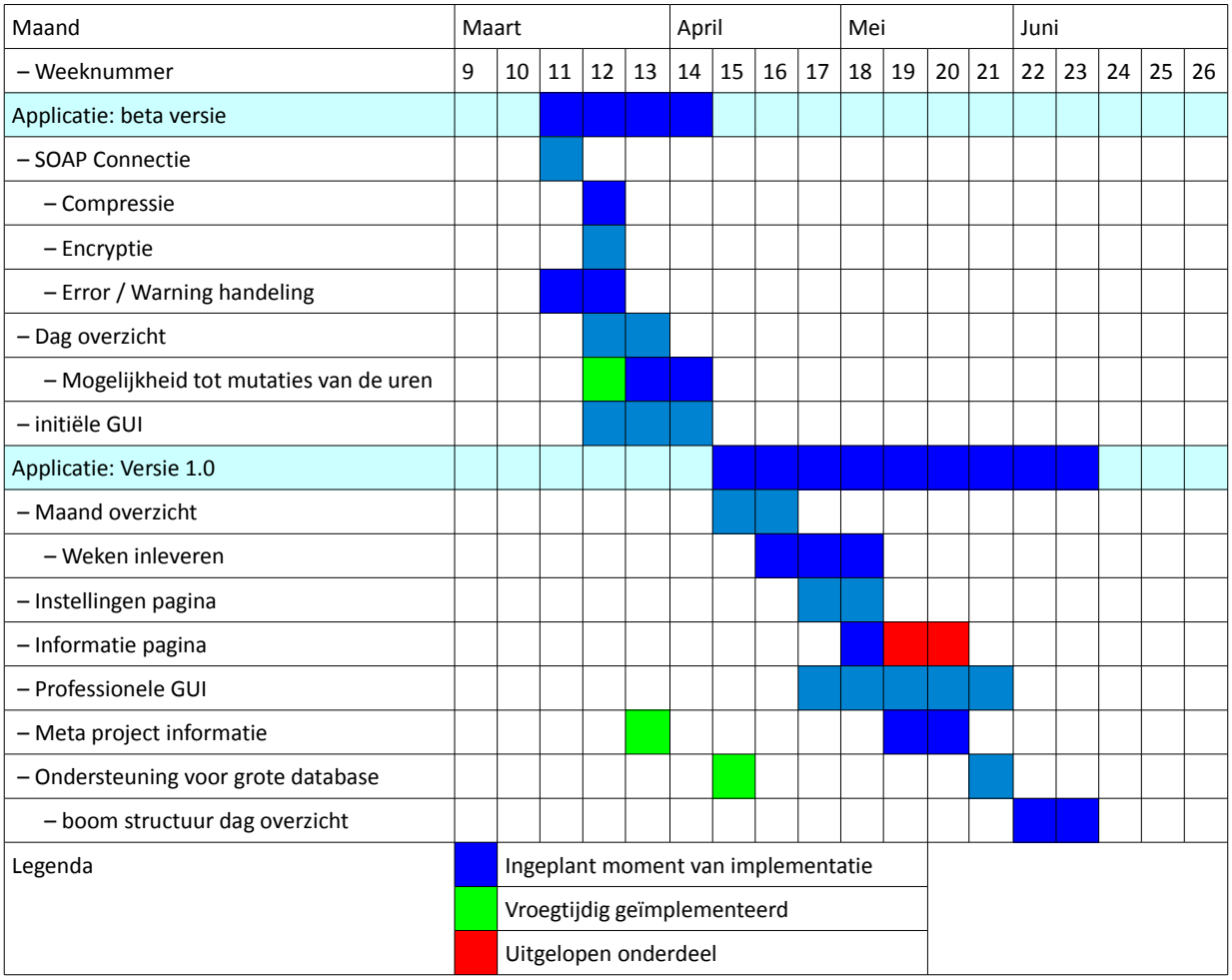
8.2 Strokenplanning

Project verloop
Ik heb het verloop van het project en de creatie van de applicatie opgesplitst. Dit aangezien de details rond de applicatie veel hoger zijn dan de loop van het hele project. Voor beide heb ik een plan gemaakt om te gebruiken als leidraad en daarmee kan zorgen dat ik niet te lang bezig ga met een onderdeel waardoor het geheel in geding komt. Ik heb altijd netjes aan de planning kunnen voldoen. Het creëren van de initiële versie van de Scriptie en de creatie van de informatie pagina is tot nu toe alleen uitgelopen. Op de applicatie zelf heb ik in het project altijd iets voorgelopen dus ik kan deze extra tijd opvangen.



ClockWise Applicatie

De ClockWise applicatie heeft een grote hoeveelheid aan onderdelen waarvan sommige onderling afhankelijk zijn, zo zou er geen dagoverzicht gemaakt kunnen worden zonder eerst de informatie die gepresenteerd dient te worden van de SOAP server te krijgen. Om voor een goede loop van het project te zorgen is gepland welke onderdelen op welke volgorde gedaan dienen te worden. Ook hier ben ik vrij omgegaan in de implementatie van de onderdelen. Ik heb me niet geforceerd me perfect aan mijn eigen planning te houden en heb de vrijheid genomen met sommige onderdelen langer bezig te zijn terwijl anderen in een paar uur al geïmplementeerd waren. Over het algemeen heb ik me prima aan de planning gehouden. Door het uitlopen van de initiële versie van de scriptie is het creëren van de informatie pagina uitgelopen, wel is de Professionele GUI al voor het grootste gedeelte ontwikkeld dus daar valt weer tijd te winnen. De Meta project informatie is al geïmplementeerd al moet daar nog een controle in komen welke meta informatie wel of niet gepresenteerd dient te worden. Dus ook hier komt weer tijdswinst vandaan. Op dit moment loop ik op de meeste punten voor op schema.



8.3 Project evaluatie

Ik heb relatief weinig aan planning gedaan, de planning die ik heb gebruikt was in grote lijnen en deze planning heeft prima gewerkt in dit project. Ik weet uit ervaring dat deze vrijheid niet mogelijk is in een project waar meerdere mensen aan het zelfde softwarepakket werken. Vooral omdat er dan veel overhead verschijnt in het implementatie proces. De planning die ik heb gehad, heb ik wel ervaren als een fijne leidraad om te zorgen dat ik overzicht heb kunnen houden in de staat van de applicatie en om duidelijk te zien hoever ik was in de totale implementatie.

9 – Reflectie

9.1 Persoonlijke Reflectie

Als ik terug kijk naar mijn functioneren in het project vallen mij zeker een aantal dingen op. Ik kan goed programmeren en heb daar een hoop ervaring mee. In dit project ben ik tegen een probleem aangelopen dat ik niet kon oplossen tijdens het ontwikkelen van een prototype van PhoneGap. Ik heb hier te veel tijd aan besteed aangezien het mijn ego raakte. Ik zou en moest het probleem oplossen. Dit is mij nog niet eerder gebeurd, maar mijn eigendunk ten opzichte van mijn vaardigheden zaten in de weg van mijn professionele houding ten opzichte van het project. Ik had na een paar dagen kunnen zien dat PhoneGap gewoon niet geschikt was als SDK voor de ClockWise applicatie, in plaats daarvan ben ik een week bezig geweest met alternatieve methoden in PhoneGap proberen en implementeren. Voor de loop van het project ben ik niet voorbij de gezette tijd gegaan voor het maken van prototypen dus er is niets mis gegaan. Wel is het voor mezelf een goede indicatie dat ik mijn kennis op programmeer gebied nog best mag aanvullen door een gezonde hoeveelheid bescheidenheid zodat het doel niet uit oog word verloren.

Op sommige momenten heb ik uitstellend gedrag ondervonden. Zoals bij de scriptie, een nieuwe functionaliteit aan de applicatie toevoegen was een stuk interessanter dan een stuk tekst schrijven. Door hier minder wilskracht op in te zetten is de creatie van de scriptie wel vertraagd. Ik weet dat vroeger wilskracht een slecht punt van mijn karakter was, maar ik ben er erg in gegroeid en voorzie hierin geen problemen in de toekomst.

Mijn opdrachtgevers zaten in de zelfde kamer, aan de communicatie was niets mis, ik was goed op de hoogte van de wensen. Als ik zo terug kijk denk ik wel dat ik op meer momenten officiële vergaderingen had kunnen inplannen om mijn voortgang in het project te laten zien.

Het commercieel aspect is in dit project ook niet duidelijk tot uiting gekomen. Ik had de vrijheid om zelf het project te plannen naar wens en heb geen problemen met hoe ik dit gedaan heb. Wel was het beter geweest als ik meer aandacht had besteedt aan dingen als uren verantwoording. Tijd is geld en ik heb de neiging gehad om gewoon een goede applicatie te maken zonder rekening te houden met tijd en geld. Nu was de stelling vanuit ClockWise “Als het maar af komt” maar dan nog had ik hier een professionelere houding kunnen nemen. Goed punt om mee te nemen naar toekomstige projecten.

Ik weet dat mijn technische kennis altijd sterk aanwezig is, en ik ben tevreden met hoe ik de problemen die ik ben tegen gekomen heb opgelost. Ik ben ook tevreden hoe ik een groot project tot een goed einde heb kunnen brengen. Ook al zijn er op dit punt zeker nog een aantal dingen die ik moet afronden.

9.2 Competentie gebaseerde Reflectie

De 4 generieke competenties van het profiel van de bachelor of engineering:

- Inzicht krijgen
- Ontwerpen
- Plannen
- Uitvoeren

In de fase van het *inzicht krijgen* heb ik me onder andere verdiept in mobiele markt. Ik had voorheen geen ervaring met wat daar zich afspeelt en wie de spelers waren. Ook heb ik inzicht verkregen in de wensen van ClockWise. De opdracht stond op papier, maar door in een gesprek het over de opdracht te hebben heb ik meer informatie gekregen over wat de opdracht voor hun inhoud. Ik ben tevreden hoe ik deze competentie heb benut, ik was aan het begin van de opleiding mediatechnologie wel de persoon die meteen begon met het uitvoeren zonder de eerdere stappen te doorlopen.

Voor de *ontwerp fase* heb ik een hoop respect gekregen, vele eigen projecten die ik vroeger gemaakt heb zijn hier gefaald. Als in het begin van het project niet een structuur word gecreëerd heb je niet de houvast op nieuwe onderdelen toe te voegen aan de applicatie. Het proces van een applicatie uitdenken, de datatypes ontwerpen en alvast bedenken waar problemen kunnen voorkomen is een leuk proces. Het is uitdagend en heeft bij grote projecten veel voordelen. Ook in deze competentie kan ik mezelf vinden.

De *plannings fase* is denk ik in dit project onderbelicht gebleven. Ik heb in dit project de vrijheid gehad het project te doen zoals ik zelf zou willen en ik heb daarin aan de plannings fase het minst aandacht gegeven. Nu ben ik van mening dat een éénpersoons project niet gepland zou hoeven worden door een zware constructie als Prince2 of dergelijke en ik heb mijn project wel gepland en mij aan mijn planning gehouden. Het was alleen niet een erg uitgebreide planning. Ik het dit project kunnen doen aangezien ik ervaring heb met meerdere projecten dus hier is geen probleem ontstaan. Wel zalik hiermee meer rekening kunnen houden in toekomstige projecten vooral als het om projecten met grotere teams gaat.

De *uitvoerings fase* is prima verlopen. Ik had al technische kennis, heb met plezier mijzelf nieuwe technische kennis eigen gemaakt en gebruikt om het de applicatie te maken zoals ik het ontworpen had. Dit netjes in de tijds slot die ik mezelf daarvoor gegeven had. Deze competentie is altijd een sterke kant van mij geweest.

9.3 Profielschets

Ik ben duidelijk een technisch persoon. Ik vind het heerlijk om met andere technische mensen of programmeurs in vak termen te kunnen praten en het doet me goed als ik een term als polymorfisme kan gebruiken zonder dat het vragende gezichten geeft. Denken over algoritmen of datastructuren voelt als een prettige uitdaging en hierover met andere kunnen denken en discussiëren houd me graag bezig. Mijn inbreng in een team zal snel komen vanuit het perspectief van de programmeur. Maar ook het denken over creatieve oplossingen en een beetje vormgeving gaat mij ondertussen goed af. Dit project was voor één persoon maar in de opleiding mediatechnologie heb ik ook andere aspecten bekeken. Ik heb met plezier de leiding van een paar projecten op mij genomen en die met succes afgerond. Al moet ik zeggen dat ik ook toen de neiging had de planning van een project uit te besteden aan mensen met meer gevoel daarvoor.
Ik ben graag bezig met programmeren en mijn kennis vergroten, vind het ook erg fijn om iemand te helpen met het vergaren van kennis of het mee te denken over een probleem. Mijn ambitie is een klein game bedrijfje te beginnen waar ik met een kleine groep met passie voor games mooie creatieve toepassingen zou maken. Dit of kruiden verbouwen in een grote groene tuin.

Afkortingen en begrippen

AirPlay
Een SDK initieel ontwikkeld om games over meerdere mobiele platformen uit te brengen, het is initieel alleen voor een selectie groep bruikbaar geweest maar de SDK is nu te gebruiker door iedereen. De AirPlay SDK is heel snel, dat moet ook wel wil je interessante games kunnen maken. Deze snelheid is ook handig bij de ClockWise applicatie voor extra gebruikers gemak. Het systeem werkt door de applicatie te compileren voor een ARM Chip. Zo sluit je alle extra logica lagen die normaal tussen de applicatie en het systeem zit uit en word de applicatie overdraagbaar tussen bijna alle systemen.

Android
Is een besturingssysteem gemaakt door Google, en is bedoelt voor mobiele apparaten. Android gebruikt een open source Linux kernel als kern van hun besturingssysteem. Hierboven op draait het een Dalvik virtual machine waar de Android applicaties op draaien zijn. De ClockWise applicatie omzeilt deze Virtuele machine (zie AirPlay)

API *Application Programming Interface*
Een API is een verzameling van methoden, objecten of functies die de programmeurs van een applicatie nieuwe mogelijkheden geven. Er zijn hier veel verschillende opties, zo zou een API een abstractie laag kunnen zijn die complexe handelingen uit handen neemt. Maar ook kan het nieuwe eigenschappen van een apparaat bloot geven.

ARM Chip
De arm chip is een processor die in een groot gedeelte van de mobiele apparaten gebruikt word.

C++
De programmeertaal C++ is de object georiënteerde opvolger van de programmeertaal C.

Class / Object
In object georiënteerde programmeer talen is het mogelijk een class te definiëren. De class is te zien als een blauwdruk. Een object is een instantie van de class die daadwerkelijk in de applicatie te gebruiken is. Van een class zijn dus meerdere objecten te maken.

Cross-platform
De term die gebruikt word om aan te geven dan een stuk code te compileren is over meerdere platformen.

Class diagram
Deze diagram laat de technische structuur zien van de applicaties, het verteld hoe de verschillende classes samen werken, welke datavelden en methoden ze hebben.

ClockWise
ClockWise is een bedrijf in Amsterdam noord die zich bezighoud met het gelijknamige uren registratie pakket.

Design pattern
Dit is een methodiek die vaker word gebruikt om bepaalde veel (of minder veel) voorkomende technische problemen op te lossen in een programma. Zie ook “Singleton”

Deployment diagram
Dit diagram laat de omgeving van een programma zien, welke externe diensten nodig zijn en gebruikt worden door de applicatie.

GUI *Graphical user interface*
Een GUI houd zich bezig met het tekenen van knopjes, vensters en andere grafische elementen die nodig zijn voor een gebruiker om te kunnen navigeren en werken met een applicatie.

HTTP *Hypertext Transfer Protocol*
HTTP is een protocol die de mogelijkheid geeft via het internet aanvragen van informatie te doen. Het is een op tekst gebaseerde protocol en daardoor niet heel efficiënt. Binaire protocollen hoeven namelijk minder data over te sturen. HTTP is door gebruik te maken van tekst wel erg simpel te implementeren op verschillende apparaten.

HTML *HyperText Markup Language*
HTML is een opmaak taal die het mogelijk maakt informatie netjes te presenteren. Het is geen programmeertaal zoals sommige mensen denken.

iOS
Het besturingssystemen die de mobiele apparaten van Apple gebruiken heet iOS. Een voordeel wat Apple heeft ten opzichte van het mobiele besturings-systeem Android is dat er maar een paar verschillende systemen gebruik maken van iOS namelijk de iPad, iPhone en iPod. Dit geeft de programmeur zekerheid in wat het apparaat wel of niet gaat ondersteunen en voor welke resolutie de applicatie geschikt moet zijn.

Network sockets
Een socket in de context van netwerk verkeer is de poort waarover de communicatie het computer systeem binnenkomt.

Opject Georienteerd
Dit is de tegenhanger van Functioneel programmeren. Het maakt het opsplitsen van problemen iets intuïtiever aangezien het van de programmeur vereist te denken in objecten. Het is ontwikkeld met onder andere de intentie de herbruikbaarheid van een stuk code te vergroten.

Plain text
Dit is tekst zonder opmaak, meestal volgens de ASCII codering. Dit geeft 7bit per letter voor het opslaan van teksten. Meestal word 8bit per letter gebruikt aangezien dat een natuurlijker nummer is voor een computer. De extra overige bit word nog wel eens gebruikt voor de Extended ASCII set, die nog meer letters, lijntjes en leestekens definieert.

Polymorfisme
Dit is een begrip uit object georiënteerd programmeren. Het verteld dat een object op verschillende manieren geïnterpreteerd kan worden en hangt nauw samen met de mogelijkheid objecten van elkaar over te erven.

Parallel/Sequentieel
De keuze een taak door een computer parallel of sequentieel te laten doen maakt veel uit. Parallelle taken worden tegelijk uitgevoerd en zijn over het algemeen sneller gedaan dan sequentiële taken die om en om uitgevoerd worden. Het nadeel van parallelle taken is dat het niet bekend is in welke volgorde ze afgerond worden. Hierdoor kunnen snel bugs of onverwacht gedrag optreden in programma's als er niet goed mee omgegaan word.

Smartphone / Feature phone
Feature phone is de term gegeven aan de wat zwakkere mobiele telefoons die meer konden dan alleen bellen. Meestal kunnen ze Java ME applicaties draaien. Een Smartphone is een complete computer met besturingssysteem waar ontwikkelaars software voor kunnen schrijven. Er is geen duidelijke grens tussen de twee termen, aangezien hij veel verschuift. De nieuwe generatie goedkope mobiele telefoons kan steeds meer en ook kunnen de duurdere telefoons steeds meer.

SSL *Secure Sockets Layer*
SSL is een encryptie laag die communicatie via het internet kan versleutelen. Onder andere gebruikt bij een HTTPS verbinding.

SDK *software development kit*
Een is een complete oplossing voor het maken van een applicatie of game. Het heeft meestal meerdere API's, voorbeelden, documentatie en soms een IDE en/of andere begeleidende toepassingen.

SOAP *Simple Object Access Protocol*
Dit is een protocol die het mogelijk maakt gestructureerde informatie te delen over het internet. Het gebruikt XML als datadrager en doet zijn communicatie meestal via HTTP.

Singleton
Een singleton is een Design pattern die het mogelijk maakt om een object maar 1 keer te laten bestaan in een programma. Bij elk nieuw object dat gemaakt word, zou het al bestaande object mee gegeven dienen te worden. Dit is handig bijvoorbeeld voor een object die een database connectie onderhoud. Niet elke keer dat data uit een database nodig is hoeft er een nieuwe connectie te worden gemaakt. Het is handiger een connectie open te laten staan en elke keer het zelfde object te gebruiken.

Use case diagram
Dit diagram laat zien wat een gebruiker met een systeem kan doen, en hoe dat afgehandeld word. De perspectief van de gebruiker is erg sterk aanwezig in dit diagram.

UML *Unified Modeling Language*
UML is een generieke standaard die verteld over meerdere diagrammen, hoe ze er uitzien en wat ze betekenen.

WSDL *Web Services Description Language*
WSDL beschrijft een webservice, het verteld hoe met de server gesproken dient te worden en wat voor antwoord terug verwacht kan worden.

Widget
Een op zichzelf staand onderdeel zoals een knop, kalender of schuifbalk in een GUI

XML *Extensible Markup Language*
XML is een formaat die het mogelijk maakt gestructureerde informatie op te slagen. Dit doet het in tekst die door de mens leesbaar is. De opmaak komt erg overeen met HTML.

Bronnen

R. Grit. Project management. Groningen: Wolters-Noordhoff
G. Laan, Aan de Slag met C++. Schoonhoven: Academic Service
L. Barfield, Design for new media. Harlow England: Pearson education limited
E. Gamma, R. Helm, R. Johnson, J. M. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Harlow England: Pearson education limited

K de vos, A Borst. ClockWise website. 21-05-2011,
http://www.clockwise.info
Anonymus. J2ME Polish website, 21-05-2011,
http://www.j2mepolish.org
Anonymus. Micro Emu website, 21-05-2011,
http://www.microemu.org
M.Fowler. Is Design Dead?, 21-05-2011,
http://martinfowler.com/articles/designDead.html
Anonymus. Multiple phone web-based application framework 21-05-2011,
http://en.wikipedia.org/wiki/Multiple_phone_web-based_application_framework
Anonymus. Mobile application development, 21-05-2011,
http://en.wikipedia.org/wiki/Mobile_application_development
Anonymus. HTTP Compression, 21-05-2011,
http://www.websiteoptimization.com/speed/tweak/compress/
R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee.
Hypertext Transfer Protocol – HTTP/1.1, 21-05-2011
http://www.w3.org/Protocols/rfc2616/rfc2616.html
Anonymus. Mobiele besturingssystemen, 21-05-2011
http://www.portablegear.nl/achtergrond/11416/mobiele-besturingssystemen.htm
A. Avram. Developer Perception on Mobile Platforms Survey Results, 21-05-2011
http://www.infoq.com/news/2010/07/Mobile-Survey
Anonymus. U.S. Smartphone Battle Heats Up: Which is the “Most Desired” Operating System?, 21-05-2011
http://blog.nielsen.com/nielsenwire/online_mobile/us-smartphone-battle-heats-up/
Anonymus. U.S. Smartphone Market: Who’s the Most Wanted?, 21-05-2011
http://blog.nielsen.com/nielsenwire/online_mobile/u-s-smartphone-market-whos-the-most-wanted/
A. Vintu. Comparison of Android vs iPhone vs Nokia vs BlackBerry vs Windows Mobile 7, 21-05-2011
http://www.codeproject.com/Articles/73089/Comparison-of-Android-vs-iPhone-vs-Nokia-vs-BlackB.aspx

Bijlage A, afstudeer opdracht

Generieke ClockWise Applicatie voor Android, Symbian, Blackberry en Windows

ClockWise is een bedrijf in Amsterdam met 5 medewerkers (en 2 stagiaires). De core-business is een urenregistratie software oplossing die op dit moment door ongeveer 350 klanten in Nederland gebruikt wordt. De urenregistratie tool wordt aangeboden via het internet, maar kan ook via een desktop applicatie en iPhone App bedient worden middels gebruik van (bestaande) webservices. Binnen dit project gaat er gekeken worden naar het ontwikkelen van een App voor de diverse bestaande mobile devices naast iPhone.

- Onderzoek naar een techniek om een applicatie vergelijkbaar met de bestaande iPhone App voor alle of een groot deel van de andere mobile devices te ontwikkelen
- Onderzoek naar de exploitatiemogelijkheden (store en andere deployment methodes) van deze mobiele platformen
- Er gaat gebruik worden gemaakt van een bestaande XML Soap webservice.
- Zoeken naar nieuwe 'emerging properties' (nieuwe eigenschappen) voor een urenregistratie App op deze devices. Denk hierbij aan de mogelijkheden van het registreren van uren op basis van geolocatie.
- Implementatie van de gevonden oplossing voor 1 of meer van bovengenoemde mobile devices.
- Gebruikerstesten bij daadwerkelijke klanten met het prototype of de ontwikkelde App

Oplevering:

- source code
- documentatie
- uitleg/handleiding over exploitatie (op welke manier het aanbieden aan de eventuele store of deployment van de software gedaan kan worden)
- presentatie van de opdracht
- ondersteuning bij eventuele testen/vragen van klanten

Vaardigheden van de student:

- programmeren in C++, Java
- kennis over performance optimalisatie voor internet / cloud programming
- ervaring met interfaceontwerp
- ervaring met XML en webservices

Stagebegeleider:
Karel de Vos
karel@ClockWise.info

Bijlage B, MoSCoW

Groene tekst zijn punten die op het moment van het schrijven van dit document afgerond zijn.
Zwarte tekst moet nog aandacht krijgen.

Must have this - deze eis *moet* in het eindresultaat terugkomen, zonder deze eis is het product niet bruikbaar;

- Onderzoek naar systeem/techniek/SDK/API die het mogelijk maakt om 1 programma over meerdere verschillende mobiele systemen te laten draaien.
- Connectie van applicatie met bestaande SOAP server
- Implementatie van applicatie over meerdere apparaten
- Mogelijkheid uren te registreren met de applicatie
- Duidelijke weergave van klanten en projecten
- Mogelijkheden om meer dan uren te verwerken (Eenheden, reiskosten en dergelijke)
- Ondersteuning voor grote databases
- Pagina om verschillende instellingen te wijzigen

Should have this if at all possible - deze eis is zeer gewenst, maar zonder is het product wel bruikbaar;

- Onderzoek de exploitatie mogelijkheden
- Onderzoek naar gebruik van mobiele apparaten (Waar moeten we on op focussen?)
- Gebruikerstesten doen met de gemaakte software
- Mobiel inleveren van weken
- Snelle en gebruikers vriendelijke interface, rekening houdt met limitaties van mobiele apparaten
- Methoden om snel te kunnen navigeren in grote databases
- Goede afhandeling saneer het internet weg valt
- Gedocumenteerde Broncode

Could have this if it does not affect anything else - deze eis mag alleen aan bod komen als er tijd genoeg is;

- Zoeken naar eigenschappen van smartphones die het uren schrijven makkelijker en eenvoudiger kunnen maken. Of naar nieuwe technologieën zoeken die een positief effect kunnen hebben in het ClockWise systeem, en deze dan te implementeren.
- Ondersteuning van verschillende talen
- Non-blocking Informatie pop-ups om de nieuwe gebruiker te helpen met de applicatie
- Inpakken van de te versturen gegevens
- Encryptie van de te versturen gegevens
- Bewegend, ClockWise logo om activiteit aan te duiden.
- Status veld om de gebruiker op de hoogte te houden van wat er gebeurd
- Kalender om ook makkelijk op alternatieve dagen uren te schrijven
- Informatie pagina die server site van updates te voorzien is

Won't have this but would like to have this in the future

- Meerdere instelbare themes
- Foto's maken van bonnetjes en uploaden naar ClockWise voor het boekhoudings module
- Caching van informatie om navigatie te versnellen

Bijlage C, Onderzochte SDK's

	Native Widgets	Voor welke Apparaten	Native App	Commentaar	Bruikbaar
Native Applications	ja	Alles	ja	Lange ontwikkel tijd	nee
Java ME	nee	RIM, Symbian, Windows	nee	Geen goed GUI, meeste zijn slecht ondersteund	nee
Java met abstractielaag	ja	Android, RIM, Symbian, Windows	ja	Maakt een erg complexe situatie	Mogelijk
Mobile website	Lijkt Native	Alles	nee	Geen applicatie maar site	nee
Website in webkit	Lijkt Native	Alles	nee	Phonegap is een betere oplossing	nee
PhoneGap	Lijkt Native	Alles	nee	Traag, werkt slecht met SOAP, server dient aangepast te worden.	Mogelijk
Rhodes	ja	Alles	ja	Complexe SDK, ik krijg het niet werkend	Mogelijk
Appcelerator	ja	iOS, Android	ja	Alleen Android	nee
MoSync	nee	Alles	ja	Geen goed GUI	Mogelijk
WorkLight	ja	iOS, RIM, Android	ja	Te weinig apparaten, duur, gebruik externe server	nee
QuickConnect Family	ja	iOS, RIM, Palm, Android	nee	Weinig ondersteuning voor apparaten, slechte documentatie, zelfde snelheid als phonegap	nee
MobileReflex	ja	Android, iOS, RIM, Windows, Symbian,	ja	Afhankelijk van hun server, minder controle over applicatie.	nee
iPFaces	ja	iOS, RIM, Java ME	ja	Geen Android, Heel duur	nee
moppr	ja	Symbian, iOS	ja	Niet genoeg apparaten	nee
Jmango	nee	Alles	ja	Web-based, alleen maar forms, geen controle over applicatie	nee
Mobl	nee	Android, iOS, Palm	ja	Nieuwe ongeteste taal, te weinig apparaten	nee
MotherApp	ja	Android, RIM, iOS	nee	Niet genoeg mogelijk, te weinig apparaten	nee
Tersus	ja	iOS	ja	Niet genoeg mogelijk, te weinig apparaten	nee
Airplay SDK	Native Eigen gui	Alles	ja	Compilatie naar native	ja
metismo	ja	Alles	ja	Gesloten systeem	Misschien
Dragon RAD	ja	Alles	ja	4900$ jaar, RAD, weinig controle over werking van applicatie	nee
Smartface	ja	Symbian, Java ME, RIM	ja	1490$ per app, RAD, weinig controle over werking van applicatie	nee

Bijlage D, Enquête

Dank u voor het deelenemen aan de enquête.

Uit de vorige enquête is gebleken dat er een goeiende behoefte is aan de ontwikkeling van mobiele applicaties, zoals onze iPhone app. Om dit mogelijk te maken, willen we graag weten welk(e) type(n) telefoon het meest door onze klanten gebruikt wordt.

Naam bedrijf

Naam contactpersoon

Hoe vaak wordt de Nokia Smartphone (E-serie) in uw bedrijf gebruikt?

- niet
- een enkele keer
- redelijk vaak
- vaak
- uitsluitend

Hoe vaak wordt de BlackBerry in uw bedrijf gebruikt?

- niet
- een enkele keer
- redelijk vaak
- vaak
- uitsluitend

Hoe vaak wordt de iPhone in uw bedrijf gebruikt?

- niet
- een enkele keer
- redelijk vaak
- vaak
- uitsluitend

Hoe vaak worden mobiele telefoons gebaseerd op het Android platform gebruikt?

- niet
- een enkele keer
- redelijk vaak
- vaak
- uitsluitend

Hoe vaak wordt Windows Mobile in uw bedrijf gebruikt?

- niet
- een enkele keer
- redelijk vaak
- vaak
- uitsluitend

Een ander type telefoon, namelijk:

Algemene opmerking

Volgende »

Bijlage E, UML – Class diagram