Afstudeerverslag

Naam: Sander Benschop
Studentnummer: 08007381
Opleidingsinstituut: De Haagse Hogeschool
Opleiding: HBO Informatica

Opdracht: Mees
Bedrijf: 42 BV
Bedrijfsmentor: Dhr. H. Ratsma

Afstudeerperiode: 06/02/2012 – 01/06/2012
Examinatoren: Dhr. A. A. Nederend
Mevr. A. A. A. M. Jacobs

Referaat

Benschop A, Afstudeerverslag – “Mees”, Zoetermeer, 42 BV, Juni 2012.

Dit afstudeerverslag is geschreven in het kader van de afstudeeropdracht voor de opleiding Informatica aan de Haagse Hogeschool te Zoetermeer. Het beschrijft de werkzaamheden die in de periode februari tot en met juni 2012 zijn uitgevoerd voor 42 BV om de opdracht “Mees” te realiseren.

Mees is een applicatie die gebruik maakt van bestaande gegevens uit een urenregistratiedatabase van 42 en deze gebruikt om onder andere de beschikbaarheid voor medewerkers aan projecten zichtbaar te maken, met als doel het verbeteren van de doorloop van de inzet van medewerkers van 42 BV bij haar klanten.

Descriptoren

- Mees
- Postduif
- Medewerkersbeschikbaarheid
- Scrum
- Spring Security
- Crowd

Voorwoord

Mijn naam is Sander Benschop en ik studeer aan de Haagse Hogeschool te Zoetermeer. Momenteel ben ik bezig met het afronden van mijn HBO-opleiding Informatica door een afstudeeropdracht te doen bij 42 BV. Dit verslag is bedoeld om de examinatoren Arno Nederend, Nanny Jacobs en de op dit moment onbekende gecommitteerde inzicht te verlenen naar de activiteiten die ik de afgelopen periode heb doorlopen en welke keuzes ik hierbij heb gemaakt.

Graag wil ik enkele mensen bedanken die me geholpen hebben bij het doorlopen van mijn afstudeertraject.

Vanuit de opleiding is dit Arno Nederend, met wie ik gedurende mijn afstudeerproject de voortgang en complicaties van het project besproken heb. Verder wil ik Nanny Jacobs bedanken voor de input die ze heeft gegeven tijdens de definitie van mijn afstudeeropdracht. Beiden examinatoren wil ik bedanken voor de feedback op mijn afstudeerverslag.

Vanuit 42 BV wil ik allereerst Harko Ratsma bedanken, die met zeer veel enthousiasme zijn rollen als project lead en lid van het product owner team heeft vertegenwoordigd. Verder wil ik ook andere collega's zoals Robert Bor, Oscar Westra van Holte-Kind en Eric Meijer bedanken die een rol hebben gespeeld om dit project tot een succes te brengen.

Tenslotte wil ik graag mijn studiegenoten Jesse van Assen en Ferdy Perdaan bedanken voor de feedback die zij geleverd hebben op delen van het afstudeerverslag. Deze kritische blikken hebben het verslag tot een beter niveau helpen te brengen.

Inhoudsopgave

1 Inleiding

Gedurende de periode van februari tot en met mei 2012 ben ik werkzaam geweest bij 42 BV. Hierbij heb ik aan de opdracht: “Mees” gewerkt, wat onder andere de beschikbaarheid van medewerkers voor projecten in kaart brengt, gebaseerd op gegevens uit een bestaande database.

Het doel van dit verslag is het verduidelijken van de werkwijze en leerprocessen gedurende de afstudeerperiode. De afstudeercoördinatoren kunnen hierdoor een beeld vormen over de omvang en diepgang van het uitgevoerde werk.

Dit verslag is opgebouwd uit de volgende onderdelen:

- **Bedrijf**

  In dit hoofdstuk geef ik een beschrijving van het bedrijf waar ik tijdens mijn afstudeerperiode heb uitgevoerd en vertel ik het één ander over de geschiedenis ervan en het klantprofiel.

- **Opdrachtomschrijving**

  In dit hoofdstuk geef ik een korte samenvatting van mijn afstudeeropdracht en van de situatie voordat ik hieraan begon.

- **Totstandkoming Plan van Aanpak**

  In dit hoofdstuk vat ik mijn Plan van Aanpak samen waarin ik onder andere mijn planning en ingeschatte risico’s kort laat terugkomen.

- **Kern**

  In dit onderdeel van het verslag wordt de rode draad van het verslag duidelijk gemaakt: het beschrijft de uitgevoerde activiteiten, de werkwijze hierbij en de problemen die deze met zich meenamen.

  Door middel van deze toelichting wordt duidelijk wat de rode draad van de activiteiten tijdens mijn stage was.

- **Evaluatie**

  In dit hoofdstuk geef ik een oordeel over de werkwijze die ik tijdens het project heb toegepast. Dit hoofdstuk valt onder te verdelen in een procesbeschrijving waarin ik mijn aanpak evalueer, een productbeschrijving waarin ik de producten die ik heb opgeleverd evalueer en een competentieverantwoording waarin ik beschrijf hoe ik de vooraf gedefinieerde competenties heb behaald zijn.

- **Begrippenlijst**

  Lijst waarin de complexere begrippen uit het verslag duidelijk worden gemaakt. Een kortere uitleg van deze begrippen kan bij het eerste gebruik ervan in een voetnoot worden gevonden.

- **Bijlagen**

  In dit hoofdstuk bevinden zich enkele bijlagen die ik tijdens mijn project heb gemaakt die relevant zijn voor in dit verslag.

2 Bedrijf

In dit hoofdstuk vertel ik kort iets over de geschiedenis van 42 BV, wat het bedrijfsprofiel is en hoe het klantprofiel van het bedrijf eruit ziet.

Geschiedenis

De oprichter van 42 BV, Eric Meier, was tot 2003 werkzaam als directeur van enkele Nederlandse bedrijfstakken van de multinational Software AG, waarvan het hoofdkantoor in Duitsland was gevestigd. Wegens een afnemende winst in het buitenland werd het Nederlandse kantoor gesloten, ondanks het feit dat deze nog wel winstgevend was. Eric Meier besloot de ruimte in de markt die hierdoor vrij zou komen te benutten voor een eigen bedrijf.

In 2005 is 42 BV een samenwerkingsverband aangegaan met het Amsterdamse Kensas. De twee bedrijven zijn ondergebracht onder een holding genaamd M.A.D. Dit staat voor Mice and Dolphins, de twee meest intelligente diersoorten op aarde volgens the Hitchhiker's Guide To the Galaxy, het boek waar eveneens de bedrijfsnaam 42 afkomstig van is. Tegenwoordig wordt de naam Kensas niet meer gebruikt, de naam 42 is hiervoor in de plaats gekomen.

42 BV is inmiddels uitgegroeid tot een bedrijf met 40 medewerkers. Enkelen hiervan werken op kantoor in Zoetermeer of Amsterdam, maar de meeste werken intern bij de klant.

Bedrijfsprofiel

De missie van 42 BV is het ontwikkelen van hoogwaardige software waarmee haar klanten hun werkprocessen en bedrijfsresultaten kunnen verbeteren. 42 BV ontwikkelt maatwerk Java-enterprise producten, en doet dit bij voorkeur open source. Verder helpt 42 BV grote klanten bij het integreren van Atlassian-producten. 42 BV gebruikt deze software zelf ook en is tevens partner.

Binnen 42 hangt een informele, vriendelijke werksfeer. Van medewerkers wordt een hoge mate van zelfredzaamheid verwacht, wat ze realiseren middels scholing in diverse delen van het vakgebied die net iets buiten de "comfort-zone" van de Java-ontwikkelaars liggen.

Klantprofiel

42 BV heeft na de oprichting enkele grote klanten van Software AG kunnen meenemen, waaronder Schiphol en CenterParcs. Hierdoor kon 42 BV meteen beginnen met grotere opdrachten, iets wat lastig kan zijn voor beginnende bedrijven. Enkele andere (recente) grote klanten van 42 BV zijn Albert Heijn, de Consumentenbond, de Belastingdienst, ANWB, Essent en Nidera.

3 Opdrachtomschrijving

42 BV heeft op dit moment al diverse interne applicaties die vorig jaar verzameld zijn onder de paraplu-applicatie “Volière”. Een onderdeel hiervan is bijvoorbeeld het registratiesysteem “Mus”. Hierin worden de gewerkte en geplande uren geregistreerd en vervolgens opgeslagen in een database. De gegevens uit de Mus-database wil 42 BV gaan gebruiken om een tool te ontwikkelen die de managers en de salesafdeling van informatie kunnen voorzien om betere beslissingen te maken wat betreft inzet van medewerkers bij projecten.

De salesafdeling heeft op dit moment problemen met het efficiënt inzetten van medewerkers op projecten, wat in het verleden geresulteerd heeft in het feit dat medewerkers “op de bank” terecht kwamen. Voor het management is het daarentegen lastig om de medewerkers binnen lopende projecten efficiënt in te zetten. Reden hiervoor is dat in beide situaties er een gebrek is aan gecentraliseerde, duidelijke, actuele en historische data.

De bedoeling is dus dat Mees hier verandering in gaat brengen door op een visuele manier data te presenteren die de salesafdeling en managers kunnen gebruiken om betere beslissingen te nemen.

Tijdens mijn afstuderen voer ik globaal de volgende activiteiten uit. Deze activiteiten komen terug als hoofdstukken in de kern van het verslag.

Inlezen technieken, oriënteren op bestaande architectuur

Voordat er begonnen wordt met het modelleren van de te creëren webapplicatie[1] ga ik me eerst oriënteren op de bestaande architectuur van de Volière. Hierbij wordt ook gekeken naar de Mus-database. Het doel van deze activiteit is het vergaren van voldoende basiskennis van het systeem en de database om het opstellen van de requirements en het beginnen van de ontwikkeling te vergemakkelijken.

Proof of concepts ontwikkelen

Voordat er begonnen wordt aan het verzamelen van de requirements worden enkele ideeën die de product owner had getoetst, om zo te kijken of deze haalbaar zijn. De uitkomsten van deze PoC’s kunnen invloed hebben op de uiteindelijke requirements, omdat er in het geval van onhaalbaarheid andere ideeën bedacht moeten worden die wellicht andere eisen met zich meebrengen.

[1] Applicatie die vanaf het internet benaderd en gebruikt kan worden

Requirements opstellen en initiële ontwerpen maken

Na afronding van het plan van aanpak wordt begonnen aan het inventariseren van de wensen van de (hoofd)gebruikers. Deze informatie wordt vervolgens gebruikt om requirements mee op te stellen die input geven voor het verdere project.

Voordat het mogelijk is om te beginnen aan het programmeren van de webapplicatie of de Java backend[1] wordt eerst de intiële opzet van de UML-diagrammen gemaakt. Deze worden als hulpmiddel gebruikt tijdens de verdere ontwikkelingen. Een combinatie van diagrammen geeft een voldoende beeld over de wensen aan- en structuur van- de webapplicatie en de Java backend om deze te kunnen ontwikkelen. In het geval van wijzigingen in de structuur gedurende het programmeren worden de UML-diagrammen hier op aangepast.

Incrementeel ontwerpen en ontwikkelen backend en frontend[2]

Na afronding van de initiële ontwerpfase wordt begonnen met het ontwikkelen. Allereerst wordt een opzet voor de backend gemaakt zodat het mogelijk is er vanuit de frontend tegenaan te spreken. Vervolgens wordt invulling aan de webinterface gegeven en de daadwerkelijke functionaliteit van de backend geïmplementeerd. Op deze manier kan er al tegen een controller van de backend worden gesproken vanuit de frontend, waarna later de daadwerkelijke data wordt teruggegeven. Dit maakt het makkelijker om te testen of een onderdeel werkt, omdat er met gemockte data kan worden gewerkt.

De kwaliteitseisen van 42 BV staan ook in dit project centraal: het schrijven van code van lage complexiteit, die vrij is van conventieschendingen en grondig is getest. De methode voor het testen van de JavaScript/jQuery code en het inzichtelijk maken van de kwaliteitscriteria wordt tijdens de stage ook onderzocht.

Kwaliteitsbewaring

Gedurende het gehele project moet de kwaliteit van de geschreven code bewaard blijven. Dit doe ik onder andere door middel van het schrijven van unit tests en het bijhouden van metrische gegevens in Sonar. In hoofdstuk 9: *"Kwaliteitsbewaring"* ga ik uitgebreider in op het proces dat hierbij hoort.

[1] Onzichtbare deel applicatie, handelt frontend requests af
[2] Zichtbare deel applicatie, stuurt onder andere requests naar de backend

4 Plan van Aanpak

In de eerste week van mijn project ben ik begonnen met het opstellen van een Plan van Aanpak. Hiervoor ben ik onder andere gaan praten met een collega die een eerdere opzet heeft gemaakt voor een medewerkersbeschikbaarheidssysteem, zoals beschreven wordt in hoofdstuk 5: "*Inlezen technieken, oriënteren op bestaande architectuur*" om informatie over zijn methoden op te doen.

In de eerste dagen van het project heb ik in kaart gebracht wie de verschillende stakeholders zijn waarmee ik te maken heb gekregen tijdens het uitvoeren van mijn opdracht, wat de SCRUM-processen zijn die ik tijdens het doorlopen van mijn afstudeeropdracht ga volgen en wie welke rol in dat proces gaat vervullen.

Medewerker	***Scrum-rol***	***Reguliere functie***
Harko Ratsma	Project lead, Product owner	Project Manager
Eric Meijer	Product Owner, Boekhouder vertegenwoordiger	CEO
Robert Bor	Technical lead	Chief Technology Officer
Ron Oudshoorn	Sales vertegenwoordiger	Manager Marketing en Sales
Sander Benschop	Scrum Master, Developer	Software Developer

In deze periode heb ik me ook georiënteerd op de ontwikkelmethoden en ontwikkelomgeving die ik tijdens het project ga gebruiken.

Afgesproken werd om dagelijks een stand-up meeting te houden waarin werd besproken wat ik de dag ervoor had gedaan, tegen wat voor problemen ik was aangelopen en wat ik die dag wilde doen. Verder was besloten om sprints van één week te houden, en na iedere sprint een retrospective[1] te houden waarin besproken wordt hoe de sprint is verlopen.

De reden hiervoor was het feit dat op deze manier het mogelijk werd om eventuele verschillen in interpretatie tussen de product owner en mij zo vroeg mogelijk uit de weg te kunnen helpen. Ook motiveert een sprint van slechts 1 week om user story's in zo klein mogelijke delen op te splitsen. Dit helpt vervolgens weer bij het inplannen van (sub)taken, omdat het vanwege de atomaire indeling duidelijker is wat er bij een bepaalde taak komt kijken.

In overleg met de Chief Technology Officer heb ik een selectie gemaakt van enkele van de tools die momenteel binnen 42 BV gebruikt worden. Er werd besloten dat Jira gebruikt ging worden voor project tracking en issue management. Hierover is later in dit hoofstuk meer te lezen. Zowel Bamboo als Continuous integration server en Sonar worden gebruikt als kwaliteitsbewaringstool. Hierover is meer te lezen in hoofdstuk 9: "*Kwaliteitsbewaring*".

[1] Meeting na afloop van een sprint waarin de afgelopen Sprint wordt geëvalueerd.

4.1 Opstelling planning

De planning uit het Plan van Aanpak bestaat uit 5 delen: het inlezen op de bestaande architectuur, het ontwikkelen van Proof-of-Concepts voor een Fisheye view[1], het samenstellen van de requirements en maken van initiële ontwerpen, het instrumenteel ontwikkelen van de applicatie en het schrijven van het afstudeerverslag.

De planning is een manier ingedeeld dat alles voorafgaand aan de incrementele ontwikkeling van de applicatie als voorbereiding gezien zou kunnen worden en de afronding van het afstudeerverslag als slot. Voor de kern is doelbewust een grovere granulariteit gekozen: deze valt uiteen in vele kleinere user story's, maar het van tevoren inplannen van deze user story's strookt niet met de Scrum-gedachte. Hierbij is het gebruikelijk om per Sprint een planning te maken en vooruitschrijdende inzichten mee te nemen. Het maken van een specifiekere planning is daarom dus niet wenselijk.

4.2 Bepalen risicofactoren

Bij het opstellen van mijn Plan van Aanpak heb ik de relevante risicofactoren van het project in kaart gebracht. Een van deze factoren was de waarschuwing van mijn eerste coördinator, Arno Nederend, om af te studeren bij een bedrijf waar ik al werkzaam was, omdat de kans bestaat dat er in dat geval werk dat buiten de afstudeeropdracht valt wordt uitgevoerd.

De andere twee, een overhead aan op te leveren documentatie en het maken van verkeerde ontwerpkeuzes door onvoldoende architectuurkennis heb ik geïdentificeerd aan de hand van respectievelijk het afstudeerplan en persoonlijke ervaring.

In het afstudeerplan staat beschreven dat ik verwacht diverse UML-diagrammen te ontwikkelen wat uiteraard erg nuttig is voor mijzelf tijdens de ontwikkeling en anderen bij het onderhouden van de applicatie, maar ook veel tijd in beslag neemt. Indien het ontwerpen van UML-diagrammen beperkt wordt tot de modellen die verduidelijking biedt kost dit minder tijd dan van overal modellen van maken en biedt dit voor de organisatie nog steeds de juiste hoeveelheid informatie.

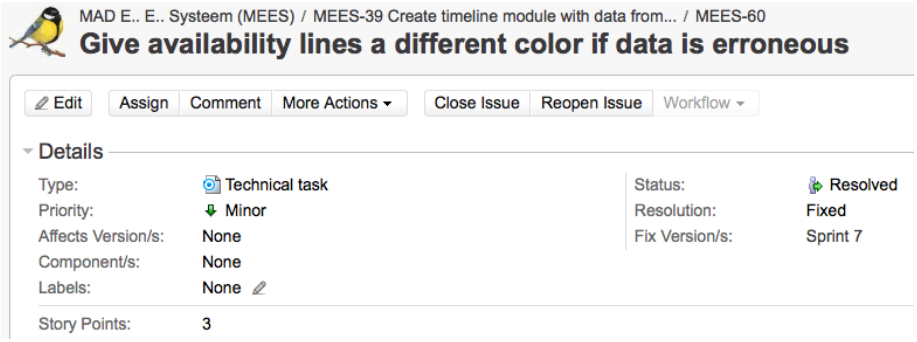
Het maken van verkeerde ontwerpbeslissingen door onvoldoende kennis van de architectuur is iets wat ik tijdens mijn werk ooit heb meegemaakt. Vergissingen worden naarmate er meer tijd verstrijkt steeds duurder om op te lossen. Om die reden vind ik het belangrijk om voldoende basiskennis op te bouwen over de architectuur voor de ontwikkeling begint.

[1] View waarin op bepaalde items en de omliggende items die hierbij horen kan worden ingezoomd, wat een bepaald vissenkomperspectief creëert

4.3 Projectmanagement met Scrum in Jira

Binnen dit project heb ik gebruik gemaakt van Scrum als projectmanagement-methodiek. Ik heb hierbij gebruik gemaakt van het projectmanagementtool Jira van het bedrijf Atlassian.

Hierin kunnen user story's worden gedefinieerd en versies worden bijgehouden.



De verschillende sprints heb ik als versies gedefinieerd, zodat er steeds gezien kon worden wat de voortgang van een bepaalde sprint. Ook werden hierbij burn-down charts gegenereerd.

Na het inplannen van een Sprint werden de ingeplande user story's in Jira toegevoegd aan de juiste versie. Voor ieder punt wordt een aantal story points bijgehouden die tijdens de Sprint-planning zijn bepaald.

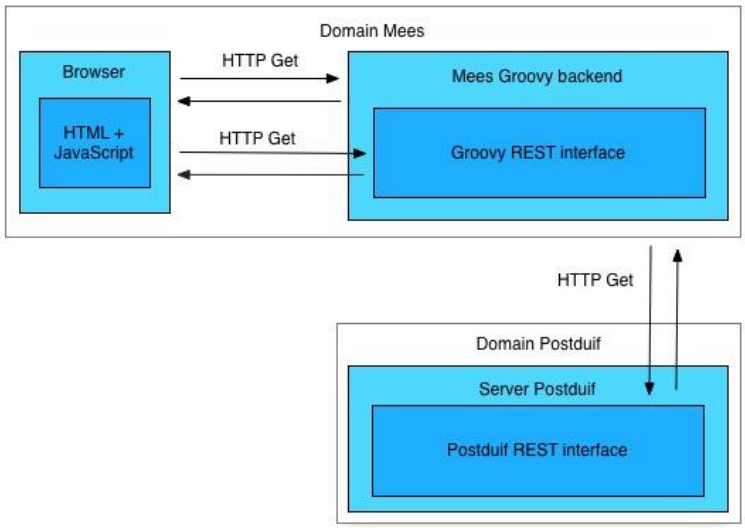
Het voordeel van een aanpak met behulp van een tool als deze boven het gebruik van bijvoorbeeld geeltjes op een bord, zoals ook vaak gebeurd, is het feit dat deze ook buiten het kantoor ingezien kunnen worden. De product owner was niet dagelijks op kantoor aanwezig, maar kon op deze manier wel inzicht krijgen in het verloop van het project gedurende de week. Indien ik ergens tegenaan liep bij het uitvoeren van een user story zette ik dit ook in het commentaar ervan.

Op deze manier hoefde ik bij de retrospective minder uit te leggen waardoor het minder tijd in beslag nam, en gaf het me tevens de mogelijkheid om later terug te kijken naar een probleem en de oplossing ervan indien ik tegen iets soortgelijks aanliep.

5 Inlezen technieken, oriënteren op bestaande architectuur

Om duidelijk te krijgen waar ik exact mee te maken had, besloot ik te beginnen met het oriënteren op de bestaande architectuur waarin Mees gaat worden ondergebracht. Hierdoor wilde ik een beter beeld krijgen van de bestaande architectuur. Omdat uitbreiding van Postduif ook noodzakelijk was om mijn opdracht tot een succes te brengen wilde ik weten hoe de communicatie en globale werking er ongeveer uit zag, zodat ik wist welke delen ik kon aanpassen zonder problemen in andere systemen te veroorzaken en welke delen wat meer vastgelegd waren.

Een collega heeft ooit een opzet gemaakt voor "Mees", of in ieder geval het idee dat ze daar toen bij hadden. Het betrof een standaard Gantt-chart[1] gevuld met de beschikbaarheid van medewerkers die getekend werd door een JavaScript library. Het gaf geen verdere informatie die de stakeholders konden gebruiken om beslissingen mee te nemen.



Verder betrof het een Grails[2] web-applicatie die ook niet erg beviel gezien het feit dat de informatiestroom al via de Java-backend, Postduif, verliep. Een extra laag hier tussenin bouwen maakte de voortgang van ontwikkeling trager, omdat er rekening moest worden gehouden met diverse mismatches die ontstaan tussen Grails en de Postduif Spring MVC web service. Dit is iets wat niet wenselijk zou zijn bij de ontwikkeling van de nieuwe Mees. Om deze reden was de wens van 42 BV dat de frontend van Mees enkel in HTML en JavaScript geschreven werd.

Ik heb een meeting gehouden met de ontwikkelaar van de Grails webapplicatie en mijn bedrijfsmentor om te kijken in hoeverre delen hiervan herbruikbaar waren. Hierin heb ik vragen gesteld over de werking van de Grails-applicatie die hij had geschreven. Ook was ik geïnteresseerd in de werking van de JavaScript library geschreven door een extern bedrijf die op dat moment gebruikt werd. Deze is door de ontwikkelaars ervan bewust tot een grote onleesbare codebrei verwerkt zodat deze onleesbaar wordt voor mensen.

---

[1] Grafieksoort die gebruikt wordt om planningen mee te maken
[2] "Groovy on Grails": een webframework voor de Groovy-taal

Ook heb ik gevraagd of hij toelichting kon geven op de vermeende mismatches tussen Grails en Spring MVC applicaties.

De JavaScript library was in staat om een Gantt-chart te presenteren, maar niet meer dan dat. Het uitbreiden tot een fisheye-view zoals de stakeholders wensten was dan ook niet mogelijk geweest. De onuitbreidbaarheid van de bestaande situatie gecombineerd met het feit dat de Grails-laag eigenlijk niet als wenselijk werd gezien heeft me overtuigd dat het herschrijven van Mees in enkel HTML en JavaScript de juiste route was om te bewandelen. Er zou geen gebruik meer worden gemaakt van de bestaande library. Indien aan die eisen werd voldaan was immers alle ruimte aanwezig om de code naar eigen wens aan te passen.

Vervolgens heb ik gekeken naar de module waar de nieuwe Mees-applicatie tegenaan gaat praten, Postduif. Ik heb het project opgestart en gekeken hoe de demopagina, die gebruikt kon worden om calls uit te voeren en de teruggave weer te geven, in elkaar zat. Op deze manier wist ik hoe ik straks de werking van de door mij geschreven controller[1] visueel kon inzien. De demo-pagina voert op eenzelfde manier calls naar Postduif uit als Mees gaat doen, middels Ajax[2]-calls. De werking van deze pagina's testen is dus een goede eerste stap om te valideren dat de calls vanuit Mees eveneens gaan functioneren.

Ook heb ik me georiënteerd op de bestaande programmacode van Postduif, in het bijzonder op de packagestructuur, de domeinklassen die binding hebben met medewerkersbeschikbaarheid en de algehele codeconventie. Ik heb dit gedaan zodat ik de code die ik aan Postduif ga schrijven zo kan opstellen dat deze in het grotere geheel past. Daarna heb ik ook even gekeken naar een ander project wat nu al tegen Postduif praat om te kijken hoe dit precies in zijn werk gaat, zodat ik een idee krijg hoe ik dat hierna moet doen. Postduif moet uitgebreid worden om de mogelijkheid te scheppen om Mees er mee te laten werken.

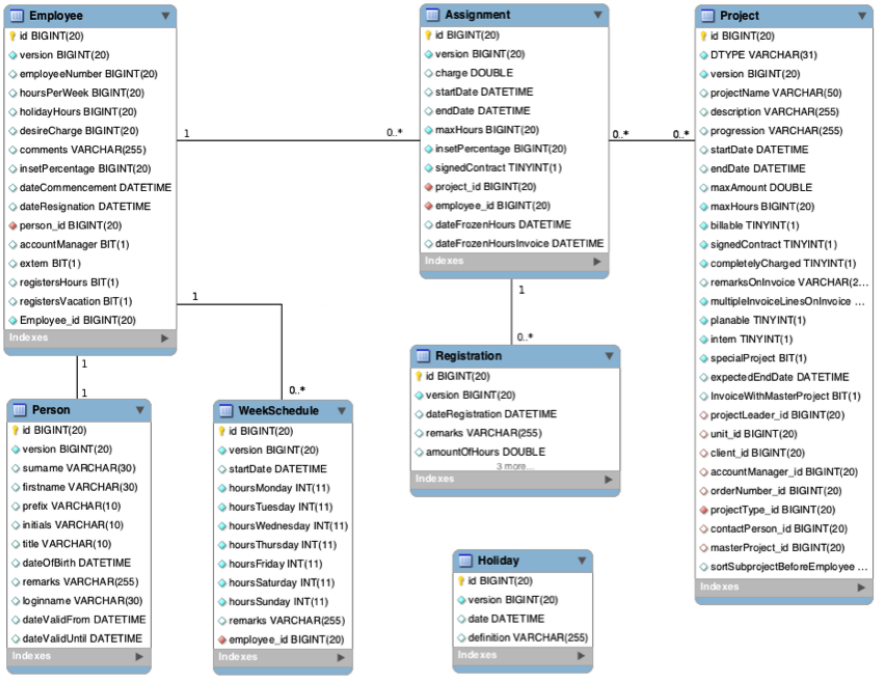
Gelijktijdig met het oriënteren op deze code heb ik me ingelezen in programmeertalen en technieken als JavaScript, jQuery, HTML5, CSS3 en Spring MVC. Dit heb ik onder andere gedaan door tutorials uit te voeren en me in te lezen in het onderwerp met behulp van diverse boeken. Dit heb ik gedaan om een solide kennisbasis te creëren die ik in de rest van de opdracht kon gebruiken om het systeem te realiseren.

Binnen de Volière is nog een andere applicatie aanwezig, genaamd Snip. Dit is een applicatie van waaruit onder andere facturen kunnen worden samengesteld en verzonden naar klanten. Het beschikt over een eigen backend geschreven in Groovy, een dynamische programmeertaal[3] voor de Java Virtual Machine.
Deze spreekt vervolgens weer met de Postduif-backend, waar de eigenlijke functionaliteiten in zijn vastgelegd.

---

[1] In deze context: klasse waar REST-calls tegenaan uitgevoerd kunnen worden en deze requests doorgeven aan een klasse op de servicelaag
[2] "Asynchronous JavaScript en XML": biedt mogelijkheden om nadat een pagina geladen is calls uit te voeren naar de server
[3] Taal waarin veel gedrag runtime wordt uitgevoerd in plaats van compile time

Tijdens deze fase heb ik me ook georiënteerd op de Mus-database om zo kennis te vergaren van de vorm en inhoud waarmee ik straks berekeningen moet gaan uitvoeren.



Er waren enkele zaken die mij opvielen aan de database waaronder een gebrek aan consistentie in de naamgeving. Omdat dit enigszins verwarrend kan zijn wilde ik het makkelijker voor mezelf maken om zaken over de database op te zoeken, daarom heb ik er een lokale kopie van gemaakt.

Ik heb op basis van deze database het domeinmodel hierboven gegenereerd en uitgeprint, die ik gemakkelijk kon gebruiken even terug te kijken naar de databasestructuur. Dit was met name in het begin van de ontwikkeling erg nuttig, toen ik de mapping naar de database heb gemaakt. Dit model kan in volledige grote gevonden worden onder bijlage A: “Databasemodel Mus”.

Aan het einde van deze oriëntatiefase had ik een goed beeld van welke functionaliteiten van Postduif gebruikt werden door Snip, wat deze globaal inhielden en hoe ik vanuit de nieuwe Mees tegen Postduif aan moest praten. Ik beschouwde het globale beeld dat ik toen van de architectuur had als voldoende om aan de slag te kunnen gaan met de ontwikkeling, de specifieke implementaties zou ik bekijken als ik ermee aan de slag zou gaan omdat dit teveel was om in één keer in me op te nemen terwijl ik die specifieke kennis waarschijnlijk niet volledig nodig zou hebben. Postduif handelt namelijk ook backend functionaliteit af die heel andere data gebruiken dan Mees en waar ik dus weinig mee te maken ga hebben.

6 Proof of concepts ontwikkelen

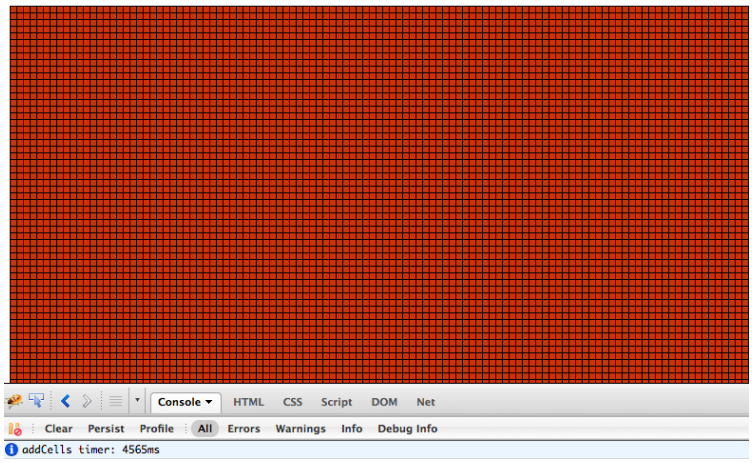
Na de fase waarin ik mij georiënteerd heb op de bestaande architectuur ben ik begonnen met het maken van enkele Proof-of-Concepts waarin ik enkele ideeën wilde toetsen.

De product owner had enkele ideeën over de manier waarop hij data gerepresenteerd wilde zien. Dit omvatte een grote hoeveelheid cellen waarin allerlei data getoond zou worden, en waar op kan worden ingezoomd door er overheen te bewegen met de cursor. Bij het inzoomen moest er extra data worden getoond.

Mijn vermoeden was dat deze grote hoeveelheden cellen zeer traag zouden functioneren, omdat er zeer veel iteraties[1] nodig zijn om alle cellen te tekenen. Ook verwachtte ik dat het met de cursor bewegen over de cellen de Fisheye-view zou kunnen vertragen, omdat er iedere keer een jQuery-selector[2] diverse cellen moet selecteren om groter in omvang te maken, zodat er meer data in gestopt kan worden. Het selecteren en aanpassen van deze cellen uit een dusdanig grote hoeveelheid kon ook aardig wat tijd in beslag nemen. Met het oog op de grote overhead heb ik aangeraden de zoomactie door middel van het bewegen van de cursor over de cel te vervangen met een klikactie hierop. Op deze manier wordt de zoomactie niet voor iedere cel waar de muis overheen beweegt uitgevoerd, wat met een hoeveelheid cellen als deze dus zeer vaak gebeurd.

De visuele representatie die de product owner aanvankelijk in gedachten had bevatte 400 kolommen en 1000 rijen. Het voordeel van deze representatie is het feit dat alle informatie die nodig kan zijn om goede sales-beslissingen te maken direct voorhanden is, het nadeel is wel dat het de performance nogal zou kunnen verlagen.

Om dit inzichtelijk te maken ben ik begonnen met het ontwikkelen van de Proof-of-concept applicatie met deze hoeveelheid cellen.



Op dit moment werd er nog geen daadwerkelijke data uit het systeem ingeladen: het ging in deze test enkel om het tekenen van de grote hoeveelheden cellen. Ik wilde namelijk eerst weten hoeveel tijd dit kostte. Het opvragen van de data zelf kost even tijd omdat er servercommunicatie plaats moet vinden, maar het plaatsen

[1] Een iteratie is een herhaling, in dit geval binnen een for-loop.
[2] Methode om HTML-elementen mee te selecteren

van de gegevens zelf in de cellen geeft niet veel overhead omdat dit in dezelfde for-loop kan gebeuren waar de cellen zelf worden getekend. Het laten renderen van de hoeveelheid cellen gaf op dit moment dan ook genoeg informatie om te kijken of een oplossing wenselijk was.

Het opbouwen en weergeven koste zoals hierboven in een screenshot wordt getoond ruim 4,5 seconde, wat enorm traag aanvoelde. Het klikken op een record selecteert de cellen met de aangrenzende cellen en voegt hier een HTML class aan toe zodat het middels CSS visueel onderscheiden kan worden van de overige cellen.

Op deze schaal duurde de totale mutatie die werd uitgevoerd na een klik ruim 1,7 seconden. Dit lijkt niet veel, maar in een systeem als deze is het wenselijk om direct feedback te krijgen als je ergens op klikt. Een gebruiker zal waarschijnlijk meerdere klikacties gebruiken om alle gewenste informatie tot zich te nemen.

Als een gebruiker bij iedere klikactie enkele seconden moet wachten raakt hij waarschijnlijk geïrriteerd en wilt hij op den duur geen gebruik meer maken van Mees. Dit moet uiteraard worden voorkomen.

Er werd besloten om het aantal cellen te verminderen door op basis van periodes te gaan werken. Het precieze aantal cellen werd per view bepaald, zoals te lezen is in hoofdstuk 7: "Requirements opstellen en initiële ontwerpen maken". Er was met het uitvoeren van deze test dus bewezen dat de hoeveelheid cellen die de product owner oorspronkelijk in gedachten had niet performant genoeg was.

Vervolgens heb ik de Proof-of-Concept zo aangepast dat deze minder cellen genereert: 90 kolommen die een periode van 3 maanden representeren (hierbij werd nog geen onderscheid gemaakt tussen werkdagen en weekenddagen) en 50 rijen, wat iets meer is dan het aantal medewerkers van 42 BV op dit moment, zodat er nog wat ruimte voor uitbreiding is. Dit verbeterde de performance aanzienlijk, het opbouwen kostte nog slechts 1 seconde, en de tijdsduur voor het selecteren van data werd met 2/3^{e} verlaagd.

Dit was echter nog steeds een redelijk lange tijd, dus ben ik aan de slag gegaan met optimalisatie. Zo heb ik heb onderzocht of het verschil maakte om een table te gebruiken voor het aanmaken van de cellen of een listitem met css stijl eroverheen, zoals ook vaak gebeurd in de web-developmentwereld. Dit verschil bleek niet significant. Wat wel sterk verschil uitmaakte was het efficiënter maken van mijn jQuery selectors. Waar ik eerst op HTML-klasse selecteerde, wat feitelijk betekent dat er over ieder element geïtereerd moest worden waarbij ieder element dat de desbetreffende klasse bevatte werd geselecteerd, heb ik hierna een andere methode toegepast.

Het toevoegen van HTML-identifiers aan alle elementen, en hier voortaan op te selecteren bleek een enorm verschil te maken. Reden hiervoor is het feit dat identifiers geïndexeerd worden, waarna jQuery hier zeer snel selecties op kan uitvoeren.

Na analyse van de diverse Proof-of-concepts ben ik tot de beslissing gekomen om tables te gebruiken in plaats van listitems, omdat deze mogelijkheden bieden tot uitrekking van cellen over meerdere kolommen/rijen. Waar mogelijk worden identifiers gebruikt voor selectie, en de hoeveelheid kolommen en rijen wordt beperkt gehouden.

7 Requirements opstellen en initiële ontwerpen maken

Om in kaart te brengen wat de wensen van de diverse stakeholders aan de applicatie zijn heb ik Requirement Meetings gehouden met de vertegenwoordiger van de afdeling Sales, Ron Oudshoorn, en CEO Eric Meijer. Waar Ron vooral een view wilt waarin medewerkersbeschikbaarheid inzichtelijk wordt gemaakt, wilt Eric graag een view om historische data in weer te geven vanuit de rol van boekhouder. Harko Ratsma vertegenwoordigt de "management-afdeling", die feitelijk een aggregatie van eerder genoemde views wilt hebben. Verder in dit document beschrijf ik de views in meer detail.

Voor de requirements die ik tijdens deze fase van het afstudeerproject heb opgesteld geldt dat deze niet per se allemaal afgerond worden tijdens het verloop van de afstudeerperiode maar dat dit een overzicht is van de totaliteit aan functionaliteiten die uiteindelijk in Mees opgenomen worden. De reden dat ik voor deze aanpak heb gekozen is het feit dat ik vanaf het begin een zo totaal mogelijk beeld wil krijgen van hoe de uiteindelijke applicatie eruit komt te zien zodat ik hier bij de ontwikkeling rekening mee kan houden.

Daarnaast geeft deze aanpak de mogelijkheid om in een latere fase een bepaalde requirement naar voren te schuiven of juist later uit te voeren. In de Scrum-methodiek wordt steeds gekeken welke user stories op dat moment de meeste waarde voor de organisatie bieden, de door mij gekozen aanpak past dus goed bij dit principe.

De requirements worden eerst kort beschreven en vervolgens in een SMART-tabel[1] weergegeven en geprioriteerd. Iedere letter van deze afkorting staat voor een bepaalde doelstelling. De betekenis van de letters, en de vragen die beantwoord moeten worden zijn als volgt:

Doelstelling	***Bijbehorende vragen***
Specifiek	- Wat wil ik bereiken ? - Waarom wil ik dit bereiken ? - Wie is erbij betrokken ? - Waar vind dit plaats ? - Welke requirements en constraints horen hierbij ?
Meetbaar	- Hoeveel ? - Hoe weet ik wanneer het is afgerond ?
Acceptabel	- Hoe kan het doel worden bereikt?
Relevant	- Is het de moeite waard? - Is dit het juiste moment? - Komt dit overeen met onze wensen/doelen? - Ben ik de juiste persoon?
Tijdgebonden	- Wanneer wordt het uitgevoerd? - Wat kan ik direct doen? - Wat kan ik over enkele tijd doen?

7.1 Sales requirements

In Mees moet rekening worden gehouden met de inzetbaarheid van medewerkers, in de tijdslijn moet dat ook duidelijk zijn. Uiteindelijk is het wenselijk dat er ook per klant gesorteerd kan worden.

Uiteindelijk zou hij graag de mogelijkheid hebben om te zien welke medewerkers binnenkort vrij komen voor inzet bij projecten. Hierin zou hij bijvoorbeeld graag willen zien waar medewerkers momenteel zitten, en per medewerkers inzicht verkrijgen in de verschillende projecten waar ze momenteel werkzaam aan zijn en tot wanneer deze projecten lopen.

Geef de beschikbare uren voor medewerkers weer	
Specifiek	- Uiteindelijk moet er een duidelijk overzicht beschikbaar zijn van de medewerkersbeschikbaarheid over een bepaald periode. - De Sales-afdeling wil deze informatie gebruiken om zo betere beslissingen te nemen over medewerkersinzet.
Meetbaar	- Ik weet dat de user story[1] is afgerond als de medewerkersbeschikbaarheid wordt weergegeven en de product owner een acceptatietest heeft uitgevoerd waarin hij de uitkomsten van het algoritme controleert.
Acceptabel	- Het doel kan worden bereikt door inzicht te verkrijgen in de benodigde gegevens uit de Mus-database. Omdat de datakwaliteit hiervan op dit moment in veel gevallen beneden peil is kan het goed dat er situaties ontstaan die metingen lastig maken. Hier wordt naar gekeken bij de acceptatietest en het algoritme kan hierop worden aangepast indien nodig.
Relevant	- Dit is de belangrijkste user story van het gehele project. - Ik ben de juiste persoon om deze user story op te pakken, maar heb hier wel de kennis over de database bij nodig van de product owner Harko Ratsma, daarom ga ik regelmatig met hem overleggen.
Tijdgebonden	- Omdat deze user story het belangrijkste is wordt deze direct worden opgepakt. - Het initiële algoritme kan direct worden geschreven. Na de acceptatietesten wordt het algoritme aangepast indien dit noodzakelijk is.

De overige Sales Requirements zijn te vinden onder bijlage F: “Sales requirements”.

[1] Scrum-term voor definitie van requirements

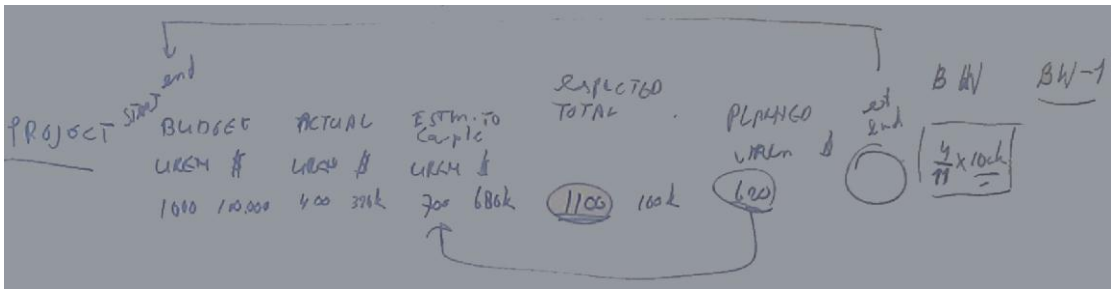
7.2 Boekhouder requirements

Eric wilt graag de mogelijkheid hebben om management informatie weer te geven voor projecten. Hier zou hij de volgende velden in terug willen zien komen:

- De begin- en einddatum van het project.
- Het aantal geplande uren en het projectbudget
- De daadwerkelijk gewerkte uren en het uitgegeven budget
- Een “Estimated Time To Complete” met geplande uren en uitgegeven budget
- Het verwachte totaal aantal uren en uitgegeven budget
- Een geschatte reële einddatum voor het project
- De boekwaarde van deze periode naast die van de vorige
- Een gewogen gemiddelde van de verkooptarieven van de medewerkers.

Een veld kan “geflagged” worden als een van de volgende situaties optreedt:

- De geschatte einddatum van het project overschrijdt de geplande einddatum
- Het geplande aantal uren of het geplande budget wordt overschreden



Verder wilt Eric de mogelijkheid hebben om de situatie in een bepaalde maand terug te halen. Hierbij zou gebruik gemaakt moeten worden van de laatste Estimated Time To Complete die op die bepaalde datum bekend was. Op basis van deze datum mogen de gegevens geëxtrapoleerd worden.

Niet alle data mag zomaar veranderen, een verandering in budget kan bijvoorbeeld worden opgelost door middel van het toevoegen van subprojecten. Het oorspronkelijke budget mag echter nooit worden aangepast, omdat dit een bepaalde schatting was op een historisch moment waar wellicht nog leer uit getrokken kan worden. Eric wil informatie zowel op project- als op subprojectniveau terug kunnen zien.

Geef projectgegevens weer	
Specifiek	- Het moet mogelijk worden om per project specifieke meetwaarden weer te geven, die in de pagina hiervoor uitgebreid worden besproken. Het doel hiervan is het verkrijgen van geschikte managementinformatie om zo betere inzichten te krijgen in het financiële verloop van het project. Vooral de CEO en boekhouder zijn geïnteresseerd in deze informatie.
Meetbaar	- Ik weet dat dit doel bereikt is als de meetwaarden die de CEO eerder heeft aangegeven te willen zien zichtbaar kunnen worden gemaakt op het scherm en ik grondige tests heb uitgevoerd om de correctheid van de berekening te controleren.
Acceptabel	-Dit doel kan worden bereikt door de projectgegevens die hiervoor benodigd zijn op te halen uit de Mus-database en hier optellingen voor te maken. Verder moet er een scherm in Mees worden gecreëerd waar de data in gepresenteerd kan worden.
Relevant	- Deze issue is de moeite waard om af te ronden omdat het de CEO en boekhouder gerichte informatie geeft om respectievelijk sturing op het project te kunnen geven en administratie bij te kunnen houden. Ik ben de juiste persoon om deze requirement af te ronden omdat ik na afronding van de Sales-view al aardig bekend ben met de Mus-data en uiteraard ook met de werking van Mees.
Tijdgebonden	- Deze issue wordt waarschijnlijk ergens na afronding van de eerste versie van de Sales View uitgevoerd omdat dit op dat moment het meeste waarde voor de organisatie toevoegt: de twee belangrijkste actoren van de applicatie kunnen dan beiden de kerntaken uitvoeren. Hierna kan extra functionaliteit worden toegevoegd op de verschillende views.

De overige Boekhouder Requirements zijn te vinden onder bijlage G: “Boekhouder requirements”.

7.3 Lijst uiteindelijke requirements

Nadat de verschillende requirements waren opgesteld heb ik deze in overleg met de product owner en stakeholders geprioriteerd. Hieronder volgt de lijst met requirements die ik heb opgesteld.

Omdat het in Scrum niet gebruikelijk is om van tevoren over zeer lange periode gedetailleerde planningen te maken, maar per sprint te kijken over wat op dat moment de meeste waarde voor de organisatie toevoegt is de lijst niet geordend op volgorde waarop deze afgerond worden. Wel valt er een globaal onderscheid te maken tussen de prioriteiten van de verschillende requirements zoals hieronder kan worden gezien:

Prioriteit rangorde	**Requirement**
Hoog	Geef de beschikbare uren voor medewerkers weer
Hoog	Sorteer medewerkers op beschikbaarheid
Hoog	Geef projectgegevens weer
Hoog	Geef ETC's tot een bepaalde tijd weer voor historische weergave
Middel	Geef winstverwachting van het project weer
Middel	ETC's kunnen worden weergegeven en ingevoerd
Laag	Sorteer medewerkers per klant
Laag	Geef jaarlijkse cumulatieve gegevens weer per klant

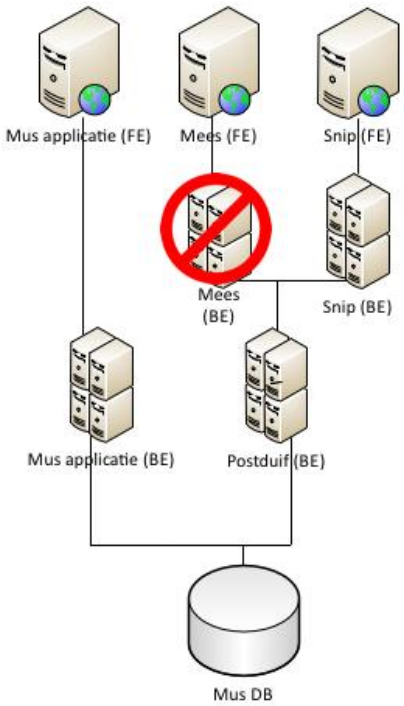
7.4 Grafische ontwerpen

Na het voeren van de gesprekken met de vertegenwoordigers van Sales en de Boekhouding had ik een redelijk beeld van welke informatie zij terug wilden zien in de verschillende views. Ik heb hierna een visueel ontwerp van beiden gemaakt die onder andere de eerder besproken Fisheye-view omvatte. Deze ontwerpen heb ik getoond aan beiden stakeholders en tevens aan de product owner. Na acceptatie hiervan kon begonnen worden aan de daadwerkelijke ontwikkeling van Mees.

De ontwerpen kunnen worden teruggezien onder bijlage H: "Grafische ontwerpen".

8 Incrementeel ontwerpen en ontwikkelen backend en frontend

8.1 Architectuur



Aanvankelijk was het plan om een Java-backend op te zetten die enerzijds via REST[1]-calls communiceert met de op te zetten webinterface en anderzijds de communicatie legt met de Mus database via de Java-module "Postduif" van de Volière. Echter, tijdens het selecteren van technologieën heb ik van diverse collega's te horen gekregen dat de methodiek die gebruikt werd in een andere applicatie binnen de Volière, Snip, niet volledig voldeed aan de verwachtingen. Omdat er bij Snip in feite sprake was van 2 backend-lagen creëerde dit telkens wat frictie bij het toevoegen van nieuwe functionaliteiten.

Voor mijn afstudeeropdracht wil ik een applicatie neerleggen die gemakkelijk te onderhouden is, daarom wordt er geen eigen backend voor Mees ontwikkeld en vinden de berekeningen plaats in Postduif. Er waren in Postduif nog geen serviceklassen[2] aanwezig voor eenzelfde soort functionaliteiten als die ik voor Mees nodig heb, dus kon ik deze toevoegen zonder dat andere applicaties hierdoor beïnvloed werden. De enige gedeelde code tussen de backend-functionaliteit van Mees en Snip waren de domeinklassen, die databasetabellen representeren. Deze waren in beiden gevallen exact gelijk omdat ze tegen dezelfde database praten. Dit conflicteerde dus evenmin. Een klassediagram van de domeinlaag is te vinden onder bijlage D: *"Klassediagram domeinlaag"*.

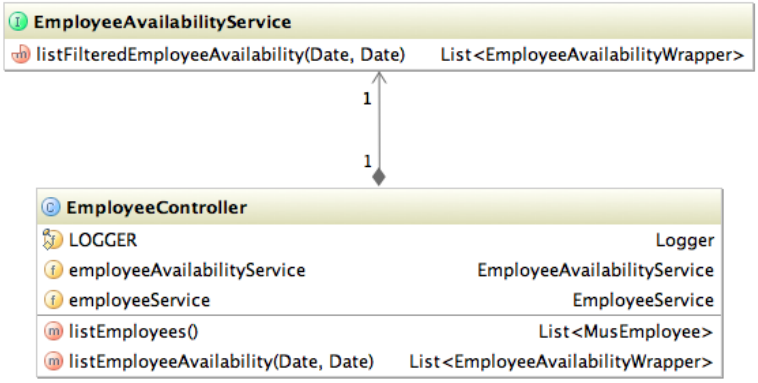
Postduif is verantwoordelijk voor alle communicatie tussen de verschillende applicaties van 42 en de Mus database. In de backend-uitbreiding die ik ga toevoegen wordt de juiste data opgehaald uit de Mus-database en na het uitvoeren van enkele berekeningen teruggegeven aan de webinterface.

De backend-functionaliteit wordt geprogrammeerd middels Test Driven Development om te garanderen dat de software die ik schrijf kwalitatief in orde is. Meer over dit onderwerp kan worden gelezen in hoofdstuk 9: "*Kwaliteitsbewaring*".

[1] Representational State Transfer, kan gebruikt worden om data op te vragen bij servers
[2] Klassen die verantwoordelijk zijn voor de business logica van een applicatie

8.2 Postduif Employee Controller

Allereerst ben ik begonnen met het schrijven van een Employee-controller waartegen calls uitgevoerd kunnen worden vanaf het web om informatie over de medewerkers en hun beschikbaarheid op te halen.

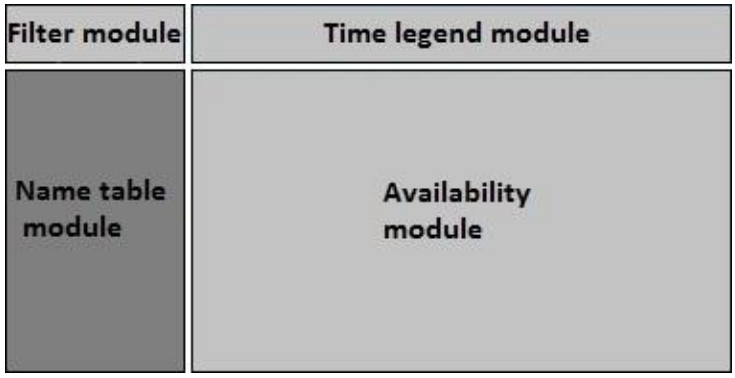


Dit klassediagram geeft de relatie aan tussen de EmployeeController en de EmployeeAvailabilityService. Voor een volledig overzicht van de controller- en het availability gedeelte van de servicelaag, zie respectievelijk bijlage B: *"Klassediagram controller-laag"* en bijlage C: *"Klassediagram availability-servicelaag"*.

De controller bevat onder meer een instantie van een serviceklasse die verantwoordelijk is voor de berekening van de medewerkersbeschikbaarheid. Voor de laatste werd een juiste structuur al geïmplementeerd, maar het ontwikkelen van de berekening zelf werd voor later bewaard. Op deze manier was het mogelijk om al tegen de controller aan te praten en een collectie medewerkers op te vragen. Deze gegevens konden vervolgens gebruikt worden om een tabel met namen van medewerkers op te bouwen.

8.3 Module-opzet

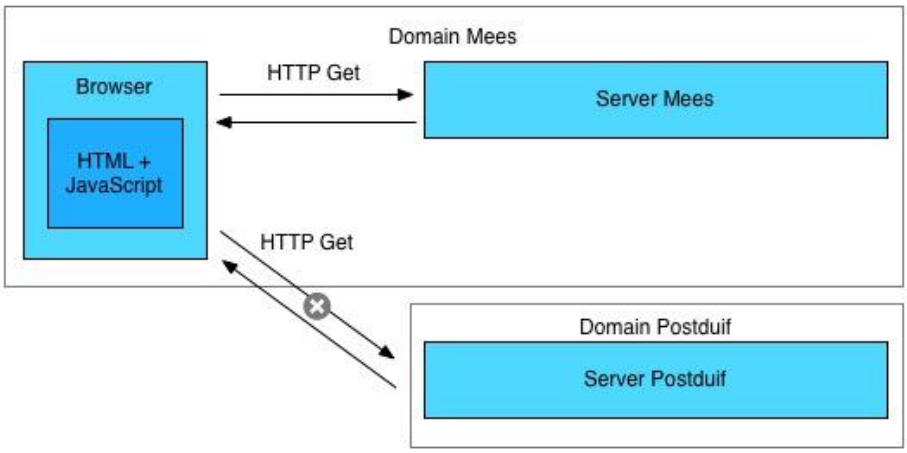
Na het schrijven van de Employee controller ben ik begonnen met het ontwikkelen van de verschillende Sales View modules die de Mees-pagina gaan vullen. Allereerst was dit de module waarin de namen van de medewerkers opgenomen worden. De reden dat ik hiermee ben begonnen was dat dit de verticale basis voor de layout vastlegde en dat zodra de module werd opgebouwd waarin medewerkersbeschikbaarheid werd aangegeven het direct duidelijk was bij welke medewerker de record hoorde. In de schematische weergave van de Sales View hierboven kan worden gezien waar de "Name table module" zich bevindt.



Ik heb een REST-call uitgevoerd naar de Employee Controller en de medewerkers en hun beschikbaarheid hieruit opgehaald, zodat ik direct daarna de andere waarin hun beschikbaarheid zichtbaar wordt gemaakt kan tekenen. Hier liep ik tegen een groot probleem van de ontwikkeling van Mees aan: mijn calls werden geblokkeerd en kwamen niet aan bij Postduif-server.

8.4 Cross-Domain call problemen

Na wat analyse over wat precies het probleem was ben ik erachter gekomen dat dit te maken heeft met de beveiliging die in browsers is opgenomen. Zij verbieden het uitvoeren van “Cross-domain calls”, omdat ze dit zien als een schending van de Same Origin Principle. *[2]* De reden dat deze calls door browsers geblokkeerd worden is het feit dat ze Cross Site Scripting tegen willen gaan, een techniek waarin JavaScript code wordt geïnjecteerd in webpagina’s om veelal malafide activiteiten uit te voeren als het stelen van data.



Met de volgende tabel kan inzicht worden verkregen in wat voor calls verboden zijn onder de Same Origin Principle. De calls worden uitgevoerd tegen de URL: “http://www.example.com/dir/page.html”.

Compared URL	Outcome	Reason
http://www.example.com/dir/page.html	Success	Same protocol and host
http://www.example.com/dir2/other.html	Success	Same protocol and host
http://www.example.com:**81**/dir/other.html	Failure	Same protocol and host but different port
https://www.example.com/dir/other.html	Failure	Different protocol
http://**en.example.com**/dir/other.html	Failure	Different host
http://**example.com**/dir/other.html	Failure	Different host (exact match required)
http://**v2.www.example.com**/dir/other.html	Failure	Different host (exact match required)

[3]

Zowel Mees als Postduif werkten op het lokale systeem onder de hostname “localhost”, maar niet op dezelfde port. Op de daadwerkelijke live-omgevingen is de port wel hetzelfde, maar verschilt de domain name. Beiden situaties zijn onder deze policy problematisch, zoals hierboven in de tabel gezien kan worden. Ik ben daarom op zoek gegaan naar conceptuele oplossingen voor dit probleem. De belangrijkste worden in dit hoofdstuk toegelicht.

De belangrijkste beslissingscriteria die golden bij het nemen van deze beslissing waren als volgt:

Criterium	Rationale
Veiligheid	De oplossing moet geen achterdeuren openzetten die misbruikt kunnen worden door hackers om toegang tot het systeem te verkrijgen.
Onderhoudbaarheid	Mees is de eerste applicatie in zijn soort (een web-applicatie zonder eigen backend) binnen de Volière, maar er volgen later meer. De oplossing moet gemakkelijk uit te breiden zijn voor deze projecten. Ook in het eventuele geval van verplaatsing van een client naar een ander domeinadres moet de oplossing gemakkelijk aan te passen zijn.
Portabiliteit	De oplossing moet gemakkelijk over te dragen zijn naar andere omgevingen. Zo praat de test-omgeving van Mees met de test-omgeving van Postduif, de acceptatie-omgeving van Mees naar de acceptatie-omgeving van Postduif, enzovoort. We willen niet calls van alle Mees-domeinen toestaan naar Postduif. De testomgeving mag bijvoorbeeld nooit een call uitvoeren naar de productie-omgeving.
Naleving beoogde architectuur	De oplossing moet voldoen aan de architectonische eisen van de organisatie, er moet direct gecommuniceerd kunnen worden tussen Mees en Postduif zonder tussenliggende systemen.

De uiteindelijke bedachte conceptoplossingen waren:

I. Cross Origin Resource Sharing
II. Set HTTP headers[1] in Postduif responsebody
III. Schrijf Mees-backend
IV. JSON[2] with padding gebruiken
V. Proxy-server opzetten.

De uitkomsten van de verschillende criteria worden in dit hoofdstuk in samenvattende vorm besproken. De volledige analyse is terug te vinden onder bijlage I: *"Volledige conceptoplossingen Same Origin Policy"*.

[1] Parameters van HTTP-transacties
[2] JavaScript Object Notation

8.4.1 Analyseren en afwegen mogelijkheden

Allereerst geef ik een korte uitleg over wat de verschillende mogelijkheden inhouden, verder in dit hoofdstuk beschrijf ik hoe de keuze voor een bepaalde oplossing tot stand is gekomen.

I. Cross Origin Resource Sharing

In deze methode wordt in de Apache-configuratie van Postduif ingesteld dat calls vanaf het webadres van Mees worden toegestaan. De server geeft bij een call een header mee die de browser verteld dat de call is toegestaan, waardoor deze het niet blokkeert. Deze technologie is vastgelegd in een W3C[1]-standaard[2].
Deze methode is veilig, omdat alleen domeinnamen die in de configuratie staan calls kunnen uitvoeren is makkelijk over te nemen naar een andere omgeving en past goed binnen de beoogde architectuur.

II. Set HTTP headers in Postduif responsebody

Een andere optie is het instellen van dezelfde HTTP-headers in de responsebody van de Postduif-controllers. Apache-configuratie is in dit geval niet benodigd.
Deze methode is om dezelfde reden als bij oplossing I veilig. Ook past deze binnen de beoogde architectuur. De portabiliteit is echter minder dan bij de vorige oplossing, omdat er geen scheiding is tussen de programmacode en externe configuratie. Dit betekend dat alle omgevingen van Postduif calls vanaf alle omgevingen van Mees toe moeten staan, wat dus een minder nette scheiding is dan de vorige oplossing.

III. Schrijf Mees-backend

Deze oplossing houdt in dat er alsnog een backend voor Mees wordt gebouwd van waaruit de calls worden uitgevoerd, de browser blokkeert in dit geval de calls dus ook niet. Deze oplossing is veilig en portable, maar creëert een extra laag die 42 BV juist niet wilt hebben en strookt daarom niet met de beoogde architectuur.

IV. JSON with padding (JSONP) gebruiken

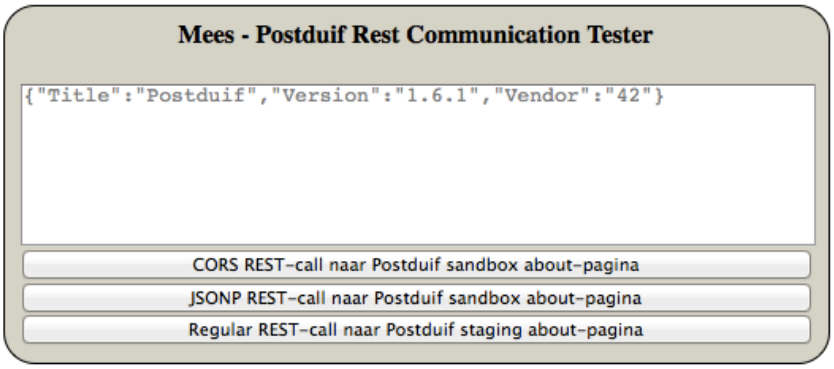
Een andere oplossing is het gebruik van JSON with padding in plaats van reguliere JSON. Dit misbruikt een zwakte in de Same Origin Policy om calls naar andere adressen toch mogelijk te maken door de teruggegeven waarde in een script-tag in te laden. Enige server-side aanpassing is benodigd, maar dit zet de deur open voor calls van ieder domeinadres en is dus minder veilig dan de bovengenoemde oplossingen. Het is portable en leeft de beoogde architectuur na. Omdat het eigenlijk gaat om een hack en het server-side aanpassingen benodigd is besloten niet voor deze oplossing te kiezen.

[1] Organisatie die standaarden voor het web ontwikkelt

V. Proxy-server opzetten

Tenslotte is er de mogelijkheid om de call via een proxy te laten redirecten, het uitgaande “Origin” adres wordt hierdoor zo aangepast dat de teruggegeven waarde afkomstig lijkt te zijn van Mees zelf. De oplossing is veilig, onderhoudbaar en portable maar strookt niet met de beoogde architectuur, omdat de proxy hier eigenlijk de rol van een backend op zich neemt.

8.4.2 Proof-of-Concept



Voordat ik mijn advies heb uitgebracht aan de organisatie heb ik de mogelijkheden die ik als serieuze opties zag, namelijk CORS en JSONP, getest in Proof-of-Concepts. Ik heb me hiertoe beperkt omdat het vrij duidelijk was dat het een van deze twee opties zou worden, aangezien de andere ofwel een backend vereisten of een backend simuleren, wat 42 BV juist niet meer wilde met Mees.

Hierin heb ik onder andere getest of de headers aan de Postduif of Mees-kant moesten worden neergezet en of de teruggave van JSONP inderdaad zonder enige overige configuratie werkte. Deze tests waren nog vrij basaal en heb ik op mijn lokale systeem uitgevoerd. De calls zonder CORS of JSONP werden hier geblokkeerd, omdat de servers op verschillende porten werken.

8.4.3 Overleg en beslissing

Omdat de keuze die ik maak voor de oplossing van dit probleem architectonische gevolgen heeft, heb ik besloten om een meeting team bij elkaar te roepen waar ik de verschillende opties aan heb uitgelegd en per punt heb verteld wat ik hierbij als de voor- en nadelen zag, en of ik een implementatie wel of niet zou adviseren.

Het gebruik van de headers voor Cross Origin Resource Sharing (zie: "*Set HTTP headers in Apache configuratie Postduif*") had mijn voorkeur, omdat dit een veilige oplossing is die goed is uit te breiden naar andere omgevingen, goed past binnen de architectonische wensen van de applicatie en is vastgelegd in een W3C-standaard, wat de kans groot maakt dat steeds meer browsers dit gaan ondersteunen. De browsers die bij 42 worden gebruikt doen dit op dit moment al. Op dit moment is de ondersteuning al behoorlijk goed, zoals kan worden ingezien in bijlage K: *"*Cross Origin Resource Sharing browser support*"*.

Ik vond het belangrijk dat deze keuze genoeg draagvlak zou vinden binnen de organisatie, omdat Mees een voorbeeldproject is voor de opvolgende backend-loze applicaties waardoor ik direct een goede oplossing wilde neerleggen die ook in die projecten inzetbaar is.

Voor iedere concept-oplossing werden de voor een nadelen besproken, en uiteindelijk was het team er mee eens dat deze methode de beste oplossing voor het probleem is om dezelfde redenen als hierboven genoemd.

Uiteindelijk heb ik een live-demoversie van de CORS Proof-of-Concept opgezet op de Mees testserver. Deze sprak met twee Postduif-servers waarvan een de header had en de andere niet. Het uploaden van deze demoversie naar de server diende vooral als demo-pagina en om te verifiëren dat er geen server-specifieke problemen ontstonden. Zoals verwacht werkte het echter net als op mijn lokale omgeving. Zie bijlage J: *"*Proof-of-concept Cross Origin Resource Sharing*"* voor meer informatie.

8.4.4 Implementatie

Na het ontwikkelen van een Proof-of-concept voor het probleem waarmee de precieze werking mij duidelijk werd en de werking hiervan werd bewezen, ben ik aan de slag gegaan om deze oplossing ook in Mees zelf te integreren. Dankzij de tests die ik al had gedaan was dit eenvoudig, het enige dat nog opgelost moest worden was het oplossen van timingsfouten die nu optraden bij calls naar Postduif.

jQuery voert calls standaard asynchroon uit, dus naast het uitvoeren van andere code. Omdat de rest van mijn code afhankelijk was op het JSON object dat ik van Postduif terugkrijg heb ik deze call synchroon moeten maken. Dit betekent dat jQuery eerst wacht tot het JSON-object wordt teruggeven voor hij verder gaat met het uitvoeren van de rest van de code. Indien er een error status teruggegeven wordt vanaf Postduif wordt er een foutmelding op de pagina weergegeven in plaats van de gewenste medewerkersbeschikbaarheid.

8.4.5 Presentatie

Omdat er binnenkort andere collega's applicaties met een soortgelijke architectuur als Mees gaan programmeren heeft de Chief Technology Officer van 42 mij gevraagd om een presentatie te geven bij de Knowledge Transfer Meeting die een week later was ingepland, met als doel het overbrengen van mijn kennis van het onderwerp aan diegenen die kort daarna tegen dezelfde problemen aan zouden gaan lopen.

Ik heb de presentatie voorbereid en geoefend met andere presentatoren. Toen de uiteindelijke presentatie werd gegeven reageerden collega's positief en enthousiast op de materie en beoogde oplossing.

8.5 Berekening medewerkersbeschikbaarheid

8.5.1 Plan

Zoals van tevoren al voorzien bleek het berekenen van de medewerkersbeschikbaarheid een complexe taak te zijn. Allereerst waren er diverse factoren waarmee rekening gehouden moest worden: zoals het inzetpercentage van medewerkers aan projecten, het werkrooster van medewerkers, vakantiedagen en urenregistraties.

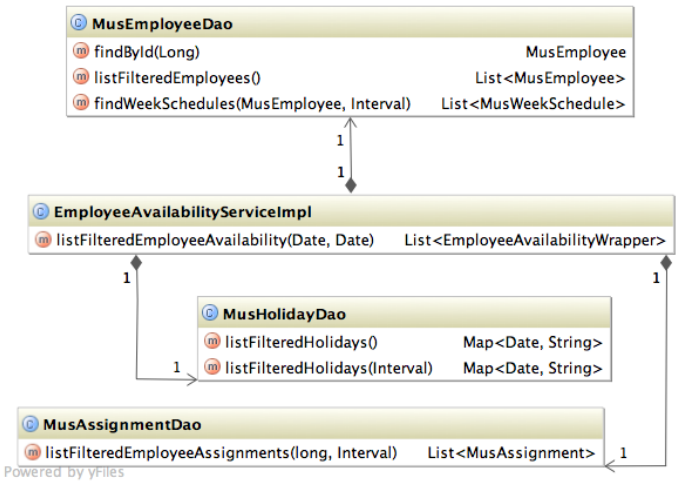
Verder is de datakwaliteit van de Mus database niet altijd even goed, deze database is in de loop der jaren steeds verder uitgebouwd en het bevat meerdere tabelstructuren en naamgevingsconventies. Ook zijn gegevens als weekroosters en inzetpercentages niet altijd aanwezig. De afwezigheid hiervan maakt het berekenen van medewerkersbeschikbaarheid zonder het maken van aannames onmogelijk. Omdat deze applicatie echt gebruikt gaat worden door de Sales Director, Ron Oudshoorn, willen we geen berekeningen doen op basis van aannames.

Daarom werd besloten om alle datums waarvoor onvoldoende gegevens beschikbaar zijn om de berekening uit te voeren als onjuist te markeren. Deze verschijnen bovenaan in het overzicht. Dit moet dienen als prikkel om de benodigde data in te voeren in Mus.

De rest van de medewerkers worden gesorteerd naar aanleiding van de beschikbaarheid in de aankomende maand. De medewerkers die het meest beschikbaar zijn verschijnen bovenaan in het overzicht, omdat inzet voor deze medewerkers het meest urgent is.

8.5.2 Uitwerking

Omdat de structuur van de Employee controller-, service en data-access-object[1] klassen al gecreëerd was kon ik hier nu een implementatie achter hangen. Allereerst heb ik de invulling van enkele data-access-objects geschreven, onder andere voor Holidays en Employees, zodat ik kon beginnen met het uitvoeren van berekeningen.



Zie voor een klassediagram van de data-access-objectlaag op volledige grootte bijlage E: "Klassediagram data-access-object laag".

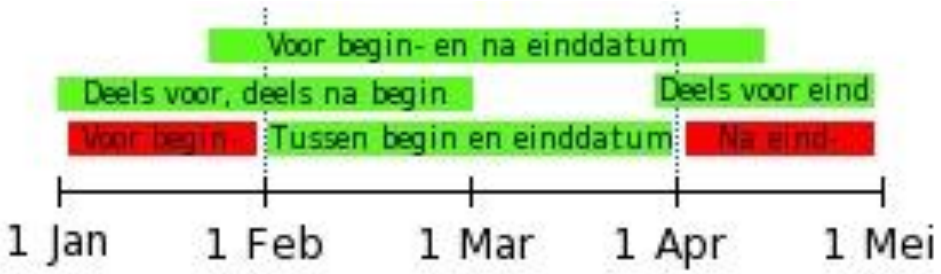
[1] Klasse die verantwoordelijk is voor de communicatie met de database

8.5.2.1 Assignaties

In de Mus-database wordt een koppeling tussen een medewerker en een project een assignatie (of assignment) genoemd. Deze bevatten onder ander een begin- en einddatum en bepalen hoe lang een specifieke medewerker aan een specifiek project zit. Aan deze assignaties hangen vervolgens weer urenregistraties. In dit project hou ik de terminologie die door 42 BV gebruikt wordt aan.

Oorspronkelijk dacht ik dat de assignaties via de Employees opgehaald konden worden en dat ik in de join-conditie op kon geven dat enkel de assignaties binnen een bepaalde periode worden opgehaald. Er werd echter gebruik gemaakt van een query-taal voor Java, QueryDSL, waarin dit niet mogelijk is.

Aanvankelijk werden alle assignaties van een Employee opgehaald, maar werden enkel die assignaties die relevant waren voor de opgevraagde periode verwerkt. Toen ik heb geprobeerd om hier een join-conditie aan te koppelen bleek dit met de Java Persistence API niet mogelijk te zijn. Daarom heb ik een Assignment DAO geschreven die alle assignaties voor een bepaalde periode relevant is voor een bepaalde medewerker.

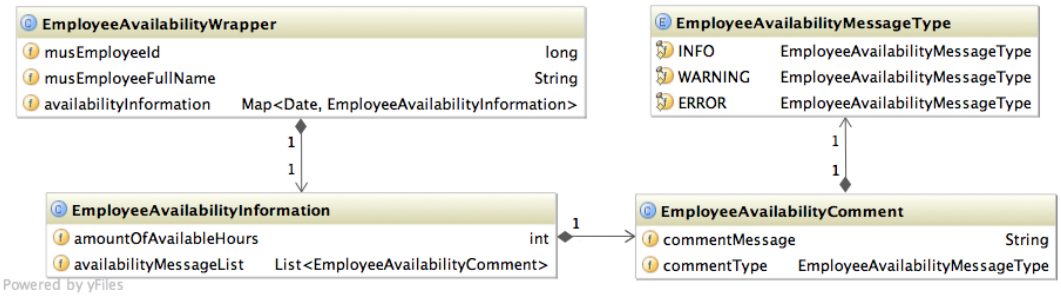


Zoals hierboven schematisch is weergegeven wordt een assignatie als relevant beschouwd als deze zich binnen de begin- en einddatum van de opgevraagde periode bevindt, of als tenminste een deel van deze periode waarop de assignatie actief is overlapt met de periode waarvoor de beschikbaarheid wordt opgevraagd. Voor elke dag waarop de beschikbaarheid wordt berekend wordt vervolgens bekeken of de assignaties die groen zijn gemarkeerd op die specifieke dag nog actief zijn. De rode assignaties worden überhaupt niet opgehaald uit de database omdat deze gegevens niet interessant zijn.

8.5.2.2 Datastructuur beschikbaarheidsteruggave

Vervolgens heb ik een wrapper-klasse ontwikkeld die informatie over een medewerker bevat en een TreeMap[1] waarin de beschikbare uren voor de verschillende dagen binnen de periode in gaan verschijnen. De datums werden hierbij als keys gebruikt en hierdoor werd de TreeMap er impliciet hierop gesorteerd.

De structuur van de wrapper is enigszins aangepast tijdens de ontwikkeling van de berekening van medewerkersbeschikbaarheid, om zo efficiënt mogelijk gegevens heen en weer te kunnen sturen. Niet alle informatie over de medewerkers is relevant om weer te geven in de beschikbaarheidsview, enkel de identifier en naam zijn voldoende. De grootte van dit soort informatiestromen beperken zorgt voor een efficiëntere en snellere communicatie.



De wrapper bevat op zijn beurt een EmployeeAvailabilityInformation object, dat een aantal beschikbare uren bevat en een lijst met AvailabilityMessages. Deze messages kunnen van drie types zijn:

Info
Het informatie type geeft extra informatie die een verklaring kan geven bij een aantal beschikbare uren; zoals een uitleg van overallocatie.

Warning
Het warning-type toont een validatie-uitkomst van de bijbehorende Mus-data die de berekening van medewerkersbeschikbaarheid niet onmogelijk maakt.

Error
Het error-type toont validatie-uitkomsten die de berekening van medewerkersbeschikbaarheid onmogelijk maken. Een medewerker die de laatste van de drie types bevat wordt bovenaan het overzicht van medewerkersbeschikbaarheid getoond en “geflagged”.

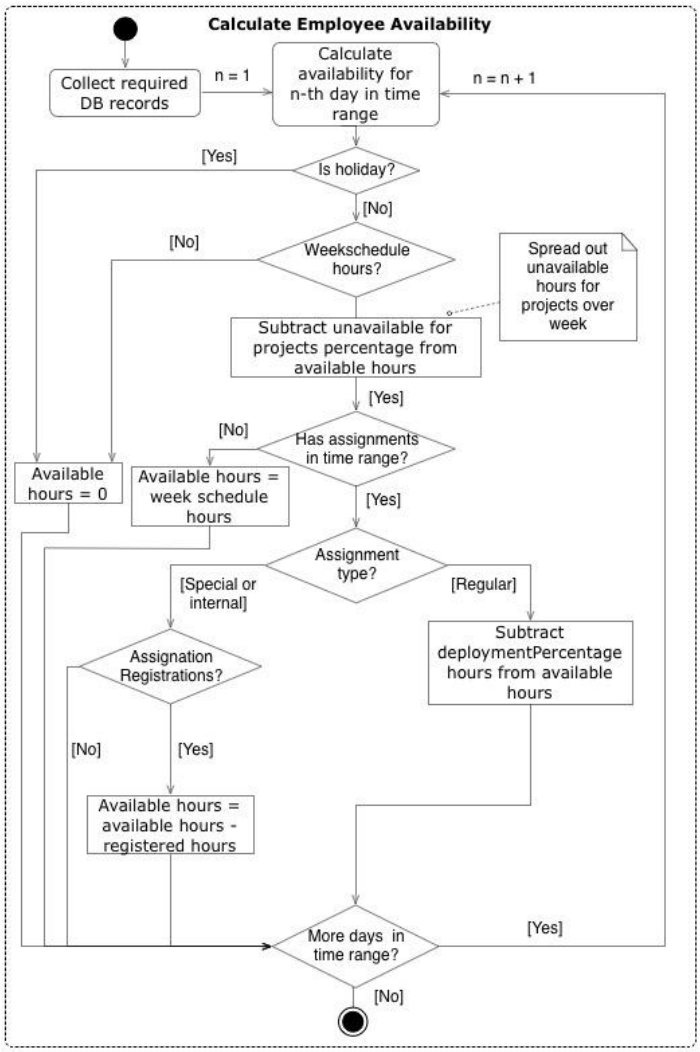
[1] Geordende implementatie van Java’s map interface, gebruikt key/value pairs.

8.5.2.3 Calculatie-algoritme

Om te voorkomen dat er gedurende het project verwarring zou ontstaan over de betekenis en het gebruik van de Mus-data heb ik een Activity Diagram uitgetekend waarin ik het hele proces van de uitrekening van medewerkersbeschikbaarheid in kaart heb gebracht. De versie op volledige grootte kan worden gevonden onder bijlage L: “Activity diagram berekening medewerkersbeschikbaarheid”.

Ik heb de Activity Diagram getoond aan de product owner om te valideren dat ik de betekenis van alle data en het bijbehorende proces goed begrepen had, waarna begonnen worden aan de daadwerkelijke ontwikkeling van het algoritme.

De reden dat ik niet heb gekozen voor een andere methodiek om de data in kaart te brengen, zoals een data dictionary, is het feit dat ik niet alleen maar iets wilde vastleggen over de betekenis van de Mus-data maar tevens hoe deze gebruikt dient te worden. Een Activity Diagram voorzag in beiden behoeften, het kon zowel gebruikt worden als geheugensteun voor mijzelf of als medium om bij de product owner te valideren of het door mij uitgedachte proces klopte. Een data dictionary opstellen was hierna dus niet meer noodzakelijk.



Zoals in het Activity Diagram kan worden gezien is het berekenen van de medewerkersbeschikbaarheid een complexe taak. Het omvat meerdere tabellen uit de Mus database en kan meerdere scenario’s per medewerker per dag bevatten. Hieronder volgt een korte toelichting op het Activity Diagram.

Wanneer een beschikbaarheid voor een medewerker op een bepaalde dag wordt uitgerekend wordt allereerst gekeken of de dag in de lijst van verplichte vrije dagen (in de Mus-database “Holiday” genoemd) voorkomt.

Als dit het geval is weten we dat de medewerker nul uur beschikbaar is en kan de beschikbaarheid van de volgende dag worden uitgerekend.

Indien het niet gaat om een Holiday wordt gekeken naar het aantal werkuren in het weekrooster (in de Mus-database WeekSchedule genoemd) van de medewerker. Deze bevatten voor iedere werkdag een aantal uur dat de medewerker werkt, alsmede een startdatum. Een WeekSchedule blijft actief tot de datum dat een volgende WeekSchedule ingaat.
Het getal dat uit het weekrooster wordt gehaald geldt als basis voor de medewerkersbeschikbaarheid. Het kan nooit meer zijn dan dit getal. Indien het aantal uren voor de dag uit het weekrooster nul is wordt er weer doorgegaan naar de berekening van de volgende dag.

Iedere medewerker heeft een percentage dat aangeeft voor welk deel van de tijd dat zij werken ze beschikbaar zijn voor projecten (in de Mus-database "availableForProjectsPercentage" genoemd). Het percentage wordt geïnverteerd zodat we het percentage hebben dat de onbeschikbaarheid van medewerkers voor projecten aangeeft. Dit percentage wordt verdeeld over het aantal werkdagen dat een Employee werkt in een week afgetrokken. Hierbij wordt afgerond op hele uren omdat medewerkers van 42 nooit een ongeheel aantal uren werken bij een klant.

Stel dat een medewerker 40 uur per week werkt en een project-beschikbaarheidspercentage heeft van 90% dan is de uitkomst van deze som dus een afgeronde 7 uur beschikbaarheid per dag.

Indien het aantal beschikbare uren voor de dag waarvoor we de beschikbaarheid aan het berekenen zijn meer is dan nul worden de relevante assignaties opgehaald. Dit gebeurd op de manier zoals eerder in dit hoofdstuk werd beschreven.

Deze assignaties kunnen van verschillende typen zijn die allemaal ander gedrag met zich meebrengen. Hieronder een overzicht van de verschillende mogelijkheden, de concrete situatie die hierbij hoort en de praktische betekenis voor de berekening van medewerkersbeschikbaarheid.

Assignatie-type	**Concrete situatie**	**Betekenis availability**
Intern	Lijnactiviteiten voor 42	Gebruik urenregistraties
Special	Betaald speciaal project	Gebruik urenregistraties
Non-special, non-billable	Investeringsprojecten 42	Negeer, medewerkers zijn nog steeds beschikbaar.
Non-special, billable	Betaald regulier project	Gebruik deploymentPercentage

Zoals hierboven te lezen is zijn er feitelijk twee situaties waar we iets mee moeten doen: een situatie waarin we urenregistraties gebruiken om beschikbaarheid uit te rekenen en een situatie waarin we een "deploymentPercentage" gebruiken om beschikbaarheid uit te rekenen.

Voordat ik uitleg hoe deze processen eruit ziet geef ik eerst een korte toelichting over wat deze gegevens precies betekenen.

In een gewone situatie zijn urenregistraties registraties die medewerkers doen om aan te geven dat zij op een bepaalde dag een bepaald aantal uren gewerkt hebben. Dit is echter anders voor speciale projecten en interne lijnactiviteiten van 42.

Onder lijnactiviteiten worden langdurige werkzaamheden verstaan die eigenlijk geen projecten zijn, zoals bijvoorbeeld het bijwonen van een vergadering als medewerkersvertegenwoordiger. Dit soort activiteiten worden van tevoren ingepland en worden daarom dan ook van tevoren geregistreerd.

Onder speciale projecten worden allerlei bijzondere reden van afwezigheid geregistreerd waarvan de momenten al van tevoren bekend zijn, zoals bijvoorbeeld dokterbezoek, ouderschapsverlof of verlof. Ook hiervan worden de uren van tevoren geregistreerd.

De moeilijkheid hiervan is dus dat er meerdere situaties zijn waarin data uit dezelfde kolommen andere betekenissen hebben. Dit is helaas niet de enige onduidelijkheid of ambiguïteit die in de Mus database aanwezig is. Een ander voorbeeld is de “deploymentPercentage” die present is bij assignaties op reguliere projecten. Dit getal wordt gebruikt om te berekenen voor welk deel van de tijd die medewerkers beschikbaar zijn voor dat specifieke project waar de assignatie naar wijst. In tegenstelling tot het “availableForProjects” percentage is dit percentage echter niet gebaseerd op het totaal aantal uren uit een Employee’s WeekSchedule maar op een complete 40-urige werkweek.

Uiteindelijk worden voor de assignatie op reguliere projecten met het deploymentPercentage uitgerekend hoeveel uren een medewerker per week besteed aan dat project. Dit wordt verdeeld over het aantal dagen dat hij werkt en het deel wordt afgetrokken van de dag waarvoor medewerkersbeschikbaarheid wordt uitgerekend.

Voor speciale projecten is de berekening wat gemakkelijker: het aantal uren van de registraties die op de specifieke dag waarvoor de medewerkersbeschikbaarheid wordt uitgerekend geldig zijn worden afgetrokken van het overgebleven beschikbaar aantal uren.

Dit proces wordt vervolgens herhaald voor iedere dag in de reeks waarvoor medewerkersbeschikbaarheid wordt aangevraagd, en voor iedere medewerker die werkzaam is bij 42 BV.

8.5.2.4 Validatie Mus-data

Los van het feit dat de datastructuur van de Mus-database in sommige gevallen zeer verwarrend is houdt de kwaliteit van de aanwezige data zelf vaak ook te wensen over. De database bevat geen triggers en de code-base van de Mus-applicatie is dusdanig groot en complex geworden dat fouten die datavervuiling konden veroorzaken vaak lange tijd aanwezig blijven, tot iemand opvalt dat er iets misgaat.

Langzaamaan is 42 BV bezig om deze applicatie uit te faseren en de datakwaliteit toe te laten nemen, echter gaat hier dusdanig veel tijd in zitten dat het nog een redelijke tijd kan duren voordat de situatie verbeterd. Dit betekende voor mijn werk dus dat het zeer belangrijk was om foute situaties in de Mus database op te sporen voordat deze mee worden genomen in een berekening.

De matige datakwaliteit was dus extra reden om de gegevens die ik ophaal uit de Mus-database goed te valideren. Dit heb ik gedaan door een klasse te schrijven die bepaalde datakwaliteitsschendingen opspoort en als foutief markeert. Hierin gaat het in de meeste gevallen om schendingen die het onmogelijk maakten om de medewerkersbeschikbaarheid te berekenen zonder te vervallen in allerlei assumpties als “zonder WeekSchedule gaan we er vanuit dat een medewerker veertig uur werkt”.

Omdat het niet wenselijk is dat de Sales-afdeling allerlei informatie krijgt die eigenlijk het verkeerde beeld schetsen heb ik in overleg met de product owner afgesproken dat de dagen waarvoor om dit soort redenen geen beschikbaarheid kan worden uitgerekend rood gemarkeerd worden en nul uren van beschikbaarheid aangeven. Verder wordt een medewerker waarbij dit in de periode minstens één keer voorkomt bovenaan het overzicht van medewerkersbeschikbaarheid gezet. Op deze manier geeft dit een duidelijk signaal af om de datakwaliteit te verhogen.

Ik heb geadviseerd om dit soort datavervuiling ook bij invoer in de Mus-applicatie te valideren en af te straffen, zodat er nieuwe vervuiling meer optreedt. Verder heb ik geadviseerd om dit soort onmogelijkheden keihard door de database te laten verbieden om een “laatste verdedigingslinie” te bieden tegen dit probleem. Het uitvoeren hiervan valt echter buiten de scope van mijn eigen afstudeeropdracht en wordt daarom door iemand anders gedaan.

8.5.2.5 Testen algoritme

Tijdens het schrijven van het algoritme dat medewerkersbeschikbaarheid uitrekent heb ik uitvoerige tests geschreven om te valideren dat de verschillende routes die ik in het Activity Diagram heb uitgetekend het verwachte gedrag vertonen.

Voor ieder pad heb ik test data geschreven die een bepaalde route triggert in het diagram. Ik heb zelf uitgerekend wat de uitkomst van het algoritme moest zijn en vervolgens gekeken of de uitkomst overeenkwam met mijn verwachting.

```
MusAssignment regularAssignment = new MusAssignment(1L);
Date startDateOfProject = new DateMidnight(2012, 1, 1).toDate();
Date endDateOfProject = new DateMidnight(2012, 3, 1).toDate();
int deploymentPercentage = 60;
setFieldValue(regularAssignment, "startDate", startDateOfProject);
setFieldValue(regularAssignment, "endDate", endDateOfProject);
setFieldValue(regularAssignment, "deploymentPercentage", deploymentPercentage);
setFieldValue(regularAssignment, "project", regularProject);
assignments.add(regularAssignment);
```

Ook heb ik bij het testen rekening gehouden met combinaties van factoren die elkaar kunnen beïnvloeden: zo veranderd een combinatie van een speciaal en regulier project de berekening iets. De geregistreerde uren van de speciale projecten worden namelijk als eerste van de beschikbare uren afgetrokken. Vervolgens worden de deploymentPercentages van de reguliere projecten berekend over de restwaarde die overblijft na aftrek van de speciale urenregistraties.

EmployeeAvailabilityCalculatorTest (nl.mad.voliere.postduif.service.availability)
- testCalculateAvailableHoursForEmployeeNoAssignments
- testCalculateAvailableHoursForEmployeeAssignedToRegularProjectOnly
- testCalculateAvailableHoursForEmployeeAssignedToSpecialProjectOnly
- testCalculateAvailableHoursForEmployeeAssignedToInternalProjectOnly
- testCalculateAvailableHoursForEmployeeAssignedToBoth
- testCalculateAvailableHoursForEmployeeWithNegativeAvailabilityBecauseOfDeploymentPercentage
- testCalculateAvailableHoursForEmployeeHigherDeploymentPercentageThanWeekScheduleHours

De EmployeeAvailabilityCalculator wordt niet aangeroepen als het een datum betreft die een Holiday is, omdat in dat geval de berekening niet benodigd is. In de klasse die de calculator aanroept heb ik ook enkele tests geschreven die onder andere valideren of een datum die een Holiday is nul beschikbare uren teruggeeft.

Verder heb ik ook de validatie uitvoerig getest, alle mogelijke violations aan de Mus-data zijn gesimuleerd, waarna ik heb gecontroleerd of de validator deze ook als zodanig opmerkt en teruggeeft.

DefaultEmployeeAvailabilityValidationImplTest (nl.mad.voliere.postduif.service.availability.validation)
- testValidateHappyFlow
- testValidateNoWeekSchedule
- testValidateNoProjectAvailabilitySpecified
- testValidateInvalidProjectAvailabilityPercentage
- testValidateProjectAvailabilityPercentageMoreThanWeekScheduleHours
- testValidateAvailabilityViolationForNoDeploymentPercentage
- testValidateAvailabilityViolationForInvalidDeploymentPercentage

8.5.2.6 Testscenario's

Zoals eerder beschreven zijn er bij het testen van het algoritme dat medewerkersbeschikbaarheid uitrekent meerdere testscenario's opgesteld. Hierop volgt een korte toelichting over de invoer, verwerking en uitvoer van de belangrijksten hiervan.

De eerste belangrijke situatie is er een waarin een medewerker beschikt over een weekrooster met ingevulde uren, maar niet over assignaties (en dus ook urenregistraties). In dat geval verwachten we dat het aantal beschikbare uren dat voor die dag is aangegeven ook wordt teruggegeven als beschikbaar aantal uren.

Een andere testsituatie bevat een WeekSchedule en een assignatie naar een regulier project. Stel dat het deploymentPercentage van de assignatie 60% is en een medewerker voor 40 uur per week in dienst is. Dit betekent dat voor de specifieke datum een medewerker voor 60% van de volledige werkdag van acht uur (immers, hij is full-time medewerker) is ingezet op een project. Dit komt overeen met een afgeronde vijf uur. In dit geval verwachten we dus ook dat de medewerker voor die specifieke dag nog drie uur beschikbaar is.

Een testsituatie waarin medewerker een assignatie heeft idee toebehoort aan een speciaal of intern project is ook aanwezig. Hierin worden de deploymentPercentages niet gebruikt, maar wordt gekeken naar urenregistraties ie aan de assignatie toebehoren. Stel dat een medewerker een urenregistratie heeft staan voor de datum waarvoor we de beschikbaarheid uit willen rekenen van zes uur en deze medewerker full-time werkt, dan is hij dus twee uur beschikbaar op die dag.

Een combinatie van de twee types assignaties maakt de berekening iets complexer: allereerst worden de urenregistraties van de speciale/interne projecten van het basisgetal afgetrokken, en over het restgetal worden de deploymentPercentages berekend. Stel dat een medewerker een speciale assignatie heeft voor twee uur, en een reguliere assignatie voor 60%. Wederom gaat het hier over een full-time medewerker. Het basisgetal van acht uur wordt verminderd met de twee uur waarvoor de medewerker al is toegewezen aan het speciale project. Dit betekent dat het deploymentPercentage voor het andere project over de resterende zes uur wordt berekend, wat neerkomt op een afgeronde vier uur. De medewerker is voor die dag dus twee uur beschikbaar.

8.6 Onderzoek JavaScript MVC frameworks

8.6.1 Probleemstelling

Omdat 42 BV de komende tijd meer losstaande HTML/JavaScript projecten zonder middenlaag wilt creëren wordt het extra belangrijk om de aanwezige JavaScript-code goed te structureren. Daarom ben ik op zoek gegaan naar enkele (MVC) frameworks die ons in deze behoefte konden voorzien.

Er was voor Mees aanvankelijk geen standaard op dit gebied, dus is enkel gebruik gemaakt van standaard JavaScript en jQuery. Naarmate Mees groter wordt groeit echter ook de behoefte aan gestructureerde JavaScript-code. Omdat dit project behalve een realisatie van een daadwerkelijk systeem ook bedoeld is als test of een middenlaagloos systeem beter bevalt dan bijvoorbeeld een Grails-applicatie, is besloten dat ik dit als een eerste ga proberen en mijn bevindingen vastleg op de corporate Wiki van 42 BV: Confluence.

De volgende JavaScript frameworks zijn als optie overwogen:
JavaScript MVC, Knockout-js, Backbone-js, Bootstrap, Ember, Mustache

Deze lijst is samengesteld na een zoektocht op het internet en is beoordeeld aan de hand van de volgende criteria die ik vooraf heb opgesteld:

Methode structurering
Belangrijkste criterium, een MVC-aanpak geniet hier de voorkeur boven een andere omdat dit een duidelijkere structurering met zich meebrengt.

Omvang framework
De grootte van het framework zelf heeft ook invloed op de beslissing, een framework dat zo klein mogelijk is dat deze het minste tijd en bandbreedte kost om op te halen van de server heeft hier de voorkeur.

Documentatie framework
Een framework waarvan de functionaliteiten goed gedocumenteerd zijn op het internet, of waarvoor actieve internetcommunities ontstaan genieten de voorkeur boven frameworks waar dit niet het geval is.

Om een goede beslissing te kunnen nemen die draagvlak voor hergebruik bij andere applicaties kan vinden binnen de organisatie heb ik een meeting ingepland. Hierbij waren diverse medewerkers van 42 BV met expertise van onder andere JavaScript betrokken. De criteria die ik had bedacht werden aangevuld met twee criteria “HTML markup vs Declaratieve, Selector-based data-invoer” en “Frontend + Backend positionering vs Frontend positionering”.

Deze criteria zijn samengesteld op basis van de bestaande kwaliteitsattributen die 42 BV hanteert voor haar Java-software, alsmede enkele browser-specifieke complicaties als package-grootte. Zie voor meer informatie hoofdstuk 9: *“Kwaliteitsbewaring”*.

8.6.2 Vergelijking

We hebben de verschillende frameworks geanalyseerd, besproken en ingedeeld in enkele categorien zoals hieronder gezien kan worden. Met het vinkje is aangegeven wat 42 BV als “de wenselijke optie” beschouwt.

Binding	*HTML Markup* Mustache	*Declaratief, selector-based* ✓ Backbone-js	*Onindeelbaar* JavaScript MVC, ~~Bootstrap~~, Ember, Knockout-js
Structurering code	*MVC (like)* ✓ JavaScript MVC, Knockout-js, Backbone-js, Ember	*Spaghetti* ~~Bootstrap~~	*Onbekend* Mustache
Omvang framework	*Groot* JavaScript MVC, Ember	*Klein* ✓ Knockout-js, Backbone-js, ~~Bootstrap~~, Mustache	
Documentatie framework	*Veel en goed* ✓ JavaScript MVC, Knockout-js, Backbone-js	*Weinig en slecht* Ember	*Onbekend* ~~Bootstrap~~, Mustache
Positie in architectuur	*Frontend + Backend* JavaScript MVC, Ember	*Frontend* ✓ Knockout-js, Backbone-js, Bootstrap, Mustache	

Op de volgende pagina staat een korte toelichting wat de rationale achter de voorkeuren was en in welke categorieën de frameworks vallen:

HTML Markup vs Declaratief, selector-based

Over dit punt was tussen de leden van de meeting onderling de meeste disussie: enerzijds biedt een HTML Markup binding je de mogelijkheid om duidelijk aan te kunnen geven wat voor data er in een bepaald element gaat verschijnen zodat je met één blik op de HTML al een voorspelling over de toekomstige inhoud kunt maken. Anderzijds zou je dit vervuiling van de HTML kunnen noemen en zou een id- of class-selector een nettere vorm van programmeren zijn.

Verder biedt een selector-based oplossing je de mogelijkheid om later een aanpassing op exact dezelfde manier te kunnen doen als eerder, in tegenstelling tot bij de HTML markup optie.Vanwege dit laatste argument is uiteindelijk is de voorkeur gegeven aan de declaratieve optie. Op dit punt komt Backbone-js dan ook beter uit de vergelijking dan de rest.

MVC (like) vs Spaghetti

Het MVC-pattern biedt goede mogelijkheden voor separation of concerns in tegenstelling de kluwen spaghetticode die kunnen ontstaan als er zonder een duidelijk patroon als deze ontwikkeld wordt. Het is duidelijk waarom de eerste optie de voorkeur geniet boven de tweede. Enkel Bootstrap valt hier dus buiten de boot, dit is reden genoeg geweest voor ons om deze optie niet te kiezen.

Groot vs Klein

De opbouw van de verschillende frameworks onderling kan behoorlijk verschillen. JavaScript MVC en Ember (vooral de voorloper van de huidige versie) bieden out-of-the-box een paar prachtige mogelijkheden als Dependency Management, ingebouwde Testing Frameworks, etc. Andere frameworks als Knockout-js en Backbone-js zijn heel klein en modulair opgebouwd: je hebt mogelijkheden om plugins te gebruiken maar standaard is de opzet zeer eenvoudig. Het verschil in bestandsgrootte tussen bijvoorbeeld de JavaScript MVC library (31MB) en de Backbone-js library (52Kb) is dan ook duidelijk aanwezig. Omdat we niet zo'n grote hoeveelheid data heen en weer willen pompen tussen de server en de client, terwijl we waarschijnlijk maar een deel nodig hebben geniet de compacte aanpak hier de voorkeur. Knockout-js, Backbone-js en Mustache voldoen aan deze criteria.

Veel documentatie vs Weinig documentatie

Voor sommige frameworks is enorm veel documentatie te vinden op het internet. Helaas is dit niet voor ieder framework het geval: in het geval van Ember wordt bijvoorbeeld veel globaler over de mogelijkheden geschreven dan bij Backbone-js, waar redelijk diep op de materie wordt ingegaan. De mogelijkheid om informatie op te zoeken indien je even met een bepaald onderdeel in de knoop zit biedt natuurlijk voordelen.

Frontend + Backend vs Frontend

Waar sommige frameworks zowel frontend als backend functionaliteit bevatten (zoals het creëren van een objectmodel, wat je waarschijnlijk ook al in Java hebt gedaan) bevatten andere frameworks enkel de frontend mogelijkheden. JavaScript MVC en Ember pretenderen dat het nuttig is om aan de JavaScript-side kant een objectmodel op te bouwen dat gelijk is aan de Java-side, maar dit levert wel duplicatie met zich mee. Omdat we de frontend laag juist zo minimaal mogelijk willen houden zonder tussenkomende laag is de optie van gemengde frontend en back-end voor de JavaScript projecten niet wenselijk. Daarom genieten Knockout-js, Backbone-js, Bootstrap en Mustache hier de voorkeur.

8.6.3 Conclusie

De twee opties die het vaakst aan de positievere van de twee opties stonden waren Backbone-js en Knockout-js. Beiden beschikken over een aantal handige plugins. Met Knockout-js kun je aan binding op HTML markup of op basis van selectors doen, in Backbone-js alleen op de laatste. Dit is echter niet erg, omdat we toch selectors willen gebruiken.

Hoewel de stand in deze vergelijking gelijk is, lijkt er binnen het bedrijf het meeste draagvlak te zijn voor Backbone-js. Het feit dat van de beiden frameworks Backbone-js het meeste in de publiciteit staat op het moment en de meeste informatie hierover beschikwaar is was een belangrijke reden hiervoor.

8.7 Opzetten eerste demo-versie Mees

Gelijktijdig met de ontwikkeling van de code in Postduif die medewerkersbeschikbaarheid uitgerekend was al een kleine structuur voor de Mees applicatie opgebouwd. Enerzijds bood dit mogelijkheden om in een vroeg stadium nog meer frontend kennis op te bouwen, anderzijds maakte dit de weergave van het berekende aantal uren inzichtelijker.

Deze was aanvankelijk geschreven zonder framework als Backbone, omdat de beslissing welk framework het zou worden later pas genomen zou worden. Wel heb ik de code modulair geschreven, zodat het later gemakkelijk omgeschreven kon worden naar Backbone JS views en models. Het omschrijven van deze code heeft aanvankelijk een aantal dagen gekost, mede omdat de tests ook omgeschreven moesten worden, maar heeft in de langere termijn geresulteerd in beter gestructureerde en gemakkelijker onderhoudbare code.

Aanvankelijk werd besloten nog geen plaatjes met strepen van verschillende diktes te gebruiken voor de weergave van medewerkersbeschikbaarheid, maar dit te doen met getallen in cellen. De legenda waar de datum van de verschillende beschikbaarheidscellen mee kan worden afgelezen werd in de eerste versie eveneens nog niet opgenomen, in plaats daarvan kon de datum worden afgelezen door over het getal dat medewerkersbeschikbaarheid aangaf heen te bewegen met de muis. Dit is in een latere Sprint toegevoegd.

Filter namen	Mei			Juni																																		
	Week 22			Week 23							Week 24							Week 25							Week 26							Week 27							Week 28			
	Wo	Do	Vr	Ma	Di	Wo	Do	Vr	Ma	Di	Wo	Do	Vr	Ma	Di	Wo	Do	Vr	Ma	Di	Wo	Do	Vr	Ma	Di	Wo	Do	Vr	Ma	Di	Wo	Do	Vr	Ma	Di	Wo	Do	Vr
Ailbert Riksen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Stefan Rieken	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	0	8	8	8	8	0
Eric Meijer	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Marvin Koot	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Jolanda de Lange	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Gerrit Bes	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8
Willem Dekker	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Sebastiaan Wibier	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8
Theo van Essen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Michel Noppert	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Marcel Noordzij	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Luc Peerdeman	0	8	8	8	8	0	8	8	8	8	0	8	8	8	8	8	0	8	8	8	8	8	0	8	8	8	8	0	8	8	8	0	8	8	8	0	8	8
Erik Hooijmeijer	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8
Harko Ratsma	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	7	7	7	7	7	7	7	7	7
Stefan Köhler	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Michel Greve	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	0	8	8	8	8	0
Wiebe van Ginkel	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	0	8	8	8	8	8
Richard Brijde	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ron Smeets	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ron Oudshoorn	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Bas Biesheuvel	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8
Michel Hendriksen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8
Margret Kouwen-...	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Rud Mooij	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Gerard Visser	0	0	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Bas de Vos	8	8	0	8	8	8	8	0	8	8	8	8	0	8	8	8	8	0	8	8	8	8	0	8	8	8	8	0	8	8	8	8	0	8	8	8	8	0
Paulien de Wind	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bram Pramono	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8
Robert Bor	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	0	0	0	0	0	0
Mat Huizing	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	4	4	4	4	0	4	4	4
Jeroen van Schagen	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	0	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Johan Scholten	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Dave van Eijck	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ilona Bregard	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8
Oscar Westra van ...	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Michiel Nieuwstra...	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Sander Benschop	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	8
Wilco Stuyfersant	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Pim van Dongen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Deze concessies zijn gedaan zodat het algoritme zo snel mogelijk getest kon worden.

Zoals hiernaast gezien kan worden gaf dit echter wel al de mogelijkheid voor de product owner om een acceptatietest uit te voeren waarin hij wilde kijken of het algoritme dat ik had ontwikkeld tot dezelfde uitkomsten kwam als hij verwachtte. Voor een uiteindelijke versie oogt dit echter nog iets drukker dan wenselijk is.

Een beeld van hoe de weergave van beschikbaarheid uiteindelijk zal worden weergegeven kan worden ingezien in bijlage H: *"Grafische ontwerpen"*.

8.8 Acceptatietest

De test-omgeving van Mees was gemakkelijk up te daten via FTP waar de project source naar de juiste Apache folder kon worden geüpload. De Postduif-kant was een iets lastiger probleem. Met als doel het zo snel mogelijk afronden van de Sales View is aanvankelijk besloten om een ander onderdeel op de langere baan te schuiven: namelijk de integratie met het centrale authenticatiesysteem Crowd. De verwachting was dat hier dusdanig veel tijd in zou zitten dat het de voortgang van het project teveel zou tegenhouden, en om te kunnen voldoen aan een van de belangrijkste wensen van 42 BV aan mijn afstudeeropdracht: namelijk het ontdekken van impediments[1] bij het gebruik van een pure HTML/JavaScript applicatie met directe Java Backend heb ik besloten om dit inderdaad pas later in het project aan te passen.

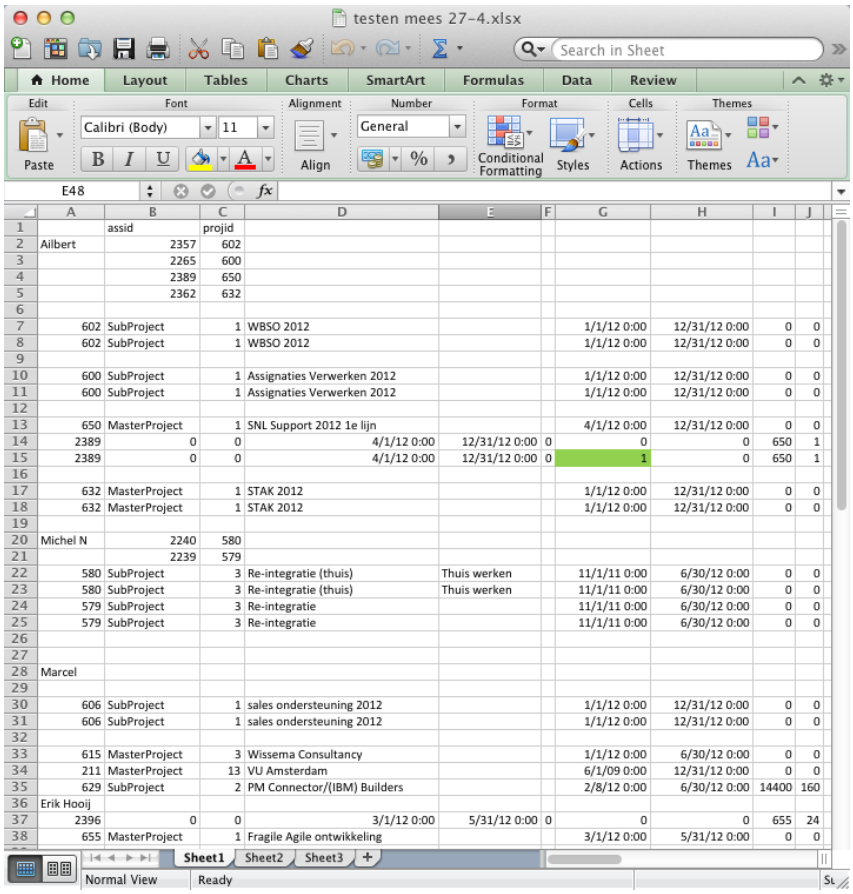
Het is niet wenselijk als de gegevens uit de testomgeving vrij beschikbaar worden op het internet, omdat deze afkomstig zijn uit de productiedatabase van Mus en het dus om echte data gaat. Daarom is besloten een kopie van de Postduif test-omgeving op te zetten en hier de Mees test-omgeving mee te laten communiceren.

Op beide servers werd vervolgens een ip-adresfilter gezet: enkel vanuit het kantoor van Zoetermeer en Amsterdam werd de data beschikbaar gemaakt. De kopie van de Postduif-server stond ook calls toe vanaf het ip-adres van de Mees-server. Op deze manier was er toch nog een vorm van beveiliging aanwezig en kon de product owner aan de slag met het uitvoeren van zijn acceptatietests.

Nadat de testomgeving live was wilde de product owner graag een test uitvoeren, waarin hij onder andere keek of het algoritme de beschikbare uren zodanig uitrekende als hij had bedoeld. Hij maakte als testset gebruik van de aanwezige data op de Mus-database van de testomgeving. Deze testset bevatte 39 medewerkers die nog in dienst waren, en bevatte alle typen assignaties, urenregistraties en iedere dataschending die ik in de datavalidatie heb gemarkeerd. Omdat deze data afkomstig is uit de productie-database en alle situaties erin voorkomen (foutieve data, reguliere assignaties, speciale assignaties, etc.) dient het als goede testset.

[1] Scrum terminologie voor “alles dat de voortgang tegenhoudt”

In een Excel-bestand hield hij voor verschillende situaties bij wat hij verwachtte terug te zien in de Sales View. De uitkomsten hiervan vergelijk hij vervolgens met elkaar en in geval van ongelijkheid noteerde hij dit.



	A	B	C	D	E	F	G	H	I	J
1		assid	projid							
2	Ailbert	2357	602							
3		2265	600							
4		2389	650							
5		2362	632							
6										
7	602	SubProject	1	WBSO 2012			1/1/12 0:00	12/31/12 0:00	0	0
8	602	SubProject	1	WBSO 2012			1/1/12 0:00	12/31/12 0:00	0	0
9										
10	600	SubProject	1	Assignaties Verwerken 2012			1/1/12 0:00	12/31/12 0:00	0	0
11	600	SubProject	1	Assignaties Verwerken 2012			1/1/12 0:00	12/31/12 0:00	0	0
12										
13	650	MasterProject	1	SNL Support 2012 1e lijn			4/1/12 0:00	12/31/12 0:00	0	0
14	2389	0	0	4/1/12 0:00	12/31/12 0:00	0	0	0	650	1
15	2389	0	0	4/1/12 0:00	12/31/12 0:00	0	1	0	650	1
16										
17	632	MasterProject	1	STAK 2012			1/1/12 0:00	12/31/12 0:00	0	0
18	632	MasterProject	1	STAK 2012			1/1/12 0:00	12/31/12 0:00	0	0
19										
20	Michel N	2240	580							
21		2239	579							
22	580	SubProject	3	Re-integratie (thuis)	Thuis werken		11/1/11 0:00	6/30/12 0:00	0	0
23	580	SubProject	3	Re-integratie (thuis)	Thuis werken		11/1/11 0:00	6/30/12 0:00	0	0
24	579	SubProject	3	Re-integratie			11/1/11 0:00	6/30/12 0:00	0	0
25	579	SubProject	3	Re-integratie			11/1/11 0:00	6/30/12 0:00	0	0
26										
27										
28	Marcel									
29										
30	606	SubProject	1	sales ondersteuning 2012			1/1/12 0:00	12/31/12 0:00	0	0
31	606	SubProject	1	sales ondersteuning 2012			1/1/12 0:00	12/31/12 0:00	0	0
32										
33	615	MasterProject	3	Wissema Consultancy			1/1/12 0:00	6/30/12 0:00	0	0
34	211	MasterProject	13	VU Amsterdam			6/1/09 0:00	12/31/12 0:00	0	0
35	629	SubProject	2	PM Connector/(IBM) Builders			2/8/12 0:00	6/30/12 0:00	14400	160
36	Erik Hooij									
37	2396	0	0	3/1/12 0:00	5/31/12 0:00	0	0	0	655	24
38	655	MasterProject	1	Fragile Agile ontwikkeling			3/1/12 0:00	5/31/12 0:00	0	0

Er zijn een paar uitzonderingen uit deze test gekomen, dit had te maken met onvoorziene situaties die uitzonderingen op de regels bleken te zijn die we tot die tijd als kloppend hadden beschouwd.

Een voorbeeld hiervan is een medewerker die in het medewerkersadviesorgaan werkt en hiervoor een bepaald aantal uur gereserveerd is. Het aantal werkuur en het aantal contracturen kwam hierdoor niet overeen, waarvan de product owner aanvankelijk dacht dat dit een schending van de datakwaliteit was. Dit bleek dus uiteindelijk niet het geval te zijn.

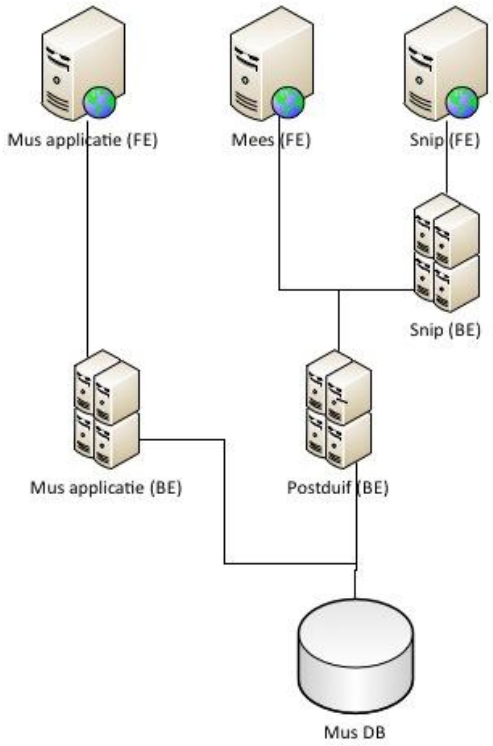
Dit proces heeft een aantal weken gelopen, omdat de product owner slechts twee dagen per week enkele uren beschikbaar was hiervoor. De wijzigingen zelf waren meestal niet van zeer grote omvang, dus konden ze snel worden opgelost. Voordat de product owner weer aanwezig was heb ik de nieuwe versie naar de testomgeving gedeployd[1] zodat hij er mee aan de slag kon gaan. Deze werkwijze was niet ideaal vanwege de grote vertragingen tussen de testmomenten, maar helaas was de situatie niet anders gezien het feit dat de product owner zelf bij een klant was ingezet.

[1] De geschreven software klaar maken voor gebruik en plaatsen op de juiste omgeving

8.9 Crowd Security

Na het deployen van de eerste demoversie van Mees met de bijbehorende versies van Postduif zonder authenticatie, ben ik me verder gaan oriënteren op de bestaande security met als doel het beveiligen van de Employee-controller en deze aanroepbaar te maken vanuit Mees.

Momenteel is er slechts een ander intern systeem van 42 BV, genaamd Snip, dat net als Mees via Postduif communiceert met de Mus database. Het verschil is echter dat Snip een Grails-applicatie is, net zoals de opzet die een andere medewerker ooit voor Mees had gemaakt maar die in zijn geheel is komen te vervallen, en Mees een pure HTML/JavaScript applicatie is.



Het verloop van het proces in Snip werkt als volgt: de gebruiker bezoekt de Snip frontend met de browser en vult een inlogformulier in met de benodigde informatie. Dit formulier wordt verzonden en opgevangen door de Snip backend. Deze praat vervolgens met een specifieke controller in Postduif die via een aparte module communiceert met Crowd om te valideren of een medewerker geautoriseerd is om Snip te gebruiken.

De communicatie tussen Snip en Postduif gebeurd met behulp van een "Authorization" header die aan Postduif wordt meegegeven, die een base64[1] versleutelde versie bevat van de username en password van de gebruiker. Indien de medewerker inderdaad geautoriseerd is wordt er een bericht naar Snip afgegeven dat de authenticatie geslaagd is. Snip's backend houdt in een sessie bij wat de username en password van de gebruiker is, en geeft de session identifier terug aan de Snip frontend die in een cookie wordt opgeslagen. Deze cookie wordt vervolgens bij iedere call teruggegeven aan de Snip backend, en op basis van de identifier worden de juiste username en password weer doorgegeven aan Postduif.

De security werd afgedwongen door in het web-configuratiebestand van Postduif aan te geven welke controllers protected zijn, vervolgens wordt dit in een losstaande JAAS-containerbeveiliging (Java Authentication and Authentication Service) afgedwongen en gecontroleerd met de Crowd-server. Zowel de authenticatie als autorisatie wordt nu feitelijk door Crowd uitgevoerd.

[1] Zeer basale encryptiemethode die door HTTP Basic gebruikt wordt

Ik heb echter tijdens mijn research op de werking van Crowd Security voor Postduif de volgende architectonische fout ontdekt: bij de ontwikkeling van Postduif is geen rekening gehouden met complexere autorisatiebehoeften, zoals authenticatie middels een Crowd-token.

Het probleem van de huidige aanpak is dat deze niet uitbreidbaar is naar Mees. Immers, het is niet wenselijk om een username en password in een vorm opslaan in je cookies die triviaal zijn om te ontsleutelen. Het is tevens niet mogelijk om vanuit Mees direct met Crowd te praten, omdat je bij iedere call behalve een username en password ook een applicatie-username en password mee moet geven. Dit mag absoluut niet in JavaScript code terecht komen, omdat iedereen het dan kan lezen.

Crowd ondersteunt het gebruik van security tokens om een gebruiker mee te authentiseren, deze moeten eenmalig worden aangevraagd bij de Crowd server. Hierbij worden de username en password van de gebruiker aangeleverd. Het grote verschil hierbij is dat de username en password niet onthouden hoeven te worden in de browser, maar in plaats daarvan een token waar deze gevoelige gegevens niet uit gehaald kunnen worden. De token heeft tevens een verlooptijd, dus mocht deze al uitlekken dan is dit een stuk minder groot probleem dan als dit met een username en password zou gebeuren.

Probleem is echter dat de huidige containerbeveiliging niet uitbreidbaar is om deze methode te dienen, mede vanwege het feit dat Snip al tegen Postduif praat via de JAAS container security, die niet uitbreidbaar is om deze tevens aan te spreken met een user token. Ook het aanvragen van een token bij Crowd behoort in dat geval niet tot de mogelijkheden.

Ik heb gezocht naar alternatieve oplossingen, maar het is mij duidelijk geworden dat de enige optie die eigenlijk overblijft het implementeren is van een rijkere authenticatiemethode met behulp van een security framework. Spring Security is hier erg geschikt voor. Momenteel maakt 42 BV voor Postduif al gebruik van de andere onderdelen van het Spring Framework, dus dit sluit erg goed bij elkaar aan.

Dit framework is vele malen flexibeler dan de huidige implementatie en biedt mogelijkheden om applicaties als Mees te beveiligen zonder een username/password-combinatie heen en weer te sturen bij iedere call. Dit hoeft maar eenmaal te gebeuren om een token aan te vragen, daarna wordt deze ervoor in de plaats gestuurd.

Wat voor 42 BV een van de belangrijkste onderdelen was van mijn afstudeeropdracht was bekendwording met een aanpak waarbij gebruik wordt gemaakt van pure JavaScript/HTML applicaties die niet over een eigen backend beschikken maar dit via een centrale applicatie doen (in dit geval Postduif).

Toen ik deze bottleneck had ontdekt en de verschillende opties en afwegingen duidelijk had gemaakt aan het bedrijf was het zowel voor mijn eigen afstudeeropdracht, de afstudeeropdracht van een andere afstudeerder die binnenkort aan zijn opdracht begint en ander projectteam van 42 BV cruciaal dat dit probleem werd opgelost.

Zonder een oplossing voor dit probleem kunnen geen van deze applicaties in productie worden genomen. 42 heeft liever een wat meer basale frontend die nu al wel voldoende informatie toont om sales-beslissingen mee te maken maar die nog geen "uitklapinformatie" weergeeft dan een applicatie die dit wel doet en niet in productie genomen kan worden.

Dit betekende ook dat er niet meer begonnen kon worden aan een opzet van de Boekhouder view. Omdat de structuur van de Sales view en die van de beoogde Boekhouder view erg op elkaar lijkt, had ik hier waarschijnlijk een heel stuk van kunnen afronden zonder de security bottleneck. De Sales view is echter de belangrijkste omdat deze het primaire doel van Mees verzorgd: namelijk het inzichtelijk maken van medewerkersbeschikbaarheid zodat er geen collega's "op de bank" komen te zitten wanneer zij ook betaalde projecten hadden kunnen doen.

Net als de Sales view zou de Boekhouder view niet in productie kunnen voor deze security bottleneck was opgelost. Het implementeren van de security had dus de prioriteit boven alle andere taken en requirements op dat moment.

Deze oplossing omvat vooral wijzigingen aan de Postduif kant, en verder moet er een login-functionaliteit worden gebouwd voor Mees en moet de inlogmethode van Snip enigszins herzien worden.

Daarom is in overleg met de product owner, Chief Technology Officer Robert Bor en CEO besproken dat dit hoge prioriteit heeft. Er is een team van twee collega's beschikbaar gesteld om mij te helpen met deze omvangrijke taak. Zij hebben het volledige Snip-gedeelte van de mutatie op zich genomen omdat dit buiten de scope van mijn opdracht valt. Gezamenlijk hebben we de wijzigingen aan Postduif gedaan in een Sprint van het moment dat de CEO toestemming gaf voor de vorming van dit tijdelijke projectteam tot de week erna.

Dit heeft geresulteerd in een nettere en veilige oplossing voor authenticatie met Postduif. Mees en Snip zijn aangepast om van deze nieuwe vorm van beveiliging gebruik te maken, en de nieuwe versies zijn gedeployed. Wanneer deze binnenkort in productie gaan kan er gebruik worden gemaakt van de Sales view van Mees.

Uiteindelijk is de Crowd-oplossing die we hebben geïmplementeerd in een los jar[1]-bestand gezet, zodat projecten in de toekomst die niet via Postduif communiceren maar direct met Crowd moeten praten deze oplossing gemakkelijk kunnen overnemen.

[1] Java ARchive, wordt gebruikt om Java-code te archiveren

9 Kwaliteitsbewaring

9.1 Design Principles

Omdat 42 BV en ikzelf het zeer belangrijk vinden om code te schrijven die zo net mogelijk is heb ik vanaf het begin dit als centraal uitgangspunt van de ontwikkeling van Mees en Postduif beschouwd.

Om te bepalen wat nette code is hou ik de regels aan die worden beschreven in Robert C. Martin's "Clean Code"[6]. Dit boek bevat vele richtlijnen voor een moderne developer en geldt binnen 42 BV als een standaard voor kwalitatief software development. Enkele richtlijnen die hierin terugkomen zijn onder andere het gebruik van duidelijke naamgeving en het gebruik van kleine functies en klassen met zo min mogelijk verantwoordelijkheden. Door regels als deze aan te houden blijft de code beter leesbaar en is deze derhalve ook makkelijker onderhoudbaar door andere developers.

Ik heb me tijdens de oriëntatiefase van mijn afstuderen al enigszins ingelezen over de theorie van Dependency Injection en Mocking. Hier heb ik wederom naar gekeken tijdens de opstart van de programmeerfase, ditmaal op programmacode-niveau. Door mezelf te oriënteren op voorbeelden uit de bestaande codebase van Postduif. Toen ik deze techniek beter begon te beheersen ben ik in staat geraakt om nettere code te schrijven voor in het project.

Ik heb regelmatig gekeken of ik sommige stukken code netter kon schrijven dan ik op dat moment gedaan had. Om een voorbeeld te noemen: toen de product owner zich bedacht dat er nog meer combinaties van databasewaarden waren die hij graag gevalideerd wilde zien dreigde de code die oorspronkelijk een paar zaken controleerde van te grote omvang te worden om opgenomen te houden in de klasse waar de beschikbaarheid werd berekend. Hier wilde ik dus iets aan doen.

Daarom heb ik besloten de validatie te splitsen in een aparte klasse die ik implementeer vanaf een generieke EmployeeAvailabilityValidationInterface. Door naar interfaces te schrijven in plaats van implementaties behoudt je een losse koppeling, wat erg prettig is vanwege het gemak om een interface een andere implementatie te geven. Zo zou er bijvoorbeeld een situatie kunnen ontstaan in de toekomst waarin in een bepaald geval andere validatieregels gelden dan in het huidige geval. Op deze manier is dat zeer gemakkelijk, er hoeft slechts een andere implementatie te worden gebruikt.

Mocking is een manier om je unit tests compact en op zichzelf staand te houden, aanroepen naar andere klassen kunnen worden nagebootst waarin een waarde kan worden opgegeven die de "mock" teruggeeft. Op deze manier beperk je je unit-test tot een enkele functie die je wilt testen, zoals de bedoeling is van deze testsoort. Dit houdt de tests overzichtelijker, wat tevens gemakkelijker is wanneer andere programmeurs iets aan deze tests moeten aanpassen.

9.2 Unit testing

De code die ik in Postduif heb geschreven heb ik getest met het JUnit framework, wat de de-facto standaard is voor het testen van Java code. In tegenstelling tot bij Java is er in JavaScript geen dergelijke standaard.

Er zijn een legio test-frameworks met allemaal hun eigen voor- en nadelen. Tijdens de ontwikkeling van Mees heb ik naar enkele van deze frameworks gekeken met als doel het uitkiezen van de meest geschikte, die nu voor Mees en in de toekomst voor andere projecten gebruikt kan worden. In dit hoofdstuk behandel ik de uitkomsten van dit onderzoek.

9.2.1 JSTestDriver

Het eerste framework waar ik naar heb gegeken was JSTestDriver. Het biedt support om met meerdere browsers simultaan te testen en bevat tevens een Coverage plugin. Code coverage is het percentage van je code dat wordt uitgevoerd in minstens één unit-test, oftewel het percentage code dat door tests wordt “geraakt”. De syntax van de taal komt bijna exact overeen met die van JUnit, wat voor Java-programmeurs erg gemakkelijk is.

Ik heb in een testproject een simpele functie aangemaakt die ik wilde testen. Nadat ik de benodigde configuratie had uitgevoerd heb ik een testje geschreven die ik wilde runnen. Er zit echter een bug in JSTestDriver die ervoor zorgde dat de path naar het bestand dat ik wilde testen niet klopte. Na enig onderzoek bleek dit een bekende bug te zijn, die in een eerdere versie is veroorzaakt en schijnbaar wederom is teruggekomen. Een historisch archief was noch voor Eclipse noch voor IntelliJ IDEA beschikbaar, dit zijn de twee ontwikkelomgevingen zijn die gebruikt worden binnen 42 BV.

Los van het feit dat de tests überhaupt niet uitgevoerd werden, vereist JSTestDriver één of meerdere browsers die op dezelfde machine geïnstalleerd zijn. Dit is in je IDE natuurlijk niet zo'n probleem, maar op het moment dat we de tests ook geautomatiseerd op de command-line server van Bamboo uit willen laten voeren kan het gebrek aan grafische operating system dit toch bemoeilijken. Het starten van een browser op een omgeving als deze is standaard niet mogelijk, we zouden hiervoor gebruik moeten maken van een techniek als een virtual frame buffer, die nog ingesteld zou moeten worden op de server.

9.2.2 Jasmine

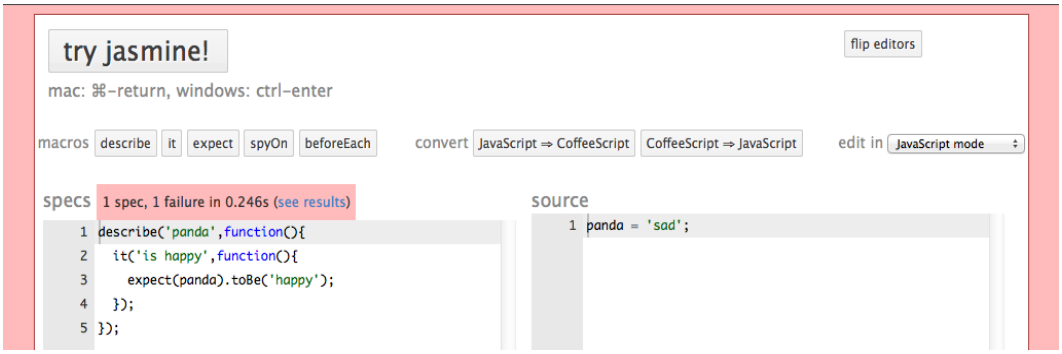
De Chief Technology Officer heeft mij gewezen op een ander testing framework: Jasmine. Deze werkt iets anders dan de andere frameworks: ze noemen het zelf een behaviour driven development testing framework. Er is een plugin voor de build automatization tool Maven beschikbaar die het uitvoeren van alle tests in een bepaalde folder kan automatiseren. Ook biedt deze plugin de mogelijkheid om een HTTP server te starten waarop de resultaten bekeken kunnen worden.

Behaviour driven development is een uitbreiding op het test driven development paradigma met een extra focus op leesbaarheid. In principe moeten ook niet-programmeurs deze code kunnen begrijpen. Hieronder is een voorbeeld te zien dat afkomstig is van hun eigen website.

De parameter 'Panda' op de describe-functie is de naam van de Jasmine-test, dit is Engels en geen code. De inhoud van de test controleert of de emotional condition 'happy' is. Zo niet, dan licht deze test rood op in het overzicht.

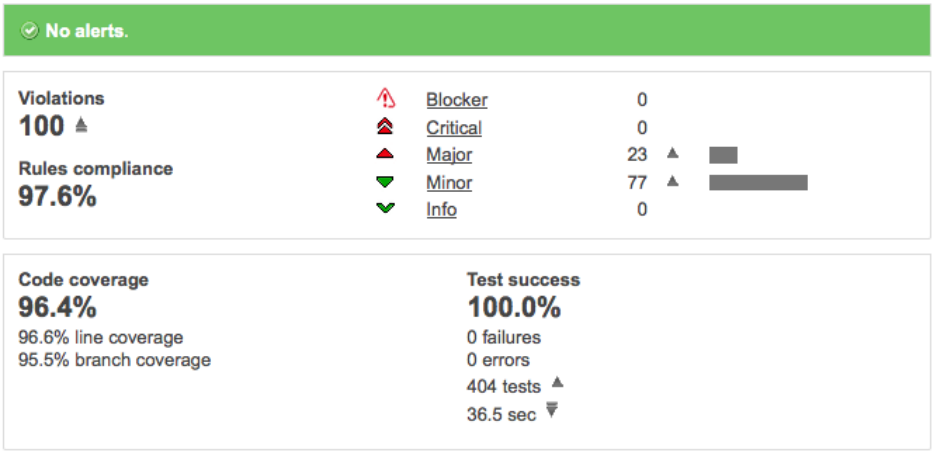
```
describe('Panda',function(){
  it('is happy',function(){
    expect(panda.emotionalCondition).toBe('happy');
  });
});
```

Deze duidelijke testmethode die tevens mogelijkheden biedt voor integratie met de bestaande ontwikkelstraat van 42 sprak mij erg aan, daarom heb ik besloten om dit testing framework te gebruiken voor het testen van mijn JavaScript-code.



[7]

9.3 Sonar



Binnen 42 BV wordt gebruik gemaakt van de applicatie Sonar om inzicht te verkrijgen over de kwaliteit van de geschreven software. Deze tool krijgt de gecompileerde code van het project aangeleverd van Bamboo en gebruikt deze data om diverse "metrics" te bepalen. Een voorbeeld hiervan is bijvoorbeeld het percentage geteste code of de complexiteit van een functie. Die laatste meetwaarde wordt bepaald aan de hand van het aantal statements per functie: hoe meer, hoe complexer.

Bij aanvang van het project is in een Definition of Done vastgesteld wat de waarden van deze meetwaarden moesten inhouden. Deze gegevens zijn gebaseerd op gezamelijke ervaring met deze meetwaarden van mijzelf en de product owner.

Een user story kon pas als "done" worden gezien als naast de functionele eisen eraan deze gegevens ook op het gewenste niveau waren. Op deze manier werd de kwaliteit van de code gedurende het project consequent op niveau gehouden, wat het verder ontwikkelen door mijzelf of een andere programmeur vergemakkelijkt. De volledige Definition of Done kan worden gevonden onder bijlage M: *"Definition of Done".*

10 Evaluatie

10.1 Procesevaluatie

Het proces van de afstudeeropdracht is naar mijn goede tevredenheid verlopen. Van tevoren was mij verteld dat een aanpak zoals de mijne, een pure HTML/JavaScript applicatie bouwen die zonder eigen backend communiceert met een centraal systeem als Postduif nog nooit binnen het bedrijf was uitgevoerd en dat een van de belangrijkste zaken van mijn afstudeeropdracht het te weten komen van allerlei impediments was.

Meteen in het begin van de afstudeeropdracht ben ik hier al tegenaan gelopen toen de Same Origin Principle de communicatie tussen de Mees en Postduif server belemmerde. Ik heb een onderzoek gedaan naar de oplossingen, de beste voorgesteld aan de architectonische beslissingsnemers en de oplossing gepresenteerd aan alle collega's. Hierdoor kunnen zij bij een volgend project dat op dezelfde wijze wordt opgepakt direct de oplossing implementeren en lopen ze dus niet tegen dezelfde problemen aan.

Deze impediment heeft inclusief het onderzoek, overleg, implementatie en presentatie een hoop tijd gekost. Dit was echter ruimschoots de moeite waard als ik me bedenk dat andere collega's deze oplossing direct mee kunnen nemen in nieuwe projecten, dat we nu weten nu dit moet en waarom dit de beste oplossing voor het probleem is.

Een andere grote impediment was de huidige implementatie van security. Bij het ontwerpen van Postduif waren hier architectonische beslissingen gemaakt die uitbreiding voor projecten als deze onmogelijk maakten. Ik heb dit tijdens mij onderzoek naar het huidige security mechanisme aangetoond, bewezen en aangegeven bij de architectonische beslissingsnemers van 42.

Toen deze nieuwe vorm van security was ingebouwd konden zowel Mees als Snip op een veilige manier communiceren met Postduif en was de mogelijkheid voor andere projecten die in de toekomst met Postduif gaan praten ook direct voltooid, of dit nu een soortgelijk project als Mees zou worden, een applicatie met eigen backend of een mobiele applicatie (zoals een andere afstudeerder momenteel bouwt) wordt.

Wederom heb ik hier dus gekozen voor een lange-termijnsoplossing die direct toepasbaar is voor andere collega's. Ik sta achteraf nog steeds achter de beslissingen die ik toen heb genomen, ondanks het feit dat dit het aantal afgeronde requirements van Mees zelf heeft verminderd.

Ik ben tevreden over het gebruik van Scrum tijdens dit project. Ik heb hier al eerder het een en ander mee gedaan, maar ditmaal was mijn afstudeerbegeleider degene die 42 BV met de ontwikkelmethodiek heeft geïntroduceerd. Hij had dus een hoop kennis over de processen die hierbij hoorde, waardoor ik sommige aspecten ervan beter heb leren begrijpen.

Zo is het bijvoorbeeld gebruikelijk om bij iedere sprint planning per user story te bespreken hoe complex alle betrokkenen denken dat deze story is. Dit gebeurd bij 42 door een potje “planningpoker” te spelen. Per user story wordt besproken wat deze precies betekent waarna iedereen een kaart gesloten neerlegt. Als deze worden omgedraaid en het aantal toegekende punten onderling verschilden moet degene met de minste punten overleggen met de persoon met de meeste punten hoe ze tot hun keuze zijn gekomen.

Soms treedt dit verschil op doordat iemand een punt onderschat of juist overschat, en soms komen er subtaken naar voren waar iemand überhaupt nog niet aan gedacht had. Ik vind dit een handig manier om duidelijkheid over alle betrokken te creëren over wat een user story precies inhoudt.

Ik heb echter altijd gedacht dat een aantal punten gelijk stond aan een bepaalde tijdseenheid, en dat dit als een soort contract geldt voor de komende sprint. Tot op zekere hoogte is dit waar, iedere sprint wordt aan de hand van het aantal punten gekeken hoeveel punten er de volgende beurt worden toegekend. De waarde van een aantal punten is echter niet in steen gebeiteld. Dit is meer een hulpmiddel om te kunnen ontdekken of iedereen hetzelfde beeld had bij een user story. Toen ik dit besef kreeg was de toegevoegde waarde door mij makkelijker om in te zien.

De “Definition of done” die ik van tevoren heb opgesteld maakte het gemakkelijker om te kunnen bepalen wanneer een punt voltooid was. Een user story werd dan ook pas als afgerond gezien als aan deze eisen was voldaan. Dit kostte iets meer tijd, maar zorgde wel voor code van betere kwaliteit dus was dit zeker de moeite waard.

Het wekelijkse overleg met de product owner tijdens de retrospective en het plannen van de nieuwe sprint was in principe voldoende, echter was er helaas weinig tijd beschikbaar voor hem om acceptatietests uit te voeren. De keren dat dit wel gelukt was heeft hij enkel uitkomsten gegeven die ik snel en goed kon verwerken, maar de tussenliggende tijd vond ik meestal iets te lang. De product owner was echter zelf ook ingezet op een project dus hier was helaas niets aan te doen.

Bij deze afstudeeropdracht waren diverse stakeholders betrokken, waaronder bijvoorbeeld de vertegenwoordiger van de Sales-afdeling. Tijdens het ontwikkelen is het voorgekomen dat er tegenstrijdige visies tussen de stakeholders merkbaar waren over de oplossing van een bepaald probleem. Ik heb hierbij naar allen geluisterd, maar de wensen van de daadwerkelijke gebruiker, de Sales-vertegenwoordiger, nogmaals aangegeven bij de product owner. Uiteindelijk zijn deze oplossingen ook meegenomen in het eindresultaat.

De reden dat ik dit op deze manier heb uitgevoerd was het feit dat ik wilde dat de gebruiker een product in handen zou krijgen waarvan hij vond dat het aansloot op zijn manier van werken. Ik ben tevreden dat ik het probleem op deze manier heb opgelost, iedereen heeft goede ideeën geleverd voor dit project, maar uiteindelijk is een tevreden gebruiker natuurlijk het belangrijkste. Daar waren alle betrokkenen het overigens mee eens.

10.2 Productevaluatie

Ik ben tevreden hoe het eindresultaat eruit ziet, ik heb in een programmeertaal die compleet nieuw voor me was een product neergezet waar de Sales-afdeling binnenkort mee aan de slag kan en hopelijk tot het gewenste doel gaat leiden, namelijk het voorkomen dat medewerkers door planningsfouten “op de bank” komen te zitten.

Ik vind het wel jammer dat ik niet meer ben toegekomen aan de Boekhouder view, omdat ik dan nog een stakeholder tevreden had kunnen maken. De opgetreden impediments hebben dit helaas verhinderd. Wel is de basis nu gelegd om deze view gemakkelijk toe te kunnen voegen, omdat alle ontdekte impediments reeds uit de weg zijn. Ik denk achteraf nog steeds dat ik de goede beslissingen heb gemaakt in deze verschuiving van werkzaamheden.

Ik ben ook erg blij dat ik de mogelijkheid heb gehad om onderzoek te doen naar zaken als MVC-frameworks voor JavaScript, testing-frameworks voor JavaScript en Sonar-integratie. Dit biedt wederom kwalitatieve meerwaarde voor mijn eigen project en dat van anderen in de toekomst.

Ook op Java-gebied heb ik een hoop geleerd: waar ik hiervoor wel eens gehoord had van technieken als Dependency Injection of Mocking ben ik er deze keer goed ingedoken om de essentie te kunnen begrijpen. Dit heeft nettere code als gevolg gehad.

Ook het feit dat ik een oplossing heb mogen presenteren voor het architectonische beveiligingsprobleem, deze werd geaccepteerd en vervolgens ondere andere door mij is gebouwd doet me veel deugd. Niet alleen heb ik hierdoor een hoop geleerd over security frameworks en alle bijbehorende zaken, het was voor mij ook erg fijn om te kunnen concluderen dat ik een goed architectonisch voorstel had gemaakt waar de organisatie erg blij mee is.

Naar mijn mening heb ik een hoop geleerd over zowel het uitvoeren van vooronderzoek naar technieken als het afwegen hiervan en het presenteren van de oplossing die ik het meest passend vind aan een organisatie. Verder heb ik met een breed spectrum van technieken leren werken waar ik tot nog toe geen ervaring mee had.

10.3 Beroepstaken

10.3.1 Selecteren methoden, technieken en tools

Selecteren en adviseren over ontwikkeltools die een onderdeel (gaan) vormen van de ontwikkel-, beheer-, test- en productieomgeving.

Ik heb deze competentie behaald door het selecteren van een JavaScript MVC-framework, een JavaScript testingframework en een Java Security Framework. Hierbij heb ik de bestaande architectuur en ontwikkelstraat van 42 BV constant in gedachten gehouden, zodat ik wist dat projecten die ontwikkeld waren met deze frameworks hier goed in zouden passen.

10.3.2 Uitvoeren analyse door definitie van requirements

Inventariseren, analyseren en beschrijven van de gewenste informatievoorziening, zodat wordt vastgelegd welke processen objecten creëren en gebruiken.

Ik heb interviews gehouden met de verschillende stakeholders waarbij ik de functionaliteiten die ze uiteindelijk in de applicatie wensen te zien in kaart heb gebracht. Hier heb ik in een SMART-tabel gezet waarbij ik diverse vragen heb moeten beantwoorden om de essentie van de requirement te kunnen doorgronden. Het opstellen van requirements in een gestructureerde vorm was de eerste keer voor mij en daarom ben ik van mening dat ik deze competentie heb behaald.

10.3.3 Ontwerpen systeemdeel

Beschrijven van systeemdelen (subsystemen, componenten, modules), hun onderliggende structuur en het gedrag in detail, zodanig dat bouwen van het systeemdeel mogelijk is.

Ik heb met behulp van verschillende UML diagrammen aan mijzelf en anderen duidelijk kunnen maken hoe bepaalde onderdelen van het systeem en bepaalde gecompliceerde processen er uit kwamen te zien. Ik heb diagrammen gemaakt daar waar ik het nuttig achtte voor mijn eigen begrip of communicatie over het project met anderen. Hierdoor heb ik naar mijn mening deze competentie behaald.

10.3.4 Bouwen applicatie

Bouwen en documenteren van de systeemdelen.

Ik heb deze competentie behaald door te programmeren in verschillende talen en met frameworks waar ik nog nooit iets mee gedaan had. Het uitbreiden van een bestaande backend was een goede stimulans om de codebase op hetzelfde niveau te houden, wat heeft geresulteerd in het bekend worden met complexere programmeertechnieken als Dependency Injection en Mocking.

10.3.5 Uitvoeren van en rapporteren over het testproces

Uitvoeren testplan en opstellen rapportage

Ik heb deze competentie behaald door het uitvoeren van diverse soorten tests tijdens de afstudeeropdracht. Zo heb ik meerdere keren prototypes geschreven om te testen of bepaalde oplossingsrichtingen die ik in gedachten had wenselijk waren, en heb ik zowel voor JavaScript als voor Java testen hoog in de prioriteit gezet. De uitkomsten van de prototypes zijn vastgelegd op Confluence, zodat andere medewerkers hier in de toekomst informatie uit kunnen halen. De resultaten van de unit tests kunnen constant worden ingezien op Sonar.

11 Begrippenlijst

Woord	Betekenis
Ajax	“Asynchronous JavaScript en XML”: biedt mogelijkheden om nadat een pagina geladen is calls uit te voeren naar de server.
Backend	Deel van de applicatie dat onzichtbaar is voor de gebruiker, het handelt de requests van de frontend af. In dit geval is de backend Postduif.
Backlog	Verzameling van alle user stories die bij een Scrum-project horen.
Bamboo	Continuous integration server van Atlassian, wordt binnen 42 gebruikt om projecten die op SVN worden ingecheckt te builden en meetwaarden naar Sonar te sturen.
Base64	Zeer basale encryptiemethode die door HTTP Basic gebruikt wordt in de vorm “username:password”. Is triviaal om te decrypten, dus zou nooit over een onbeveiligde lijn gebruikt mogen worden.
Confluence	Corporate wikipedia van Atlassian, wordt binnen 42 gebruikt als tool om kennis mee te delen.
Controller	In de context van een webapplicatie: klasse waar REST-calls tegenaan uitgevoerd kunnen worden en deze requests doorgeven aan een klasse op de servicelaag
Database-access-object	Klasse die verantwoordelijk is voor de communicatie met de database.
Deployen	De geschreven software klaar maken voor gebruik en plaatsen op de juiste omgeving
Dynamische taal	Taal waarin veel gedrag runtime wordt uitgevoerd in plaats van compile time, zoals bijvoorbeeld uitbreiding van het programma met extra code of het extenden van objecten.
Fisheye view	View waarin op bepaalde items en de omliggende items die hierbij horen kan worden ingezoomd, wat een bepaald vissenkomperspectief creëert.
Frontend	Deel van de applicatie dat zichtbaar is voor de gebruiker en mogelijkheden biedt om requests uit te voeren naar de backend.
Gantt-chart	Grafieksoort die wordt gebruikt om planningen mee te maken.
Grails	"Groovy on Grails”: een webframework voor de Groovy-taal.
HTTP-header	Componenten van de message die een HTTP request of response bevat, ze definieren de parameters van een HTTP-transactie.
Impediment	Een term die in Scrum wordt gebruikt om iets aan te geven dat de productiviteit tegenhoudt.
Increment	Een toevoeging, in deze context een toevoeging van functionaliteit aan het grotere geheel.
Iteratie	Een herhaling, in deze context het itereren door elementen in een datareeks.
Jar	Java ARchive, wordt gebruikt om Java-code te archiveren.
jQuery-selector	Methode om een HTML-element te selecteren binnen

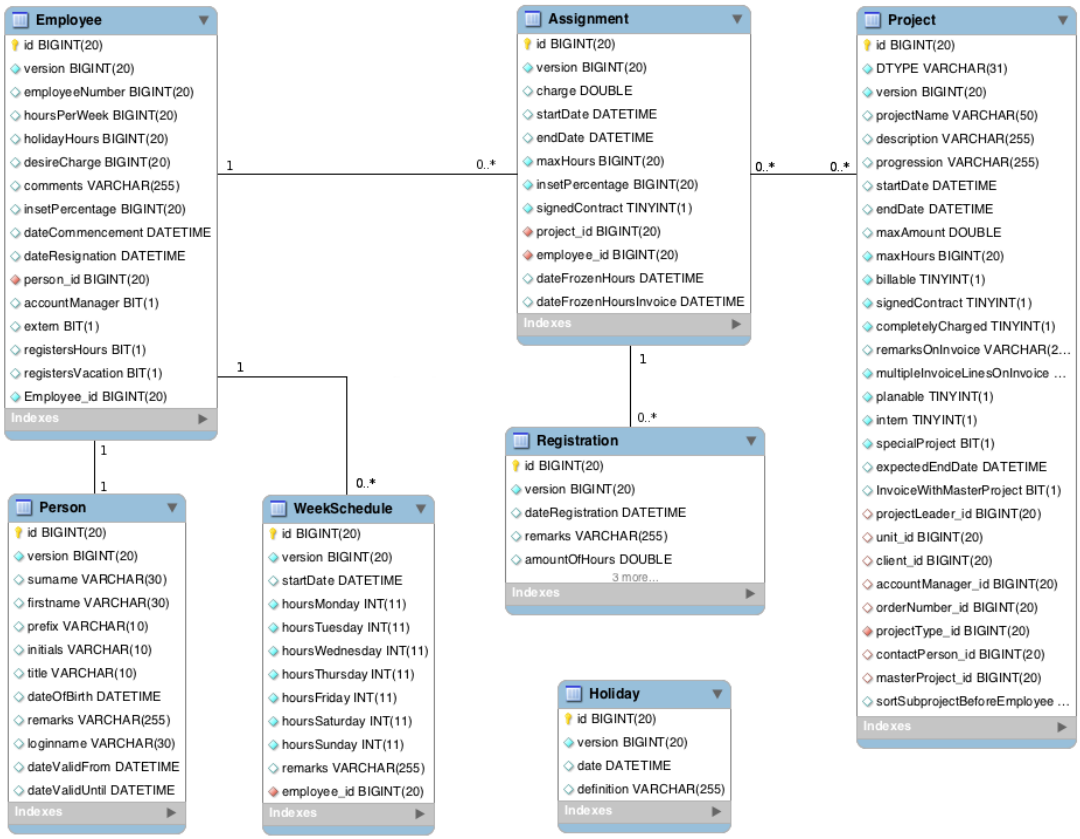
	JavaScript code met de mogelijkheid hier opdrachten op uit te voeren.
JSON	JavaScript Object Notation is een dataformaat dat onder andere veel wordt gebruikt als teruggaveformaat van REST-interfaces
REST	Representational State Transfer design model dat gebruikt kan worden om data op te vragen bij een server.
Retrospective	Meeting na afloop van een sprint waarin de afgelopen Sprint wordt geëvalueerd.
Serviceklasse	Klasse die verantwoordelijk is voor de business logica van een applicatie. De servicelaag bevindt zich typisch tussen de controllerlaag en de DAO-laag
Sonar	Tool om de kwaliteit van software duidelijk te maken aan de hand van meetwaarden als complexiteit en percentage geteste code.
TreeMap	Geordende implementatie van Java's Map interface
User story	Definitie van een requirement die de benodigde informatie voor developers bevat om deze op te kunnen pakken.
W3C	Word Wide Web Consortium, een organisatie die standaarden voor het web ontwikkelt.
Webapplicatie	Applicatie die vanaf het internet benaderd en gebruikt kan worden.
Webinterface	Frontend van een webapplicatie

12 Referenties

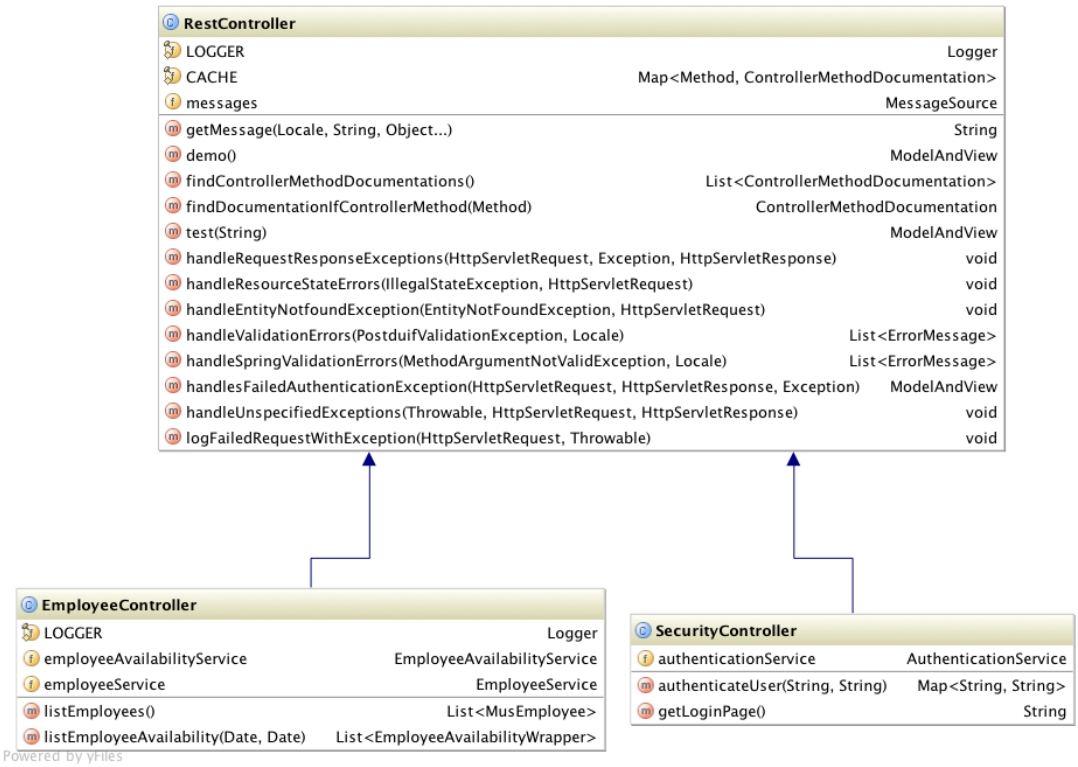
1. SMART criteria - http://en.wikipedia.org/wiki/SMART_criteria
2. Same Origin Policy W3C - http://www.w3.org/Security/wiki/Same_Origin_Policy
3. Same Origin Policy tabel - http://en.wikipedia.org/wiki/Same_origin_policy
4. JSON with Padding - http://en.wikipedia.org/wiki/JSONP
5. Proxy - http://developer.yahoo.com/javascript/howto-proxy.html
6. Robert C. Martin's "Clean Code" - http://www.amazon.com/Clean-Code-Handbook-Software-Craftsmanship/dp/0132350882
7. Try Jasmine Online – http://tryjasmine.com

13 Bijlagen

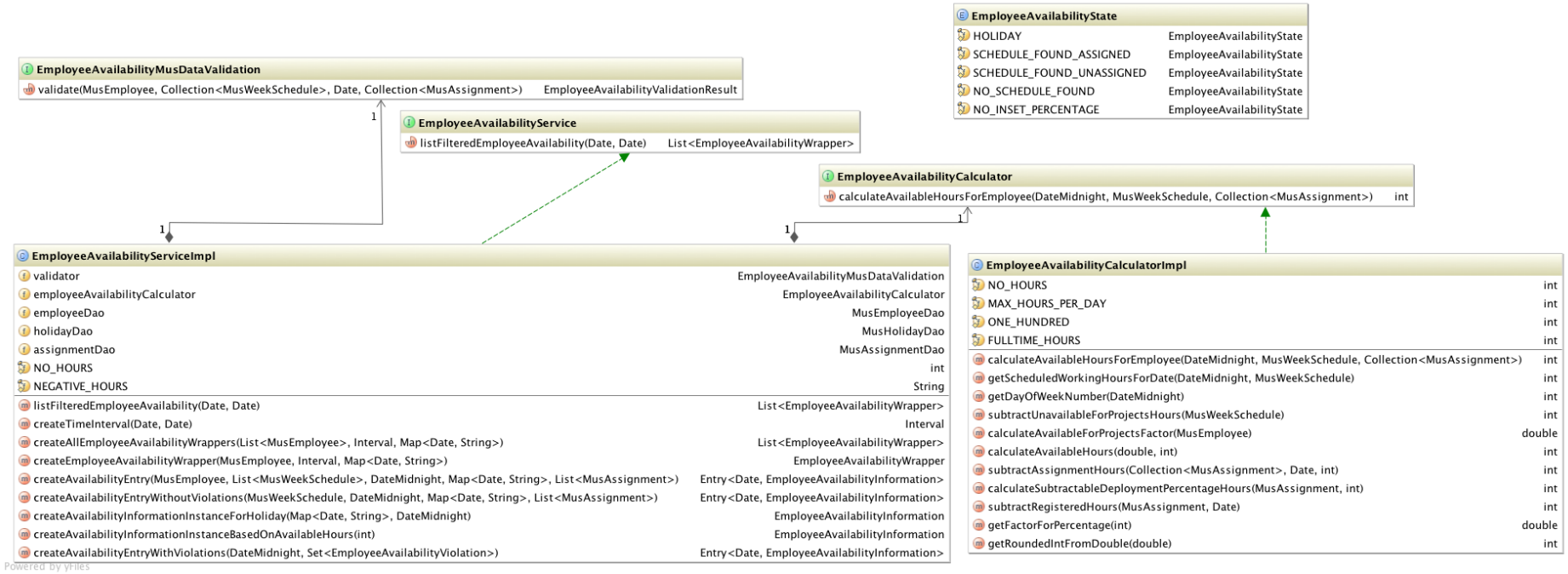
A. Databasemodel Mus



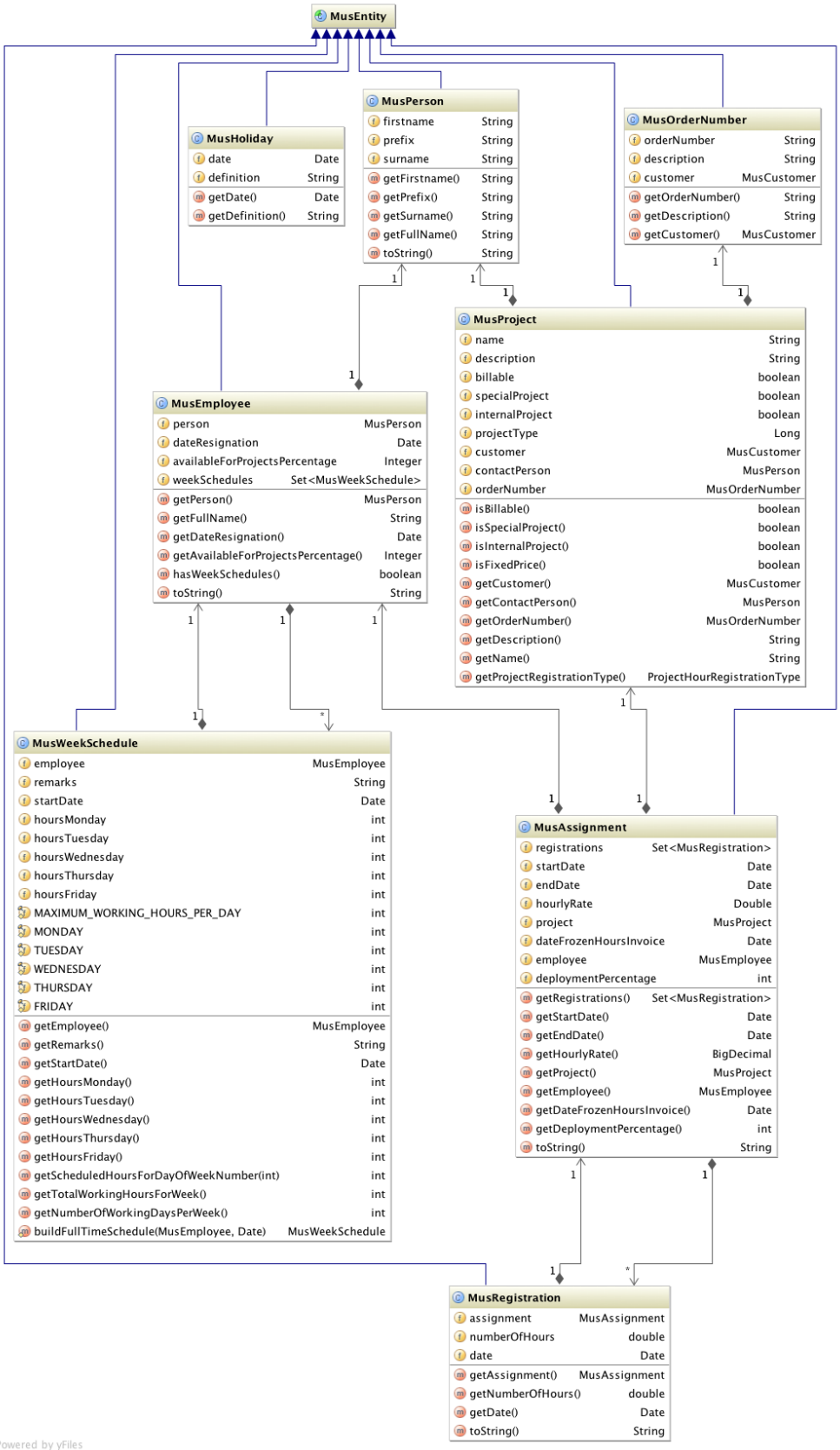
B. Klassediagram controller-laag



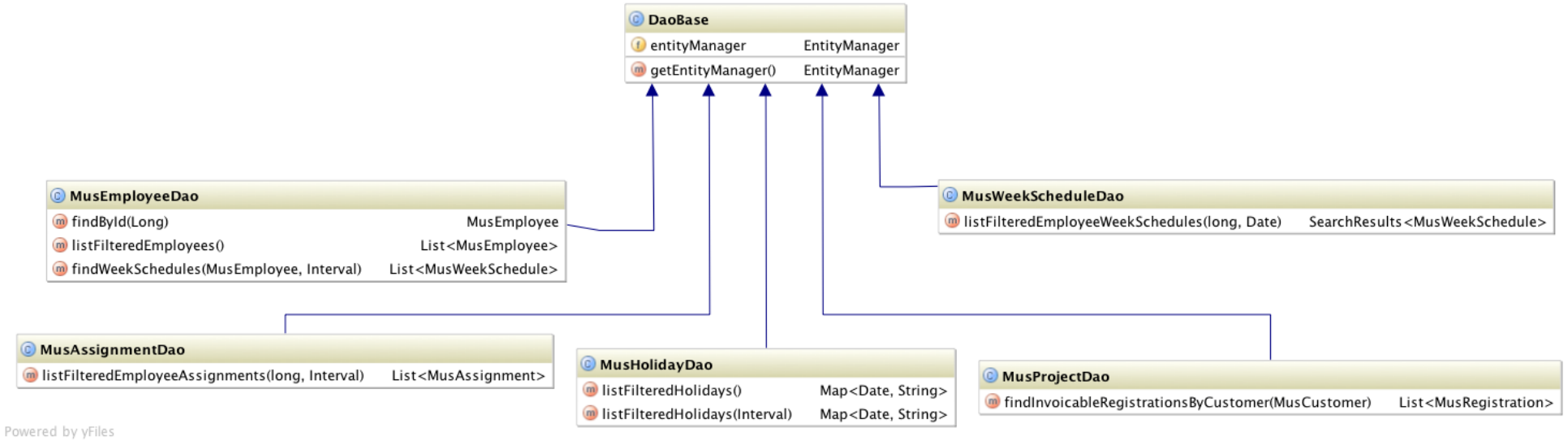
C. Klassediagram availability-servicelaag



D. Klassediagram domeinlaag



E. Klassediagram data-access-object laag



F. Sales requirements

Sorteer medewerkers op beschikbaarheid	
Specifiek	- In het overzicht van medewerkersbeschikbaarheid moeten medewerkers worden gesorteerd op basis van hoge beschikbaarheid in de komende maand naar lagere beschikbaarheid. Uitzondering hierop zijn medewerkers waarbij tijdens validatie van hun gegevens data is aangetroffen (of juist afwezig was) die benodigd was om de berekening uit te voeren, deze verschijnen bovenaan in het overzicht. Het doel hiervan is het prikkelen van de verbetering van de datakwaliteit.
Meetbaar	- Voor dit punt is het gemakkelijk om te controleren wanneer de user story voltooid is. Ik kan zelf controleren of de sortering klopt door de pagina weer te geven, alsmede het schrijven van unit-tests hiervoor aan de serverkant.
Acceptabel	- Het doel kan worden bereikt door een sorteeralgoritme te schrijven dat rekening houdt met de wensen die hierboven geschreven worden. De data kan server-side al gesorteerd worden en vervolgens worden doorgegeven aan de client. Reden hiervoor is het feit dat Java van zichzelf al beschikt over mogelijkheden om custom comparators te schrijven en het gemakkelijk testbaar is.
Relevant	- Deze user story is de moeite waard, omdat de afronding hiervan de medewerkersbeschikbaarheidsview vele malen overzichtelijker maakt. De medewerkers voor wie inzet het meest cruciaal is verschijnen bovenaan zodat de Sales-afdeling hier gemakkelijk mee aan de slag kan. Ik ben de juiste persoon om dit uit te voeren, omdat ik kennis heb van de datastructuren en bijbehorende data die gesorteerd moeten gaan worden.
Tijdgebonden	- Deze user story wordt direct uitgevoerd na afronding van de user story “Geef de beschikbare uren voor medewerkers weer”. Het is noodzakelijk hierop te wachten, omdat er anders geen data is om te sorteren.

Sorteer medewerkers per klant	
Specifiek	- Het moet mogelijk worden om medewerkers te sorteren per klant. Dit wilt de afdeling Sales kunnen doen om zo inzichten te kunnen verkrijgen van welke medewerkers bij welke klanten zijn ingezet.
Meetbaar	- Ik weet dat deze user story is afgerond als er per klant kan worden gesorteerd op medewerkers. De uitkomsten hiervan zijn visueel en dus gemakkelijk te controleren.
Acceptabel	- Het doel kan worden bereikt door server-side mee te geven welke klant er bij een bepaalde assignatie hoort. Client side kan hier dan gemakkelijk op worden gesorteerd indien er op de desbetreffende knop wordt gedrukt, omdat de gegevens dan al aanwezig zijn.
Relevant	- Deze issue is minder belangrijk dan de andere, het biedt achtergrondinformatie om betere Sales-beslissingen te nemen maar is op zichzelf niet essentieel om medewerkers in te zetten
Tijdgebonden	- Omdat deze issue minder belangrijk is wordt deze pas uitgevoerd na de hierboven weergegeven user stories, en mogelijk zelfs na sommige Boekhouder requirements. Deze user story is niet essentieel om afgerond te hebben voor afronding van het afstudeerproject.

Geef winstverwachting van het project weer	
Specifiek	- Het moet mogelijk worden om winstverwachtingen weer te geven per project. Dit kan in sommige gevallen gaan om fixed-priced projecten waarbij de enige variable het aantal uren is dat medewerkers er aan moeten besteden. Als het gaat om een hour-based project moeten de kosten ook op een bepaalde manier geschat worden.
Meetbaar	- Ik weet dat deze user story voltooid is als ik de Sales-afdeling een acceptatietest heb laten uitvoeren waarin blijkt dat mijn berekening klopt. Uiteraard moet de berekening eerst met unit tests getest worden.
Acceptabel	- Om deze user story af te ronden moet meer kennis worden vergaard over de data in de Mus-database zodat ik weet welke informatie er beschikbaar is voor deze berekening. De geschatte omzet moet worden afgezet tegen de geschatte kosten.
Relevant	- Deze informatie is nuttig voor een latere versie, maar niet essentieel voor het belangrijkste doel van Mees: namelijk het inzichtelijk maken van medewerkersbeschikbaarheid.
Tijdgebonden	- Vanwege de lagere relevantie is dit ook een user story die later (of niet meer tijdens het afstudeerproject) aan bod komt.

Geef jaarlijkse cumulatieve gegevens weer per klant	
Specifiek	- Het moet duidelijk worden om cumulatieve gegevens per klant weer te geven op een jaarlijks niveau. Zo kan worden ingezien hoeveel medewerkers er aan hebben gezeten, naar welk tarief en hoeveel uren zij gemaakt hebben.
Meetbaar	- Ik weet dat deze user story compleet is ik een goede test heb geschreven waarbij ik handmatig bepaalde gegevens heb uitgerekend uit een beperkte testdataset. Als deze gegevens overeenkomen met de gegevens die door het systeem worden uitgerekend kan het als compleet worden beschouwd.
Acceptabel	- Dit doel kan worden bereikt door de juiste urenregistraties per klant binnen een bepaalde periode (een jaar) bij elkaar op te tellen.
Relevant	- Evenals de twee user stories direct hierboven is dit een story die nuttige achtergrondinformatie geeft, maar niet essentieel is.
Tijdgebonden	- Deze user story wordt eveneens later (of niet tijdens het afstudeerproject) afgerond.

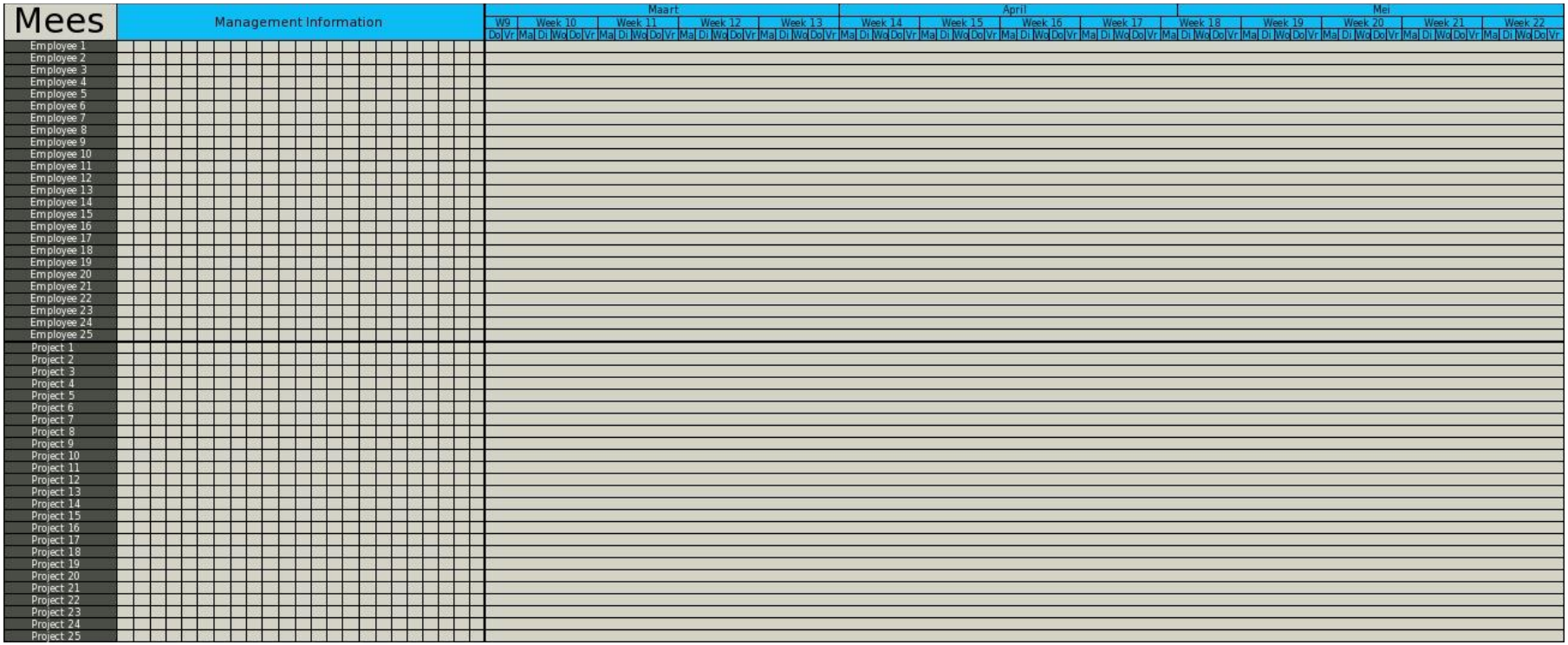
Valideer Mus-data	
Specifiek	- Het moet mogelijk zijn om fouten in de data afkomstig uit de Mus-database die de berekening van medewerkersbeschikbaarheid onmogelijk maken op te sporen en als foutief te markeren. Datums die minstens één validatieschending bevatten krijgen een beschikbaarheid van nul uur met foutmelding en worden bovenaan het overzicht getoond als prikkel om de datakwaliteit te verbeteren.
Meetbaar	- Ik weet dat deze user story compleet is als ik een testdataset gebruik met validatieschendingen erin en er voor deze dag een beschikbaarheid van nul uur uitkomt en deze tevens een foutmelding bevat die aangeeft wat er mis is met de record. Ook kan er worden gecontroleerd of medewerkers die records met schendingen bevatten bovenaan het overzicht verschijnen.
Acceptabel	- Dit doel kan worden bereikt door functies te schrijven die de verschillende schendingen opsporen en als zodanig te markeren
Relevant	- Deze user story is zeer belangrijk, omdat voor een record met schendingen geen berekening kan worden gemaakt en er in deze situatie dus iets moet worden weergegeven, omdat de data anders ook niet verbeterd kan worden.
Tijdgebonden	- Deze user story wordt gelijktijdig opgepakt met het maken van het algoritme dat medewerkersbeschikbaarheid uitrekent.

G. Boekhouder requirements

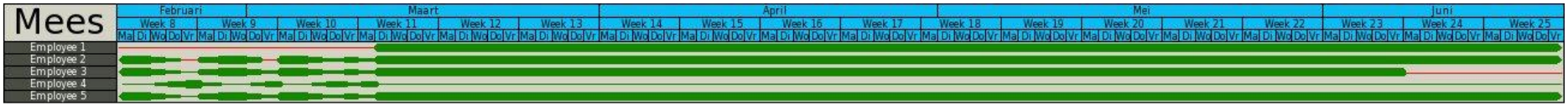
Geef ETC's tot een bepaalde tijd weer voor historische weergave	
Specifiek	- De CEO wenst voor de analyse van een project een datum van weergave uit het verleden kunnen instellen zodat er een historische view kan worden weergegeven met data die voor dat specifieke moment relevant waren. De laatste Estimated to Complete die op dat moment relevant was wordt als leidraad gebruikt. Deze functionaliteit wilt hij hebben om zo de mogelijkheid te creëren de forecasts van een bepaald moment naast de uiteindelijke uitkomst van het project te kunnen leggen. Op deze manier kunnen er voor volgende projecten nieuwe inzichten ontstaan die meegenomen kunnen worden. Enige toevoeging aan de Mees view is benodigd omdat er ergens een datum gekozen moet kunnen worden
Meetbaar	- Ik weet als het doel van deze test bereikt is als dezelfde data als hierboven kan worden weergegeven met uitzondering van de gebeurtenissen na de ingestelde historische datum. De werking van deze berekening kan met een test gecontroleerd worden door de vorige test te dupliceren en aan te passen voor historische datums.
Acceptabel	- Voor een groot deel kan meegelift worden op de berekening behorende bij de voorgaande requirement, enkel moet deze worden uitgebreid om datums na een bepaalde datum niet in het overzicht mee te nemen. Omdat alle gegevens datumnotities bevatten is dit een gemakkelijke opgave.
Relevant	- Deze requirement heeft waarde omdat dit de mogelijkheid geeft het huidige beeld over een project te vergelijken met een historisch beeld. Op deze manier kan gekeken worden of de eerdere inzichten juist waren. Ik ben de juiste persoon om deze requirement af te ronden omdat ik op dat moment al bekend ben met de berekening van de voorgaande requirement.
Tijdgebonden	- Deze requirement wordt opgepakt na afronding van de vorige requirement, omdat deze op dat moment gemakkelijk is om uit te breiden en de afronding van deze requirement een zinnige meerwaarde geeft boven de vorige.

ETC's kunnen worden weergegeven en ingevoerd	
Specifiek	- De CEO wil de mogelijkheid hebben om Estimated-To-Complete datums in te kunnen voeren en weer te kunnen geven in het systeem. Dit is een toevoeging op de vorige twee requirements, zodat er niet meer via een extern systeem data ingevoerd hoeft te worden. Wel moet wederom de user interface worden aangepast om deze datums te kunnen presenteren.
Meetbaar	- Ik weet dat deze issue compleet is als de ETC's die ik aan Postduif presenteer weer terug te zien zijn in Mees. Voor de tests wordt er gebruik gemaakt van een in memory database zodat er geen gegevens in de productiedatabase aangepast hoeven te worden.
Acceptabel	- Dit is simpele CRUD-activiteit die gemakkelijk kan worden uitgevoerd middels de
Relevant	- Het toevoegen van deze functionaliteit is relatief weinig werk en voegt een hoop gebruiksvriendelijkheid toe.
Tijdgebonden	- Deze requirement wordt opgepakt na afronding van de vorige requirement, omdat de mogelijkheid om in Mees ETC's toe te voegen vanaf dat moment eigenlijk pas nuttig is. Dit is een secundair doel, want op zichzelf zegt een ETC niet zoveel. Gecombineerd met de vorige requirement is het echter van grote waarde.

H. Grafische ontwerpen



Management view met informationmodule en tijdslijn.



Salesmodule met invulling, gesorteerd op availability met records die errors bevatten bovenaan het overzicht.

I. Volledige conceptoplossingen Same Origin Policy

I. Cross Origin Resource Sharing

Het is mogelijk om HTTP headers in te stellen in de Apache configuratie van Postduif die communicatie toestaan tussen Mees en Postduif door het instellen van de volgende header:

"*Access-Control-Allow-Origin: http://mees.42.nl*"

Dit is een header die het uitvoeren van Cross-domain calls vanaf vastgestelde domeinen toestaat, en is vastgelegd in een W3C standaard[2]. Dit is een goed teken voor de ondersteuning in de toekomst, een functie zoals deze die is vastgelegd in een standaard wordt waarschijnlijk niet zo snel weer uit browsers gehaald.

Veiligheid
Met deze methode worden de toegestane "Origins" toegepast op een niveau waar gebruikers geen toegang toe kunnen verkrijgen, namelijk het Apache-configuratiebestand. Het is onmogelijk om andere waarden op te geven dus is deze methode zeer veilig. Enkel calls van mees.42.nl worden toegestaan onder deze policy.

Onderhoudbaarheid
Met deze methode bestaat er echter wel het risico dat de oplossing op een later moment gebroken raakt, omdat in geval van een eventuele verhuizing van Mees het configuratiebestand van Postduif aangepast zou moeten worden. Een ontwikkelaar die nog niet veel kennis over Mees heeft vergaard kijkt mogelijk niet als eerste in zo'n bestand. Dit is echter wel op te lossen door een duidelijke beschrijving te geven (op Confluence).

Portabiliteit
Omdat de Apache-configuratie van de omgevingen van 42 er nagenoeg hetzelfde uitzien is het gemakkelijk om deze oplossing over te dragen naar een andere omgeving. Het enige waar hierbij op gelet moet worden is dat er naar de goede corresponderende Mees-omgeving wordt gewezen.

De ontwikkelomgeving is hierbij iets lastiger, omdat er gebruik wordt gemaakt van een Instant Developer Experience met behulp van Jetty. Deze configuratie zit iets anders in elkaar, maar hoeft maar eenmaal gedefinieerd te worden om calls vanaf "localhost" toe te staan, en is daarom dus geen show stopper.

Naleving beoogde architectuur
Deze methode past goed binnen de eis om de beoogde architectuur na te leven: het creëren van een backend is niet nodig en de calls kunnen direct vanaf Mees naar Postduif worden uitgevoerd.

II. Set HTTP headers in Postduif Responsebody

Een andere oplossing die mogelijkheden biedt is het instellen van dezelfde HTTP headers in de Postduif ReponseBody van de controllerfuncties. Het voordeel hiervan is dat de adressen direct in Postduif staan genoteerd, wat in het geval van een minimale hoeveelheid applicaties overzichtelijker kan zijn dan een apart Apache-configuratiebestand.

Veiligheid
Evenals bij de bovengenoemde oplossing is het onmogelijk om toegang te krijgen tot de beveiligingsinstellingen, omdat deze in de source code zijn verworven van Postduif, die in gecompileerde vorm op de server draaien.

Onderhoudbaarheid
Net als bij de vorige oplossing geldt dat in het geval van verplaatsing van Mees er wijzigingen moeten worden gemaakt in de Postduif-configuratie, hetzij ditmaal in de source code.

Portabiliteit
Waar bij de vorige oplossing een nette scheiding aanwezig was tussen source-code van Postduif en de omgeving-specifieke configuratie die calls vanaf Mees toestaat is dit bij deze oplossing niet het geval. In de source code kan op geen enkele manier gededuceerd worden op welke omgeving het programma draait. Gevolg is dat je alle omgevingen van Mees toe zou moeten voegen aan de Responsebody headers.

Behalve het feit dat dit deuren openzet om naar de verkeerde omgevingen calls uit te kunnen voeren (van de testomgeving van Mees naar de productie-omgeving van Postduif bijvoorbeeld) levert dit niet bepaald elegante code op. Naarmate het aantal applicaties dat op eenzelfde manier met Postduif communiceert toeneemt moeten voor al deze applicaties en alle bijbehorende omgevingen de domeinnamen toegevoegd worden aan de header. Op een gegeven moment levert dit een bulk aan onleesbare code op.

Naleving beoogde architectuur
Evenals bij de vorige is het bij deze oplossing mogelijk om calls direct vanuit Mees uit te voeren naar Postduif zonder noodzaak voor een tussenliggende backend.

III. Schrijf Mees-backend

Een andere oplossingsrichting is het schrijven van een (eenvoudige) backend voor Mees die tussen de frontend en Postduif in gaat zitten, hierdoor ben je niet gebonden aan browserrestricties. Deze oplossing strookt niet met de beoogde architectuur, maar is voor de volledigheid toch opgenomen in dit overzicht.

Veiligheid
Omdat de calls niet vanuit een browser kunnen worden uitgevoerd is dit een zeer veilige oplossing. Op geen enkele andere manier is het mogelijk om calls naar Postduif uit te voeren dan via de source code van de backend. Zelfs als iemand zelf een Java-project zou opzetten om calls uit te voeren naar Postduif dan hebben ze authenticatie nodig om dit te kunnen doen.

Onderhoudbaarheid
Deze oplossing is een stuk minder onderhoudbaar dan de vorige twee: weliswaar is er geen noodzaak voor een extern configuratiebestand waar de toegestane domeinnamen in opgenomen worden, maar het feit dat er een project opgenomen moet worden enkel om calls door te sturen creëert een extra laag die onderhouden moet worden en waarop fouten kunnen gaan ontstaan. Dit moet allemaal onderhouden worden terwijl dit eigenlijk helemaal niet nodig is als een oplossing zoals de vorige twee gebruikt wordt.

Portabiliteit
Deze oplossing is extreem makkelijk over te nemen naar andere omgevingen, omdat er namelijk helemaal niets hoeft te veranderen. De Mees-backend zou blijven werken op welk domeinadres deze ook geplaatst wordt.

Naleving beoogde architectuur
Een expliciete eis aan Mees is de mogelijkheid om zonder tussenkomst van een extra middenlaag te communiceren met Postduif, omdat dit in het verleden bij andere projecten van 42 BV voor mismatches heeft gezorgd. Deze oplossing staat haaks tegenover deze eis, en is daarom ook afgevallen.

IV. JSON with padding gebruiken

JSONP *[4]* is een complementaire implementatie van JSON die is ontworpen met het doel om data van een server uit een ander domein aan te vragen. Normaliter verbiedt de Same Origin Policy het communiceren van bijvoorbeeld server1.example.com naar een ander domein.

Een uitzondering hierop is het HTML <script> element. JSONP maakt misbruik van deze open policy door JavaScript-code aan te maken op basis van het opgehaalde JSON-object. De code wordt dan ook geëvalueerd door de JavaScript interpreter en niet geparst door een JSON parser.

De structuur van een JSONP teruggave wijkt echter af van een regulier JSON object, stel de volgende JSON voor die vanaf een server wordt teruggegeven:

Reguliere JSON	**JSON with Padding**
{"Name": "Foo", "Id" : 1234, "Rank": 7}	functionCallBack({"Name": "Foo", "Id" : 1234, "Rank": 7});

Het voordeel van deze methode is dat het erg simpel is om werkend te krijgen in combinatie met jQuery. In de Ajax functie kun je opgeven dat je een formaat "jsonp" terug wilt krijgen, let wel dat de functionCall serverside moet worden toegevoegd om dit werkend te krijgen.

Het feit dat je server-side aanpassingen moet doen aan de structuur van de JSON-teruggave is het grote nadeel van deze methode. Dit zorgt ervoor dat je afwijkt van een bepaalde standaard, wat vervelend kan zijn als je meerdere applicaties hebt die tergen de server praten (zoals bij Postduif). Je zou aparte controllerfuncties kunnen schrijven voor JSON en JSONP teruggaven, maar mooier wordt dit er niet op.

Veiligheid
Van alle onderzochte mogelijkheden is JSON with padding zonder enige twijfel de meest onveilige. Het staat calls toe vanaf ieder webadres door misbruik van een loophole in de Same Origin Policy. Aan de serverkant moet er een callback-functienaam omheen gewrapd worden.

Dit betekent dat er een laag van security minder is dan de andere oplossingen: waar in dat geval de eerste laag van beveiliging de expliciet goedgekeurde “Origin” is en de tweede laag de userauthenticatie, is in het geval van JSONP alleen de tweede mogelijk. Uiteraard is het wel mogelijk om in Postduif handmatig te controleren wat de Origin van een JSON with padding call is en deze al dan niet goed te keuren, maar in dit geval neem je het enige voordeel van deze oplossing weg: namelijk het feit dat er geen applicatie-specifieke wijzigingen in hoeven te worden geplaatst in de backend.

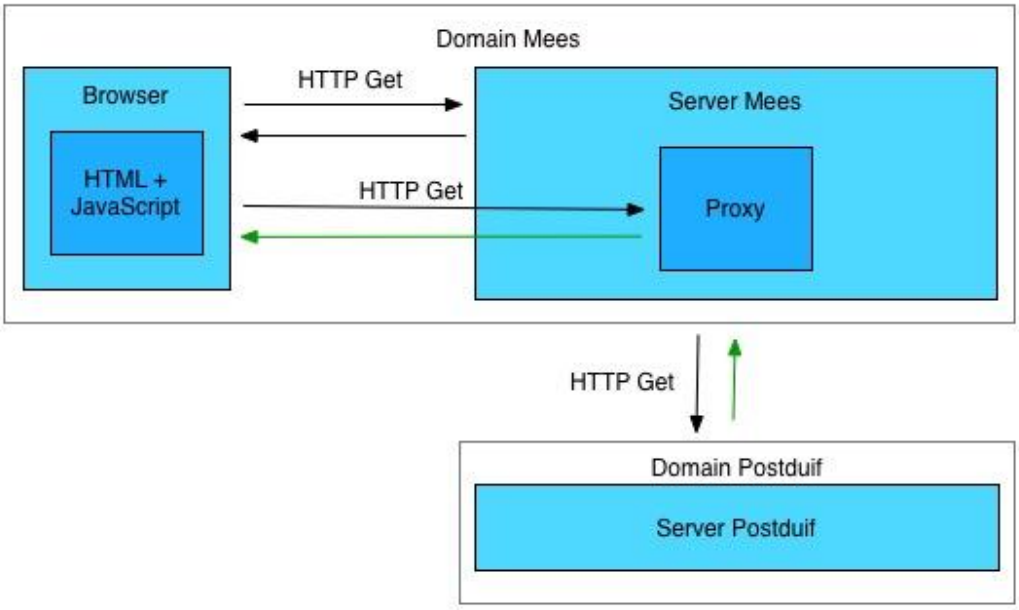
Onderhoudbaarheid
Deze oplossing is redelijk onderhoudbaar, het enige punt is dat er om iedere controllerfunctie een callback moet worden gewrapt waarmee wordt afgeweken van de JSON-standaard. Als developers dit niet weten implementeren ze dit waarschijnlijk niet bij toevoeging van een nieuwe controller. Als er dan later een andere applicatie met dezelfde controller moet praten waarbij JSON with padding wel benodigd is, levert dit problemen op.

Portabiliteit
Deze oplossing is niet omgeving-specifiek en daarom gemakkelijk over te nemen naar de andere omgevingen van Mees en Postduif.

Naleving beoogde architectuur
Deze oplossing komt overeen met de eis om Mees te bouwen zonder tussenliggende backend.

V. Proxy-server opzetten

Tenslotte is er de mogelijkheid om een proxy *[5]* op te zetten voor de REST-call. In plaats van requests direct naar de server van Postduif te sturen worden deze via een proxy op de Mees webserver uitgevoerd. Deze proxy stuurt de call door naar de Postduif server en geeft de teruggegeven waarde terug aan de browser. Voor de duidelijkheid van de "route" is de teruggave van de Postduif server groen getekend in plaats van zwart.



In deze oplossing gedraagt de Proxy zich echter in essentie als een impliciete Mees-backend, die we juist niet willen.

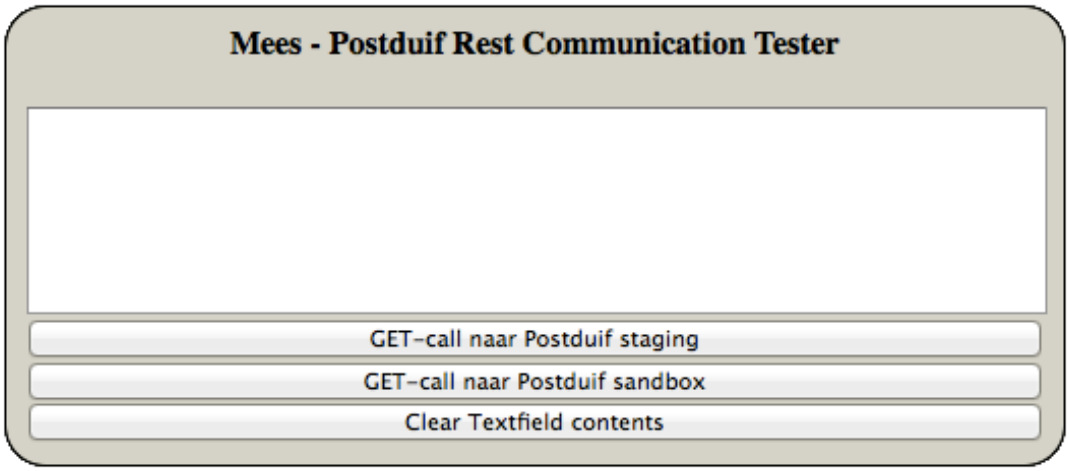
Veiligheid
Deze oplossing is veilig, omdat er enkel calls vanaf de proxy kunnen worden uitgevoerd. Wellicht is het wel handig om een filter op te nemen in de proxy zelf zodat het geen calls doorstuurt vanaf ieder adres.

Onderhoudbaarheid
Deze oplossing is redelijk onderhoudbaar, ware het niet dat er een extra laag tussen Mees en Postduif wordt ingebouwd waar wederom een hoop fouten in zouden kunnen ontstaan.

Portabiliteit
Deze oplossing is makkelijk uit te breiden naar andere omgevingen, omdat deze met dezelfde proxy kunnen communiceren.

Naleving beoogde architectuur
Deze oplossing komt niet overeen met de eis om Mees te bouwen zonder tussenliggende backend.

J. Proof-of-concept Cross Origin Resource Sharing



Allereerst wilde ik weten wie er precies toestemming moest verlenen voor de call naar Postduif: Mees, waar de browser op dat moment op stond en calls naar externe domeinen blokkeerde, of Postduif waar een call naartoe wordt geroepen. Daarom wilde ik als eerste testen of een headertoevoeging aan de Mees configuratie werkt.

Daarom heb ik een Apache server ingericht en geconfigureerd. Hier heb ik de Proof-of-Concept pagina op gezet. Vanaf deze pagina werd een verzoek naar de Postduif-server gedaan, die werd weergegeven indien deze geslaagd was.
Met de headers aan de Mees-kant kreeg ik het niet werkend, en kreeg ik wederom een onduidelijke foutmelding terug.

Vervolgens heb ik geprobeerd om dezelfde header toe te voegen aan de apache-configuratie van de Postduif testomgeving, maar ditmaal met het domeinadres van de Mees testomgeving als toegestaan adres. Bij uitvoer van de opdracht verscheen het resultaat in de tekstbalk. Het bleek dus dat de header aan de ontvangende kant (Postduif) opgenomen moest worden en niet aan de zendende kant (Mees).

K. Cross Origin Resource Sharing browser support

Cross-Origin Resource Sharing - Working Draft

Method of performing XMLHttpRequests across domains

Resources: Mozilla Hacks blog post Alternative implementation by IE8
Demo and script with cross-browser support

*Usage stats:	Global
Support:	57.8%
Partial support:	28.3%
Total:	86.1%

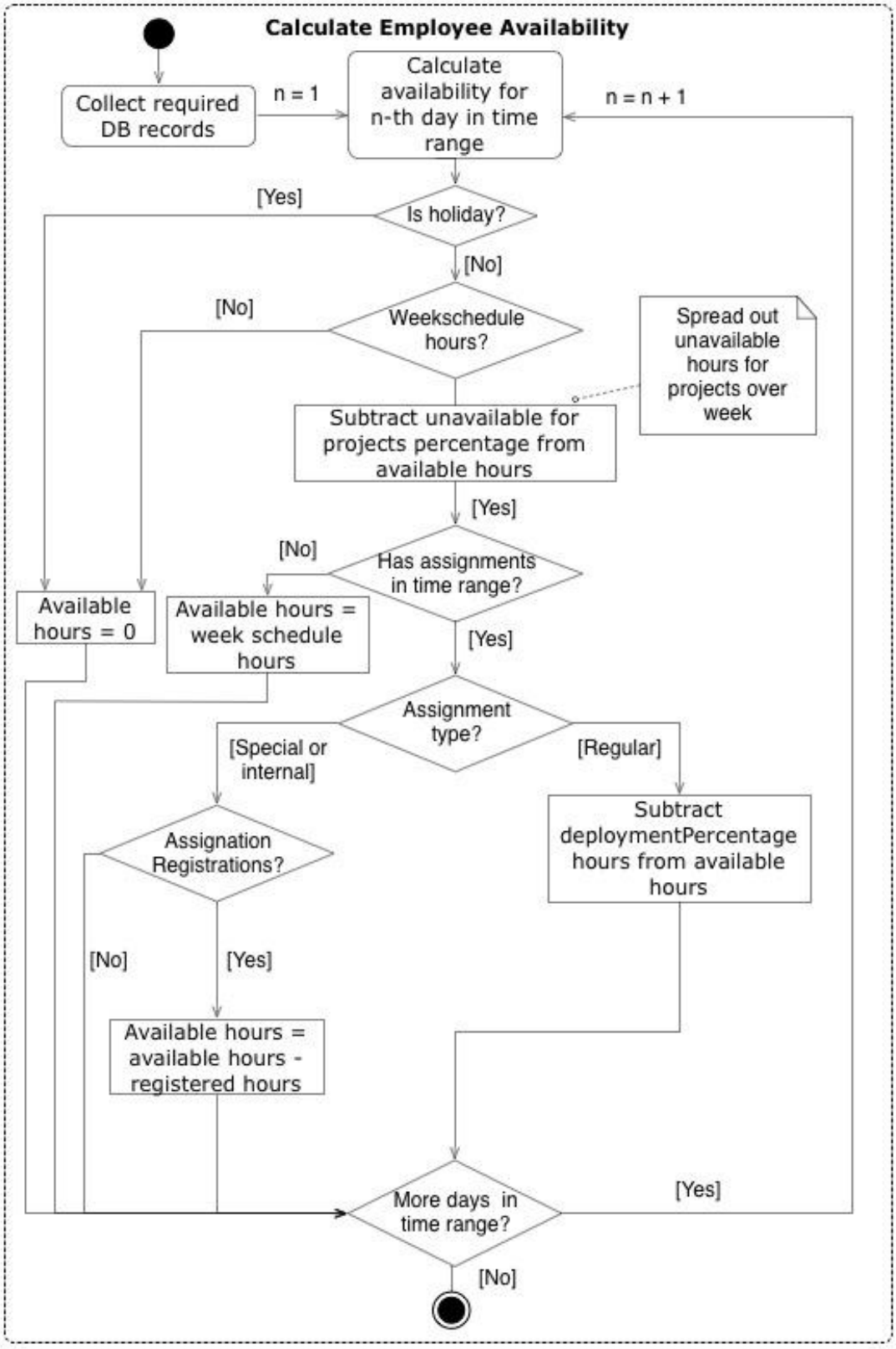
Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Opera Mobile	Android Browser
								10.0	2.1
	6.0	3.6				3.2		11.0	2.2
	7.0	8.0				4.0-4.1		11.1	2.3
	8.0	9.0	16.0	5.0		4.2-4.3		11.5	3.0
Current	9.0	10.0	17.0	5.1	11.6	5.0	5.0-6.0	12.0	4.0
Near future	10.0	11.0	18.0	6.0	12.0				
Farther future		12.0	19.0						

Note: Supported somewhat in IE8 and IE9 using the XDomainRequest object

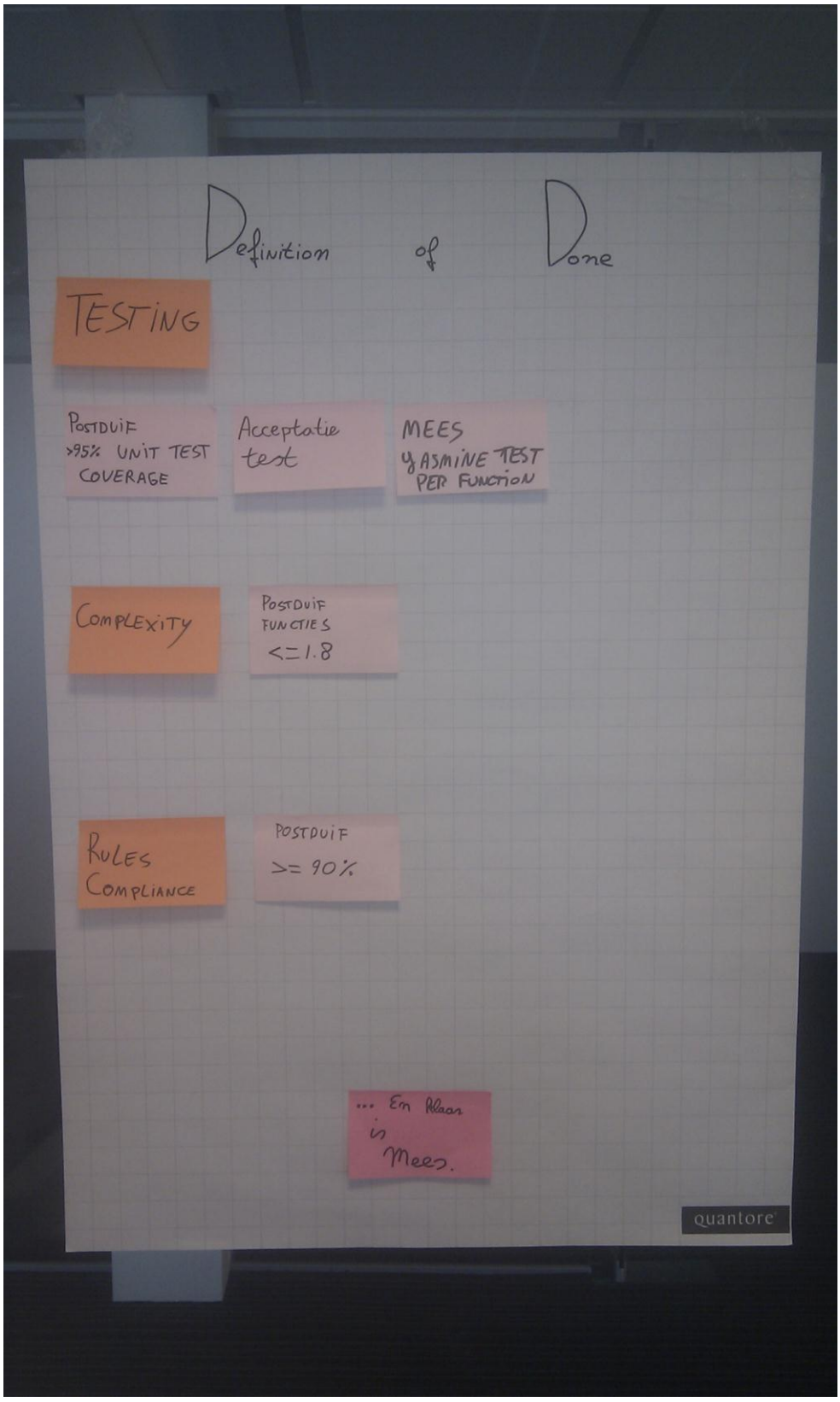
Feedback

Bron: http://caniuse.com/cors

L. Activity diagram berekening medewerkersbeschikbaarheid



M. Definition of done



De complexity en rules compliance zijn meetwaarden van Sonar.

N. Sprint-planningen

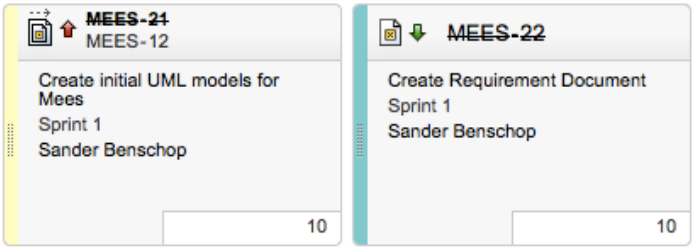
In deze bijlage kunnen de verschillende sprints gezien worden die zijn uitgevoerd tijdens de afstudeerstage. De afbeeldingen zijn afkomstig uit Jira, en bevatten meerdere types story's die een andere kleur hebben in de weergave.

De codes bovenaan (bijv: MEES-21) geven de identifier van de user story aan. Indien het gaat om een subtaak, wordt de identifier van de subtaak bovenaan aangegeven en de supertaak daaronder. Een doorgestreepte waarde betekend dat deze geresolved is.

Perioden waarin niet aan het product zelf is gewerkt maar waarin enkel tijd is besteed aan het inlezen in de bestaande architectuur, het schrijven van het PvA en het schrijven van het afstudeerverslag zijn niet in Sprints ingepland, omdat dit voor deze perioden niet erg zinnig leek vanwege het feit dat er een vaste hoeveelheid tijd voor was ingepland en er lastig user story's uit te destilleren waren.

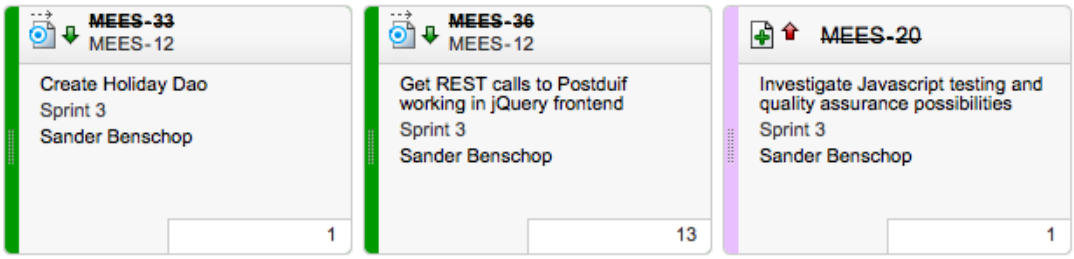
Het gaat om de eerste twee weken (6 t/m 17 februari), waarin op de architectuur werd ingelezen en het PvA werd geschreven, en de laatste drie weken waarin de volledige tijd in het verbeteren en afronden van het afstudeerverslag zat (14 mei t/m 1 juni).

Sprint 1 (20 februari – 24 februari)



Sprint 2 (27 februari – 2 maart)

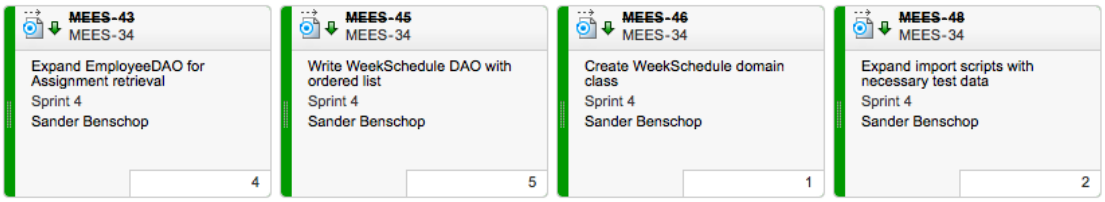
Sprint 3 (5 maart – 9 maart)



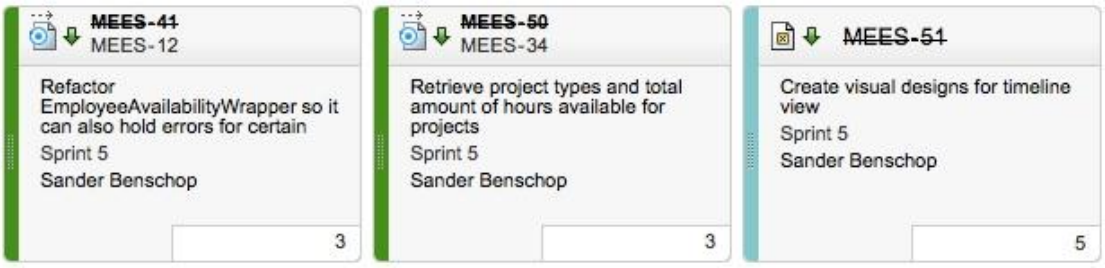
Sprint 3.5 (12 maart – 16 maart)

In de periode van Sprint 3 was de product owner verhinderd wegens ziekte, ook op het moment dat er een nieuwe planning gemaakt moest worden. Toevalligerwijs bleek issue MEES-36 groter uit te pakken dan gedacht, zodat er in de week erna is verder gegaan met die user story. Nadat deze is afgerond is de rest van de tijd in het afstudeerverslag gestoken.

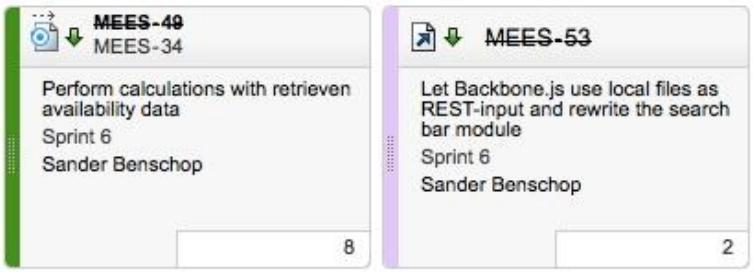
Sprint 4 (19 maart – 23 maart)



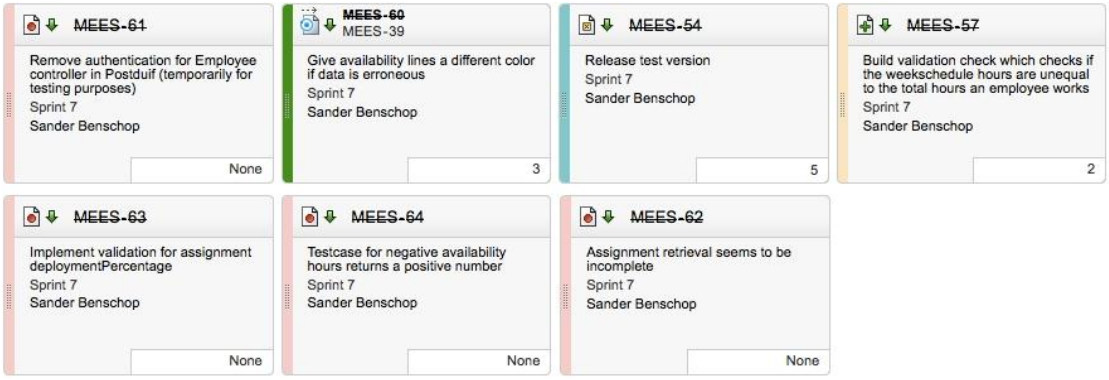
Sprint 5 (26 maart – 30 maart)



Sprint 6 (2 april – 6 april)

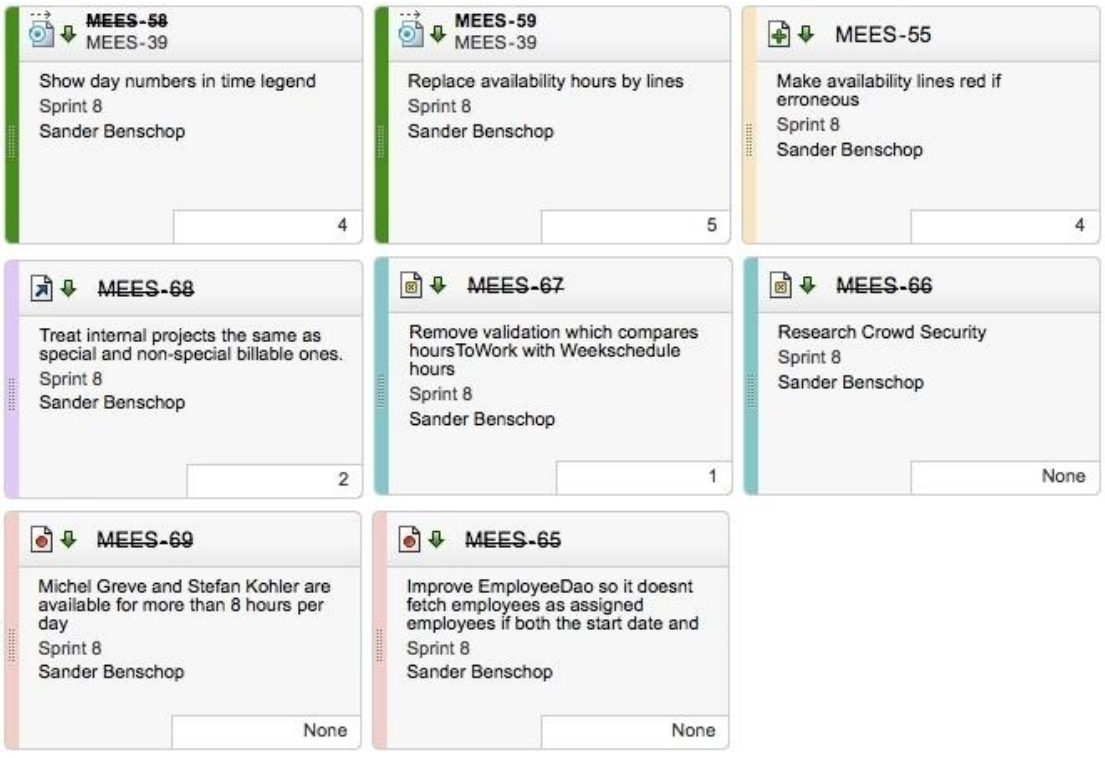


Sprint 7 (9 april – 13 april)



De rode "kaarten" uit dit overzicht zijn geconstateerde bugs, deze werden na de Sprint-planningen geconstateerd en dienden zo snel mogelijk opgepakt te worden. Om deze reden is er geen aantal story points voor gegeven.

Sprint 8 (16 april – 20 april)



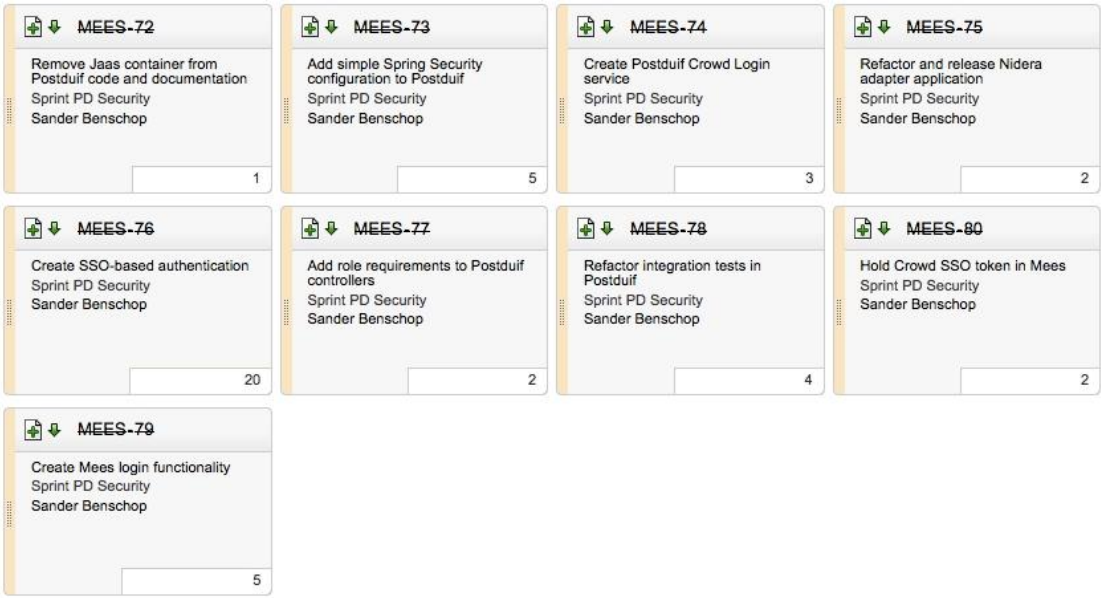
In deze sprint zijn de story's MEES-59 (bovenaan weergegeven) en MEES-55 helaas niet afgekomen, dit ten gunste aan de research naar Crowd Security. Hiervoor waren geen story points ingeschat, omdat dit onder research valt en er geen concrete user story bedacht kon worden voor het research was afgerond, omdat we niet wisten wat de aanpassing van Crowd Security in Postduif precies inhield.

Om dezelfde reden als in Sprint 7 hebben de bugs uit dit overzicht geen aantal story points.

Sprint 9 (23 april – 27 april)

In deze Sprint werd het onderzoek voortgezet. De uitkomsten zijn te lezen in de kern van het verslag in hoofdstuk 8.9: *"Crowd Security"*.

Sprint 10/11 Postduif Security (30 april – 11 mei)



Binnen het systeem is het niet mogelijk om een story aan meerdere personen toe te wijzen, terwijl dit wel de aanpak was voor de meeste weergegeven story's. tijdens het project zoals in hetzelfde hoofdstuk gelezen kan worden.

De wijzigingen die specifiek voor Mees zijn heb ik alleen gedaan, gelijktijdig met de wijzigingen die mijn collega's aan Snip hebben gemaakt. Omdat ik hier niet aan heb meegewerkt zijn deze weggelaten uit het overzicht.

De reden dat deze sprint twee weken besloeg was het feit dat we in deze sprint werkten met een grote user story waar we niet genoeg kennis over hadden om in meerdere delen te splitsen, namelijk MEES-76. Geen van de aanwezigen had ervaring met de implementatie van dit framework, dus was het inschatten ook lastig. Deze activiteiten waren dus makkelijker in een twee-weekse sprint te passen.

O. Afstudeerplan

Afstudeerplan

Informatie afstudeerder en gastbedrijf (*structuur niet wijzigen*)

Afstudeerblok: 2012-1.1 (start uiterlijk 6 februari 2012)
Startdatum uitvoering afstudeeropdracht: 06-02-2012
Inleverdatum afstudeerdossier volgens jaarrooster: 1 juni 2012

Studentnummer: 08007381
Achternaam: dhr (*) *weghalen niet van toepassing*
Voorletters: A
Roepnaam: Sander
Adres: Essehout 29
Postcode: 2719MD
Woonplaats: Zoetermeer
Telefoonnummer: 079-3614062
Mobiel nummer: 0611885444
Privé emailadres: sander1990@gmail.com

Opleiding: Informatica
Locatie: Zoetermeer (*) *weghalen niet van toepassing*
Variant: voltijd

Naam studieloopbaanbegeleider: Arno Nederend
Naam begeleidend examinator: Arno Nederend
Naam tweede examinator: Nanny Jacobs

Naam bedrijf: 42 BV
Afdeling bedrijf:
Bezoekadres bedrijf: Koraalrood 33
Postcode bezoekadres: 2718 SB
Postbusnummer: 6003
Postcode postbusnummer: 2702 AA
Plaats: Zoetermeer
Telefoon bedrijf: 088-4242042
Telefax bedrijf: 088-4242099
Internetsite bedrijf: http://www.42.nl

Achternaam opdrachtgever:Ratsma (*) *weghalen niet van toepassing*
Voorletters opdrachtgever*: H*
Titulatuur opdrachtgever: Dhr.
Functie opdrachtgever: Project Manager
Doorkiesnummer opdrachtgever: -
Email opdrachtgever: harko.ratsma@42.nl

Achternaam bedrijfsmentor: Ratsma (*) *weghalen niet van toepassing*
Voorletters bedrijfsmentor: H
Titulatuur bedrijfsmentor: Dhr.
Functie bedrijfsmentor: Project Manager
Doorkiesnummer bedrijfsmentor: -
Email bedrijfsmentor: harko.ratsma@42.nl
NB: bedrijfsmentor mag dezelfde zijn als de opdrachtgever

Doorkiesnummer afstudeerder*:* -
Functie afstudeerder (deeltijd/duaal): Software Developer

Titel afstudeeropdracht: Ontwikkelen webapplicatie “Mees”

Opdrachtomschrijving

1. Bedrijf

- ***Geschiedenis***

  De oprichter van 42 BV, Eric Meier, was tot 2003 werkzaam als directeur van enkele Nederlandse bedrijfstakken van de multinational Software AG, waarvan het hoofdkantoor in Duitsland is gevestigd. Wegens een afnemende winst in het buitenland werd het Nederlandse kantoor gesloten, ondanks het feit dat deze nog wel winstgevend was. Eric Meier besloot de ruimte in de markt die hierdoor vrij zou komen te benutten voor een eigen bedrijf.

  In 2005 is 42 BV een samenwerkingsverband aangegaan met het Amsterdamse Kensas. De twee bedrijven zijn ondergebracht onder de holding genaamd M.A.D. Dit staat voor Mice and Dolphins, de twee meest intelligente diersoorten op aarde volgens the Hitchhiker’s Guide To the Galaxy, het boek waar eveneens de naam 42 van afkomstig is. Tegenwoordig wordt de naam Kensas zo minimaal mogelijk gebruikt, de naam 42 is hiervoor in de plaats gekomen.

  42 BV is inmiddels uitgegroeid tot een bedrijf met 40 medewerkers. Enkelen hiervan werken op kantoor in Zoetermeer of Amsterdam, maar de meeste werken intern bij de klant.

- ***Bedrijfsprofiel***

  De missie van 42 BV is het ontwikkelen van hoogwaardige software waarmee haar klanten hun werkprocessen en bedrijfsresultaten kunnen verbeteren. 42 BV ontwikkelt maatwerk Java-enterprise producten, en doet dit het liefst open source. Verder helpt 42 BV grote klanten bij het integreren van Atlassian-producten. 42 BV gebruikt deze software zelf ook en is partner van Atlassian.

  Binnen 42 hangt een informele, vriendelijke werksfeer. Van medewerkers wordt een hoge mate van zelfredzaamheid verwacht, wat ze proberen te bereiken middels scholing in diverse delen van het vakgebied die net iets buiten de “comfort-zone” van de medewerkers liggen. Door Java-developers ook in aanraking te komen met de details van frontend-development en databases is het idee dat zij beter in staat raken om bij het gehele proces betrokken te zijn.

  Bij het ontwikkelen zelf staan de volgende kwaliteitsprincipes centraal binnen 42 BV: het bouwen van software van lage complexiteit, die grondig getest is en vrij is van conventieschendingen. Het bedrijf gelooft dat een investering in kwaliteit op deze manier kostenverlagend werkt op de lange termijn, omdat het onderhouden van software vergemakkelijkt wordt.

- ***Klantprofiel***

  42 BV heeft na de oprichting enkele grote klanten van Software AG kunnen behouden, waaronder Schiphol en CenterParcs. Hierdoor kon 42 BV meteen beginnen met grotere opdrachten, iets wat lastig kan zijn voor beginnende bedrijven. Enkele andere (recente) grote klanten van 42 BV zijn Albert Heijn, de Consumentenbond, de Belastingdienst, ANWB, Essent en Nidera.

2. Probleemstelling

42 BV heeft op dit moment al diverse interne applicaties die vorig jaar verzameld zijn onder de paraplu-applicatie “Volière”. Onderdelen hiervan is bijvoorbeeld het registratiesysteem “Mus”. Hierin worden de gewerkte en geplande uren geregistreerd en vervolgens opgeslagen in een database. Deze gegevens wil 42 BV gaan gebruiken om een tool te creëren die de managers en de salesafdeling van informatie kunnen voorzien om betere beslissingen te maken wat betreft medewerkersinzet bij projecten.

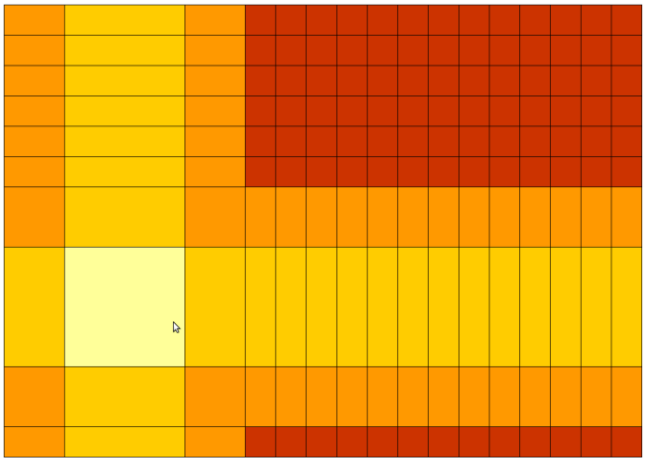
De salesafdeling heeft op dit moment problemen met het efficiënt inzetten van medewerkers op projecten, wat in het verleden geresulteerd heeft in het feit dat medewerkers “op de bank” terecht kwamen. Voor het management is het daarentegen lastig om de mederwerkers binnen lopende projecten efficiënt in te zetten. In beide situaties is het gebrek aan gecentraliseerde, duidelijke actuele en historische data het probleem.

3. Doelstelling van de afstudeeropdracht

De doelstelling van de opdracht is het inzichtelijk maken van historische en actuele data die gebruikt kan worden om betere bedrijfskundige beslissingen te nemen over medewerkersinzet en projectplanningen, alsmede het plaatsen van enkele forcasts in dezelfde weergave.

Er is een tijd nagedacht over hoe dit het beste opgelost kan worden en uiteindelijk is besloten om een webapplicatie te maken waarin vanuit verschillende perspectieven (historische) informatie kan worden ingezien in een soort “fisheye” view zoals hiernaast gezien.

Iedere cel is gevuld met kleine hoeveelheden data, waarbij de relevante informatie eromheen kan worden gezien door hier over heen te hoveren/klikken. Het idee is dat de belangrijke historische data aan de linkerkant wordt weergegeven tot een bepaald moment in het verleden. Aan de rechterkant worden forecasts over toekomstige situaties geplaatst.



Naarmate data ouder wordt is deze minder geschikt om lessen uit te leren, omdat de situatie in de loop der tijd zoveel veranderd dat deze niet per sé meer toepasbaar is op de huidige situatie. Een overschot aan onzinnige informatie maakt de tool alleen maar minder nuttig. De granulariteit van zowel historische data als forecasts wordt grover naarmate er verder van de huidige datum wordt afgegaan.

Gedurende het verloop van de afstudeerperiode zal er regelmatig met de opdrachtgever gesproken worden over de voortgang van- en de tevredenheid over het project tot op dat moment. Op deze manier kan worden voorkomen dat een van beide partijen aan het einde van de periode met ongewenste verrassingen komt te zitten.

Mijn doel is bereikt als de stakeholders binnen 42 BV de webapplicatie kunnen gebruiken om op een gemakkelijke manier alle informatie tot zich te nemen om zo betere beslissingen te kunnen nemen over de inplanning van medewerkers bij projecten. Ik ga dit doel waarborgen door regelmatig contact te houden met de verschillende stakeholders om te kijken of mijn visie op het te realiseren werk nog strookt met dat van hen, zoals verderop in het document gelezen kan worden onder het kopje “Methodiek”.

Bekend is al dat 42 BV hoge eisen stelt aan de kwaliteit van de code die haar medewerkers/stagiairs schrijven. Verder wordt bij aanvang van het project met de verschillende stakeholders besproken wat zij verder als kritieke succesfactoren zien, zodat hier al vanuit het begin rekening mee kan worden gehouden. De kritieke succesfactoren zullen worden vastgelegd in Confluence, zodat hier gemakkelijk naar kan worden teruggekeken.

4. Resultaat

Wanneer de opdracht succesvol is afgerond zal 42 BV beschikken over een nieuwe webapplicatie die ze in staat stelt om betere beslissing te nemen over de inplanning van medewerkers bij projecten. Dit zou prijzige inplanningsfouten in het vervolg moeten helpen voorkomen. Tevens zal er meer informatie beschikbaar zijn over de inrichting van een ontwikkelstraat op webgebied voor 42 BV. Deze kennis kan vervolgens worden hergebruikt voor toekomstige projecten

5. Uit te voeren werkzaamheden, inclusief een globale fasering, mijlpalen en bijbehorende activiteiten

Tijdens de afstudeerstage worden de volgende werkzaamheden uitgevoerd:

- ***Plan van aanpak schrijven (3 dagen)***

  De eerste stap van de afstudeerstage zal het schrijven van een plan van aanpak zijn, waarin ik duidelijk schets wat de route is die ik denk te gaan bewandelen gedurende mijn afstudeeropdracht. Hierin zal onder andere een globale planning terugkomen en een overzicht van de verwachte impediments van het project.

- ***Requirements opstellen (7 dagen)***

  Na afronding van het plan van aanpak wordt begonnen aan het inventariseren van de wensen van de (hoofd)gebruikers. Los van de functionele requirements zullen hier eveneens de kwaliteitsrequirements in terugkomen. De eerdere gesprekken over de kritieke succesfactoren zullen eveneens worden meegenomen in deze inventarisatie. Alle informatie samen wordt vervolgens gebruikt om requirements mee op te stellen die direct gebruikt wordt vanaf de ontwerpfase.

- ***Analyseren data uit Mus-database (5 dagen)***

  Voordat er begonnen zal worden met het modelleren van de te creëren webapplicatie, wordt eerst goed de tijd genomen voor het analyseren van de data uit de Mus database. Deze tijd zal worden gebruikt om kennis te vergaren van de vorm en inhoud van de aanwezige data, zodat er een beeld ontstaat van de informatie die benodigd is voor de webapplicatie. Een domeinmodel kan worden gegenereerd vanuit het RDBMS om een duidelijk, schematisch beeld te krijgen van de databasestructuur.

- ***Opstellen initiële ontwerpen webinterface, backend en samenhang (10 dagen)***

  Alvorens er begonnen kan worden aan het programmeren van de webapplicatie of de Java backend worden eerst de intiële opzet van de UML-diagrammen gemaakt. Deze zullen als hulpmiddel worden gebruikt tijdens de verdere ontwikkelingen. In ieder geval word(en) een klassediagram, usecasediagrammen (met beschrijvingen) en sequentiediagrammen gecreëerd. Een combinatie van deze diagrammen geeft een voldoende beeld over de wensen aan- en structuur van- de webapplicatie en de Java backend om deze te kunnen ontwikkelen.

  Na opzet van de initiële ontwerpdocumenten zal worden begonnen aan de ontwikkeling van de webapplicatie. Tijdens deze ontwikkelingen wordt meer cachet gegeven aan de verschillende modellen en worden verdere modellen die de ontwikkelingen van de verschillende functionaliteiten vergemakkelijken iteratief toegevoegd, gezamenlijk met de implementatie hiervan. Deze iteratieve aanpak helpt enerzijds om duidelijk na te denken over het ontwerp alvorens er begonnen wordt met programmeren, en biedt anderzijds ook de mogelijkheid om voortschrijdende kennis mee te nemen in het ontwerp van de webapplicatie.

- ***Het ontwikkelen van een Java backend voor de communicatie tussen de webinterface en de Mus database (20 dagen)***

  Na afronding van de initiële ontwerpfase wordt begonnen met het ontwikkelen. Allereerst zal een opzet voor de backend worden gemaakt zodat er vanuit de webinterface tegenaan kan worden gesproken. Voor iedere opvolgende functionaliteit zal na het maken van nieuwe ontwerpen in principe eerst het backend gedeelte worden ontwikkeld, gevolgd door het webinterface gedeelte. Deze iteratieve aanpak helpt met het verifiëren dat iedere losstaande functionaliteit doet wat er van verwacht wordt, wat de uiteindelijke koppeling tussen de functionaliteiten zal vergemakkelijken.

  Er wordt een Java-backend opgezet die enerzijds via REST-calls communiceert met de op te zetten webinterface en anderzijds de communicatie legt met de Mus database via de Java-module “Postduif” van de volière. Postduif is verantwoordelijk voor alle communicatie tussen de verschillende applicatie van 42 en de Mus database. In de backend wordt de aangeleverde data gefilterd en vervolgens doorgegeven aan de webinterface.

  De backend zal geprogrammeerd worden middels Test Driven Development om te voldoen aan de kwaliteitseisen van 42 BV op het gebied van software testing.

- ***Het ontwikkelen van een JavaScript/jQuery webinterface in combinatie met HTML5 en CSS3 (35 dagen)***

  Het belangrijkste onderdeel van de afstudeerstage is het ontwikkelen van de webinterface. De kwaliteitseisen van 42 BV staan ook in dit project centraal: het schrijven van code van lage complexiteit, die vrij is van conventieschendingen en grondig is getest. De methode voor het testen van de JavaScript/jQuery code en het inzichtelijk maken van de kwaliteitscriteria zal tijdens de stage ook worden onderzocht.

  Er zal een oriëntatiefase plaatsvinden waarin wordt gekeken of bepaalde delen van de webinterface betreffende de datavisualisatie al (vrij) beschikbaar zijn om te gebruiken, en of deze voldoen aan de eisen van 42 BV. De uitkomsten van deze oriëntatie worden vastgelegd op Confluence.

  Ook zal er worden nagedacht over wat voor informatieve representaties van data nog meer geschikt zijn om te maken en mogelijk zijn om te creëren in een webomgeving om de huidige problemen zo veel mogelijk aan te kunnen pakken. Deze ideeën zullen besproken worden met de stakeholders en indien geschikt geacht worden toegevoegd aan de Scrum backlog voor implementatie.

- ***Het schrijven van een afstudeerverslag (gedurende gehele project, afronding 20 dagen)***

  Gedurende het hele project wordt er tussentijds gewerkt aan het afstudeerverslag dat aan de Hogeschool opgeleverd wordt aan het einde van het afstudeerproject. Aan het einde van de periode zal er tevens tijd worden gereserveerd om de afronding van dit verslag te realiseren.

6. Methodiek

Tijdens het verloop van de afstudeeropdracht wordt er gebruik gemaakt van de ontwikkelmethodiek “Scrum”. Hier zijn enkele redenen voor te benoemen:

1. **Scrum is de bedrijfsstandaard, de medewerkers zijn hieraan gewend**

   42 BV gebruikt sinds jaar en dag Scrum als ontwikkelmethodiek voor haar projecten. Dit betekent dat de processen die hierbij horen welbekend zijn bij de medewerkers van het bedrijf. Gecombineerd met het feit dat ik eveneens ervaring heb met de ontwikkelmethodiek zal dit de communicatie tussen mij en de rest van de betrokkenen bij dit project vergemakkelijken.

2. **Interne applicaties zijn geënt op Scrum**

   42 BV maakt gebruik van diverse applicaties van Atlassian die helpen bij het managen van een project en het bijhouden van de voortgang hiervan. Deze tools zijn geënt op het gebruik van Scrum. 42 BV verlangt dat deze tools gebruikt worden, aangezien deze voor iedere medewerker beschikbaar zijn voor inzage en dus de mogelijkheden bieden om constructieve discussies op te roepen. Het gebruik van Scrum zal hier gemakkelijker bij aansluiten dan de andere ontwikkelmethodieken.

3. **De mogelijkheid om voortschrijdend inzicht mee te nemen in de planningen**

   Scrum biedt de mogelijkheid om voortschrijdend inzicht mee te nemen in de planningen van het project. Het gebeurd in de praktijk regelmatig dat bepaalde zaken die in eerdere fasen van de applicatie-ontwikkeling verstandig leken toch niet helemaal correct zijn. Een voordeel van Scrum is dat deze wijzigingen gemakkelijk meegenomen kunnen worden, vanwege de korte sprints en frequente communicatie tussen de verschillende partijen. De ontwerpfase ligt wat minder vast dan bij andere ontwikkelmethodieken.

4. **Minder overhead**

   Waar het bij andere ontwikkelmethodieken zoals RUP in iedere fase gebruikelijk is om rapporten op te leveren, is dit bij Scrum niet het geval. In plaats daarvan wordt er regelmatig overlegd tussen de verschillende partijen en wordt na iedere sprint gereflecteerd op de werkwijze en voortgang. Dit betekent dat er veel minder tijd in overhead gaat zitten, vanwege de afwezigheid van de noodzaak om diverse omvangrijke rapporten te schrijven. Wel wordt er een backlog bijgehouden waar de voortgang van iedere “user story” wordt bijgehouden, alsmede burndown-charts die de algemene voortgang van de sprint weergeven.

   Er worden gedurende het project sprints van 1 week gebruikt, waarbij aan het einde van iedere sprint een “retrospective” wordt gehouden waarin de voortgang kan worden getoond en bespreekbaar wordt gemaakt. Hierin kan worden besproken wat er goed ging tijdens de afgelopen sprint, en wat punten zijn om te verbeteren. Op deze wijze kan tevens inzichtelijk worden gemaakt of mijn beeld van het op te leveren werk nog klopt met die van de stakeholders, omdat deze retrospectives zo frequent voorkomen is de mogelijkheid voor deze controles vaak aanwezig.

7. Op te leveren (tussen)producten

Tijdens de afstudeerstage zal er (ten minste) het volgende opgeleverd worden:

- Een Plan van Aanpak
- Een overzicht van de verschillende requirements
- Een Java-backend
- Een webinterface met HTML5, CSS3, Javascript en jQuery.
- Scrum backlogs in Jira
- Scrum user stories in Jira
- Burn-down charts in Jira
- Bugs/Issues in Jira
- UML diagrammen:
    - Klassediagram
    - Usecase diagrammen met beschrijvingen
    - Sequentiediagrammen
- Documentatie in Confluence
    - Bevindingen databaseanalyse Mus met domeinmodel
    - Bevindingen oriëntatie webontwikkelstraat
    - Bevindingen oriëntatie bestaande methoden datavisualisatie
    - Overzicht kritieke succesfactoren
    - Overige informatie nuttig voor de overdracht na afronding van het project
- Afstudeerverslag aan de Haagse Hogeschool

8. Te demonstreren competenties en wijze waarop

- **Selecteren methoden, technieken en tools**

Selecteren en adviseren over ontwikkeltools die een onderdeel (gaan) vormen van de ontwikkel-, beheer-, test- en productieomgeving.

Tijdens de afstudeeropdracht wordt er gekeken naar de verschillende tools die het mogelijk maken om een ontwikkel-, beheer- test en productieomgeving voor JavaScript/jQuery code. Indien er genoeg tijd beschikbaar is zal deze ook worden opgezet.

- **Uitvoeren analyse door definitie van requirements**

Inventariseren, analyseren en beschrijven van de gewenste informatievoorziening, zodat wordt vastgelegd welke processen objecten creëren en gebruiken.

Er worden usecases in kaart gebracht die de belangrijkste functies van de webapplicatie in kaart zullen brengen. Dit zal worden gedaan op basis van de requirements die tevens in kaart moeten worden gebracht. De grove lijnen hiervan zijn al uitgezet, maar deze kunnen nog veel verder gespecificeerd worden.

- **Ontwerpen systeemdeel**

Beschrijven van systeemdelen (subsystemen, componenten, modules), hun onderliggende structuur en het gedrag in detail, zodanig dat bouwen van het systeemdeel mogelijk is.

Bij het ontwerpen van de Java-backend moet er rekening mee gehouden worden dat er gecommuniceerd moet worden met bestaande modules binnen de koepelapplicatie Volière alsmede met de nieuw te realiseren webapplicatie. Er moet dus van tevoren nagedacht en gedocumenteerd worden hoe dit gerealiseerd kan worden.

- **Bouwen applicatie**

Bouwen en documenteren van de systeemdelen.

Na afrondingen van de ontwerpfasen dient de webinterface en de bijbehorende Java-backend uiteraard ook gebouwd te worden. Hierover wordt documentatie opgeleverd in de corporate wiki van 42, Confluence, en tevens wordt de technische voortgang vastgelegd in Jira, het projectmanagement en issue-tracking systeem dat door 42 gebruikt wordt.

- **Uitvoeren van en rapporteren over het testproces**

Uitvoeren testplan en opstellen rapportage

De code die geschreven wordt moet uiteraard ook getest worden. Hiervoor zullen ten minste unit-tests geschreven worden, mogelijk nog aangevuld met webinterfacetesten als Selenium. De Java-testcode wordt met behulp van de build automation tool Maven in kaart gebracht. Dit levert een duidelijk overzicht op van de mate waarop de code getest is. De resultaten van alle testcases worden gedocumenteerd in een verslag.

P. Plan van Aanpak

Plan van Aanpak

Afstudeerstage “Mees”

Naam: Sander Benschop
Studentnummer: 08007381
Afstudeerbedrijf: 42 BV

1 Inhoudsopgave

2 Inleiding

In dit plan van aanpak zal ik beschrijven wat mijn strategie wordt om tot een succesvolle afronding van mijn afstudeerstageproject te komen. Ik zal dit doen middels de volgende hoofdstukken:

Opdrachtomschrijving
Hierin zal het bedrijf beknopt omschreven worden, en de problemen die bestaan in de huidige situatie beschreven. Er zal worden uitgelegd hoe het idee voor dit project tot stand is gekomen.

Afbakening opdracht
Hierin zullen de grenzen van het project worden afgebakend. Er zal worden beschreven of er nog dingen zijn die de opdrachtgever specifiek niet in het component wilt hebben.

Randvoorwaarden
Hierin zal beschreven worden wat voor eisen ik stel aan mijn stagebegeleider/opdrachtgever opdat ik het project succesvol kan voltooien en waar hij aan moet voldoen voordat ik het project kan starten.

Risicofactoren
Hierin zal worden bekeken wat voor risico's en knelpunten er gedurende het project verwacht kunnen worden, hoe je zo'n knelpunt tijdig kan herkennen (of nog beter: voorkomen), wat de gevolgen zouden kunnen zjn van zo'n knelpunt en hoe kan de schade beperkt worden mocht het toch voorkomen.

Rapportagewijze
Hierin wordt beschreven op wat voor wijze er verslag zal worden gedaan aan de opdrachtgever, waarover ik verslag ga doen, hoe frequent ik dit ga doen en wat voor vorm het verslag heeft.

Benodigde mensen/middelen
Hierin zal worden beschreven wat voor mensen en middelen ik denk nodig te hebben om het project succesvol af te ronden.

Planning
Hierin zal ik beschrijven wat voor activiteiten er zijn, wanneer ik hieraan begin en wanneer ik dit verwacht af te ronden.

3 Opdrachtomschrijving

42 BV heeft op dit moment al diverse interne applicaties die vorig jaar verzameld zijn onder de paraplu-applicatie “Volière”. Onderdelen hiervan is bijvoorbeeld het registratiesysteem “Mus”. Hierin worden de gewerkte en geplande uren geregistreerd en vervolgens opgeslagen in een database. Deze gegevens wil 42 BV gaan gebruiken om een tool te creëren die de managers en de salesafdeling van informatie kunnen voorzien om betere beslissingen te maken wat betreft medewerkersinzet bij projecten.

De salesafdeling heeft op dit moment problemen met het efficiënt inzetten van medewerkers op projecten, wat in het verleden geresulteerd heeft in het feit dat medewerkers “op de bank” terecht kwamen. Voor het management is het daarentegen lastig om de medewerkers binnen lopende projecten efficiënt in te zetten. In beide situaties is het gebrek aan gecentraliseerde, duidelijke actuele en historische data het probleem.

Deze opdracht valt grofweg uiteen in de volgende onderdelen:

Plan van aanpak schrijven
De eerste stap van de afstudeerstage zal het schrijven van een plan van aanpak zijn, waarin ik duidelijk schets wat de route is die ik denk te gaan bewandelen gedurende mijn afstudeeropdracht. Hierin zal onder andere een globale planning terugkomen en een overzicht van de verwachte impediments van het project.

Requirements opstellen
Na afronding van het plan van aanpak wordt begonnen aan het inventariseren van de wensen van de (hoofd)gebruikers. Los van de functionele requirements zullen hier eveneens de kwaliteitsrequirements in terugkomen. De eerdere gesprekken over de kritieke succesfactoren zullen eveneens worden meegenomen in deze inventarisatie. Alle informatie samen wordt vervolgens gebruikt om requirements mee op te stellen die direct gebruikt wordt vanaf de ontwerpfase.

Analyseren data uit Mus-database
Voordat er begonnen zal worden met het modelleren van de te creëren webapplicatie, wordt eerst goed de tijd genomen voor het analyseren van de data uit de Mus database. Deze tijd zal worden gebruikt om kennis te vergaren van de vorm en inhoud van de aanwezige data, zodat er een beeld ontstaat van de informatie die benodigd is voor de webapplicatie. Een domeinmodel kan worden gegenereerd vanuit het RDBMS om een duidelijk, schematisch beeld te krijgen van de databasestructuur.

Opstellen (data-)ontwerpen
Alvorens er begonnen kan worden aan het programmeren van de webapplicatie of de Java backend worden eerst de intiële opzet van de UML-diagrammen gemaakt. Deze zullen als hulpmiddel worden gebruikt tijdens de verdere ontwikkelingen. In ieder geval word(en) een klassediagram, usecasediagrammen (met beschrijvingen) en sequentiediagrammen gecreëerd. Een combinatie van deze diagrammen geeft een voldoende beeld over de wensen aan- en structuur van- de webapplicatie en de Java backend om deze te kunnen ontwikkelen.

Na opzet van de initiële ontwerpdocumenten zal worden begonnen aan de ontwikkeling van de webapplicatie. Tijdens deze ontwikkelingen wordt meer cachet gegeven aan de verschillende modellen en worden verdere modellen die de ontwikkelingen van de verschillende functionaliteiten vergemakkelijken iteratief toegevoegd, gezamenlijk met de implementatie hiervan. Deze iteratieve aanpak helpt enerzijds om duidelijk na te denken over het ontwerp alvorens er begonnen wordt met programmeren, en biedt anderzijds ook de mogelijkheid om voortschrijdende kennis mee te nemen in het ontwerp van de webapplicatie.

Ontwikkelen Java backend
Na afronding van de initiële ontwerpfase wordt begonnen met het ontwikkelen. Allereerst zal een opzet voor de backend worden gemaakt zodat er vanuit de webinterface tegenaan kan worden gesproken. Voor iedere opvolgende functionaliteit zal na het maken van nieuwe ontwerpen in principe eerst het backend gedeelte worden ontwikkeld, gevolgd door het webinterface gedeelte. Deze iteratieve aanpak helpt met het verifiëren dat iedere losstaande functionaliteit doet wat er van verwacht wordt, wat de uiteindelijke koppeling tussen de functionaliteiten zal vergemakkelijken.

Aanvankelijk was het plan om een Java-backend op te zetten die enerzijds via REST-calls communiceert met de op te zetten webinterface en anderzijds de communicatie legt met de Mus database via de Java-module “Postduif” van de volière. Echter, tijdens het selecteren van technologieën heb ik van diverse mensen te horen gekregen dat de methodiek die gebruikt werd in een andere Volière-applicatie, Snip, niet volledig voldeed aan de verwachtingen. Deze applicatie beschikte over een eigen Groovy-backend evenals backend-functionaliteit in Postduif. Omdat er hier in feite sprake was van 2 backend-lagen creëerde dit telkens wat frictie bij het toevoegen van nieuwe functionaliteiten. Voor mijn afstudeeropdracht wil ik een applicatie neerleggen die gemakkelijk te onderhouden is, en aangezien de berekeningen die worden uitgevoerd gemakkelijk óf in de JavaScript-code bij de client óf in Postduif kan worden uitgevoerd, heb ik besloten om de backend-functionaliteit in Postduif op te nemen in plaats van als losse module te schrijven. Ik zal meer toelichting hierop geven in mijn afstudeerverslag.

Postduif is verantwoordelijk voor alle communicatie tussen de verschillende applicaties van 42 en de Mus database. In de backend-uitbreiding hieraan wordt de juiste data opgehaald uit de Mus-database en na het uitvoeren van enkele berekeningen teruggegeven aan de JavaScript-module.

De backend-functionaliteit zal geprogrammeerd worden middels Test Driven Development om te voldoen aan de kwaliteitseisen van 42 BV op het gebied van software testing.

Ontwikkelen webinterface
Het belangrijkste onderdeel van de afstudeerstage is het ontwikkelen van de webinterface. De kwaliteitseisen van 42 BV staan ook in dit project centraal: het schrijven van code van lage complexiteit, die vrij is van conventieschendingen en grondig is getest. De methode voor het testen van de JavaScript/jQuery code en het inzichtelijk maken van de kwaliteitscriteria zal tijdens de stage ook worden onderzocht.

Er zal een oriëntatiefase plaatsvinden waarin wordt gekeken of bepaalde delen van de webinterface betreffende de datavisualisatie al (vrij) beschikbaar zijn om te gebruiken, en of deze voldoen aan de eisen van 42 BV. De uitkomsten van deze oriëntatie worden vastgelegd op Confluence.

Ook zal er worden nagedacht over wat voor informatieve representaties van data nog meer geschikt zijn om te maken en mogelijk zijn om te creëren in een webomgeving om de huidige problemen zo veel mogelijk aan te kunnen pakken. Deze ideeën zullen besproken worden met de stakeholders en indien geschikt geacht worden toegevoegd aan de Scrum backlog voor implementatie.

Schrijven afstudeerverslag
Gedurende het hele project wordt er tussentijds gewerkt aan het afstudeerverslag dat aan de Hogeschool opgeleverd wordt aan het einde van het afstudeerproject. Aan het einde van de periode zal er tevens tijd worden gereserveerd om de afronding van dit verslag te realiseren.

4 Ontwikkelmethodiek en -omgeving

Ik zal hierbij gebruik maken van de ontwikkelmethode Scrum. Dagelijks zal er een stand-up meeting plaatsvinden met enkele andere medewerkers van 42 BV, iedereen zal over zijn/haar respectievelijke projecten vertellen.
Tijdens deze meetings zal besproken worden wat er de vorige werkdag is gedaan, wat de impediments waren en wat er op de planning staat voor vandaag.
Er zal gebruik worden gemaakt van Jira, een project tracking en issue management tool ontwikkeld door het bedrijf Atlassian, om het verloop van het project in bij te houden. Alle toe te voegen functionaliteiten zullen worden toegevoegd als user stories, en gevonden bugs kunnen hier ook in worden geregistreerd. In Jira wordt de progress van deze issues bijgehouden.
Voor iedere sprint zal een planning worden opgesteld waarin een aantal user stories worden behandeld. De hoeveelheid te behandelen stories wordt bepaald aan de hand van mijn velocity, deze kan na iedere sprint worden bijgesteld om een realistischer schatting te krijgen van de hoeveelheid werk waar ik in staat toe ben om te verzetten gedurende één sprint.
Voor het versiebeheer van het project zal gebruik worden gemaakt van de subversion-server van 42 BV.

5 Afbakening opdracht

- **Het muteren van de Mus database**

  Omdat Mus traditioneel enkel is gebouwd voor het bijhouden van urenregistraties en niet voor analysesystemen als deze, sluit deze op bepaalde punten niet aan. Voorbeelden zijn bijvoorbeeld het feit dat alles in Mus een project is, van ziekte, Human Resource Manager tot administratie. Er is dus ook geen onderscheid tussen lijnactiviteiten en projecten gemaakt. Er zouden wel bepaalde bitwaarden aanwezig zijn in de database die hierbij helpen.

  Hoewel het uiteraard geen probleem is om advies uit te brengen over de wijzigingen die een betere analyse van de data mogelijk maken, valt het aanbrengen van de daadwerkelijke wijzigingen met de mogelijke conversies die hierbij betrokken zijn niet onder mijn opdracht.

- **Geen aanpassing architectuur**

  De Volière architectuur zal niet worden aangepast tijdens dit project, enkel de oude opzet van de module “Mees” zal worden vervangen door de nieuw te realiseren versie die vanaf de grond wordt opgebouwd. Dit betekend ook dat er niet om de Postduif module heengewerkt mag worden: deze moet worden gebruikt voor databasecommunicatie.

6 Randvoorwaarden

Om dit project tot een geslaagd einde te brengen zijn er enkele voorwaarden waar ik en 42 BV aan moeten voldoen.

De afstudeerder

- Afgesproken is dat ik 5 dagen per week aanwezig zal zijn voor een periode van 17 weken (85 werkdagen).

- In die periode zal ik mij beperken met werkzaamheden betreffende Mees, en zal er niet gewerkt worden aan andere projecten waar ik bij betrokken ben geweest in het verleden.

42 BV

- Het bedrijf 42 BV ondersteunt mij in het afronden van mijn afstudeerstage door middel van scrum meetings en voortgangsgesprekken.

- Van de verschillende stakeholders wordt ook verwacht dat zij actief meedenken over de op te leveren producten.

7 Rollen

Gedurende het project worden er door diverse medewerkers van 42 rollen vervuld. In dit hoofdstuk wordt besproken welke rollen er geïdentificeerd zijn en door wie deze vervuld zullen worden. Ook zal per rol worden uitgelegd wat 42 BV erover verstaat.

Medewerker	*Scrum-rol*	*Reguliere functie*
Harko Ratsma	Project lead, Product owner	Project Manager
Eric Meijer	Product Owner	CEO
Robert Bor	Technical lead	Chief Technology Officer
Ron Oudshoorn	Sales Representative	Manager Marketing en Sales
Sander Benschop	Scrum Master, Developer	Software Developer

Project lead

De project lead bewaakt de voortgang van het project en kiest samen met de developer wat er in de komende sprint uitgevoerd zal worden. Aanvankelijk zal aan hem aan het begin van de week worden getoond worden wat er de afgelopen week is gecreëerd. De product owner is echter verantwoordelijk voor de acceptatie van de applicatie.

Product owner
De product owner beslist wat er gebouwd gaat worden en in welke volgorde, hij definieert de features of gewenste uitkomst van het project. Hij kiest welke content wanneer wordt gereleast. Hij geeft prioriteit aan user stories aan de hand van de waarde van de features. Hij is tevens degene die werkresultaten accepteert of weigert.

Technical lead
De technical lead is verantwoordelijk om samen met de developer beslissingen te nemen over de technologien die gebruikt worden in de applicatie.

Sales Representative
De Sales Representative is vertegenwoordiger van de afdeling Sales en dus ook een van de beoogde gebruikersgroepen, met hem zal worden gesproken over zijn wensen aan het product. Aan hem worden de gecreëerde functionaliteiten ook getoond, maar hij is niet verantwoordelijk voor de acceptatie. Dit is de product owner.

Scrum Master
De scrum master is verantwoordelijk voor de productiviteit van het ontwikkelteam, in dit geval is dat dus enkel ikzelf. Verder verzorgt hij de communicatie met de rest van de stakeholders en verzekert dat de scrumprocessen gevolgd worden.

Developer
De developer is verantwoordelijk voor de uitvoer van het ontwikkelproces: in dit geval vanaf het opstellen van requirements, het maken van ontwerpen tot het schrijven van de programmacode. Belangrijke technologische beslissingen maakt hij samen met de Technical Lead.

8 Risicofactoren

Binnen dit project zijn enkele risicofactoren te benoemen, zowel voor het slagen van het project naar tevredenheid van 42 BV als mijn functioneren als afstudeerder. Bij het laatste punt heeft het feit dat ik al langere tijd werkzaam ben bij 42 ook invloed. In dit hoofdstuk zal ik beschrijven welke issues ik heb geïdentificeerd en hoe ik deze aan ga pakken.

- **Uitvoeren operationeel werk**
  Omdat ik al langere tijd werkzaam ben bij 42 BV werd er vanuit de Hogeschool de zorg geuit dat mij gevraagd zou kunnen worden om operationeel werk te verrichten aan mijn afgelopen project: JaRB. Een andere stagiair van de Haagse Hogeschool, Pim van Dongen, is hier nu mee aan de slag.

  Afgesproken is dat ik gedurende mijn afstudeerstage geen werk zal verrichten aan JaRB, en om te voorkomen dat hij teveel vragen over het project stelt is besloten om mij in een andere projectruimte te zetten. Enerzijds biedt dit het voordeel dat hij zelf dieper in de materie moet duiken voordat er vragen gesteld worden, en anderzijds kost mij dit een stuk minder tijd. Mochten er toch vragen zijn dan worden deze gebundeld bij mij ingeleverd en worden deze behandeld wanneer ik hier tijd voor heb.

- **Documentatie overhead**
  Vanuit de hogeschool wordt om te bewijzen dat ik aan het einde van de rit aan alle competenties voldoe gevraagd om diverse UML-modellen. Hoewel dit uiteraard erg nuttig is kan hier ook een hoop tijd in gaan zitten, en is het bovendien zo dat latere inzichten wijzigingen in het ontwerp met zich mee zullen brengen.

  Omdat hier binnen de gebruikte methodiek Scrum voldoende ruimte voor is zal ook het ontwerpen van de modellen hier op moeten aansluiten in plaats van aanvankelijk alle modellen maken, wat zeer tijdsintensief is en tevens een moeilijke fase om de beoogde webapplicatie en de omliggende architectuur te kunnen bevatten, er initiële modellen worden opgesteld die de grote lijnen van het project uitzetten, en zullen vervolgens incrementeel met de code modellen worden opgeleverd. Dit zorgt voor een recenter en kloppender beeld.

- **Foute ontwerpkeuzes door voldoende kennis Mus / Postduif**
  Als ik me niet voldoende inlees over de structuur en inhoud van de Mus database, en de Postduif-module voor communicatie tussen deze database en Mees, loop ik het risico dat ik verkeerde beslissingen ga nemen bij het ontwerpen van Mees.

  Om dit te voorkomen is het cruciaal dat ik in de beginfase van het project me goed oriënteer op deze systemen. Ook zal er met collega Theo van Essen worden afgesproken waarin ik de mogelijkheid heb om hem vragen te stellen over de oude opzet van Mees die hij gecreëerd heeft en wat voor bevindingen hij hierbij heeft gehad.

9 Rapportagewijze

Zoals eerder in dit document beschreven wordt tijdens het verloop van het project gebruik gemaakt van Scrum als ontwikkelmethodiek, met wekelijkse standups.

Dit heeft als gevolg dat alsmede de dagelijkse standups met de overige ontwikkelaars van 42 BV er wekelijks een demo zal worden gegeven aan de opdrachtgever en de nieuwe activiteiten van die week zullen worden besproken. Door deze korte periodes tussen meetings is de kans kleiner dat er verschil in interpretatie van requirements zal ontstaan tussen de opdrachtgever en mij.

De schriftelijke rapportage van de backlogitems zal op Jira gebeuren. Verder worden bepaalde “lessons learned” en overige informatie over Mees die nuttig kan zijn voor anderen op Confluence geplaatst.

10 Benodigde mensen/middelen

Om mijn afstudeerstage tot een succes te brengen heb ik enkele mensen en middelen nodig. Ik zal hierbij beginnen met welke mensen benodigd zijn:

- Mijn bedrijfsmentor en product owner: Harko Ratsma.
- Mijn examinatoren: Arno Nederend en Nanny Jacobs
- De CEO van 42 BV: Eric Meijer. Hij is een van de toekomstige gebruikers.
- De sales representative: Ron Oudshoorn

De middelen die ik hiervoor nodig heb zijn:

- Een computer geschikt voor het programmeren van Java(script)-applicaties met internetverbinding.
- IntelliJ IDEA, een IDE om applicaties in te ontwikkelen.
- Jira, een bugdatabase en Scrum planningsysteem.
- Sonar, een controleprogramma voor codekwaliteit van software.
- Subversion, om het versiebeheer van mijn project mee uit te voeren.
- Toegang tot de broncode van de bestaande applicaties en Mus database.

11 Planning

Alvorens er begonnen kan worden met het ontwikkelen van het daadwerkelijke product zal er eerst een te bewandelen route gekozen moeten worden over de specifieke vorm en technische werking ervan. Hier zijn een paar ideeën over (alle data in een keer ophalen uit database, data parkeren in geprepareerde vorm, etc) waaruit gekozen moet worden. Om dit te kunnen doen zal ik beginnen met het maken van een paar simpele proof-of-concepts waarmee we de performance en user-friendliness kunnen analyseren. Aan de hand daarvan zal er een keuze worden gemaakt over de te bewandelen route.

Om het Scrum principe van "Adding value" vast te kunnen houden zal na de ontwikkeling van de proof-of-concepts begonnen worden met het maken van een simpele webpagina met data uit Mus die inzichten geeft over de uren waarop medewerkers zijn ingezet bij projecten. Dit zal aanvankelijk in een simpele, statische HTML pagina worden gedaan.

Deze pagina kan direct al worden gebruikt door de Salesvertegenwoordiger als vervanging van de Excel sheets die hij nu handmatig bijhoud. Op deze wijze wordt er al in een vroeg stadium waarde aan het project toegevoegd, wat het draagvlak voor de applicatie onder de medewerkers van het bedrijf sterker zal maken.

Hierna zal de webapplicatie verder worden uitgebouwd om te kunnen voldoen aan de overige eisen van de stakeholders. Er zal wekelijks worden beslist wat de op te leveren user story's zullen zijn.

Week	Periode	Activiteit
1	6 februari – 10 februari	Inlezen technieken, oriënteren op bestaande architectuur en database.
2	13 februari - 17 februari	Proof of concepts ontwikkelen
3	20 februari – 24 februari	Requirements opstellen en initiële ontwerpen maken. Analyseren benodigde MUS-data.
4 t/m 14	27 februari – 4 mei	Incrementeel ontwerpen en ontwikkelen backend en front-end
15 t/m 17	7 mei – 1 juni	Afronden afstudeerverslag

Q. Formulier conceptbespreking afstudeerverslag

Formulier bespreking concept afstudeerdossier

Student: Sander Benschop **Studentnummer:** 08007381

Datum: 10 april 2012

Tijdens de bespreking is het volgende geconstateerd:		**ja**	**nee**
a	*Het voortgangsverslag is ontvangen*	X	
b	*Het afstudeerdossier is digitaal beschikbaar*		X
c	*Het afstudeerdossier is opgebouwd conform de richtlijnen*	X	
d	*Het goedgekeurde afstudeerplan is aanwezig*	X	
e	*Het plan van aanpak is aanwezig*	X	
f	*Reeds geleverd commentaar is aanwezig*	n.v.t.	
g	*Het afstudeerdossier geeft voldoende inzicht in de stand van zaken*	X	
h	*De afstudeeropdracht is tot nu toe naar behoren uitgevoerd*	X	

Verbeterpunten:

Laat de systeemontwikkelingsmethodiek SCRUM beter terugkomen; gebruik 'professionele' begrippen bij het aanduiden van kwaliteitsattributen van software; stel duidelijke criteria op bij keuzemomenten van b.v. software, frameworks of architectonische beslissingen.

Opmerkingen:

Zorg dat het afstudeerdossier digitaal beschikbaar is voor de beide examinatoren.

Naam begeleidend examinator: Arno Nederend

Datum: 10 april 2012

Dit formulier wordt door de begeleidend examinator digitaal ingevuld en per email naar de student verstuurd met een cc naar de coördinator van ICT & Media @ Work (A.M.Schipper@hhs.nl). Het formulier dient door de student te worden opgenomen in het afstudeerdossier.

R. Formulier tussentijds assessment

Formulier tussentijds assessment

Student: Sander Benschop **Studentnummer:** 08007381

Datum: 14 mei 2012 **eerste / tweede TTA:** eerste

Tijdens het tussentijds assessment is het volgende geconstateerd:		**ja**	**nee**
a	*Het voortgangsverslag is ontvangen*	x	
b	*Het afstudeerdossier is digitaal beschikbaar*	x	
c	*Het afstudeerdossier is opgebouwd conform de richtlijnen*	x	
d	*Het goedgekeurde afstudeerplan is aanwezig*	x	
e	*Het plan van aanpak is aanwezig*	x	
f	*Reeds geleverd commentaar is aanwezig*	x	
g	*Het afstudeerdossier geeft voldoende inzicht in de stand van zaken*	x	
h	*De afstudeeropdracht is tot nu toe naar behoren uitgevoerd*	x	

Aanpak	**O**	**T**	**V**	**G**
Passend			x	
Theoretisch verantwoord			x	
Samenhang uitvoering beroepstaken			x	

Beroepstaken op afgesproken niveau uitgevoerd?		**O**	**T**	**V**	**G**
1	Selecteren methoden, technieken en tools			x	
2	Uitvoeren analyse door definitie van requirements			x	
3	Ontwerpen systeemdeel			x	
4	Bouwen applicatie			x	
5	Uitvoeren van en rapporteren over het testproces			x	

Producten	O	T	V	G
Tussenproducten			x	
Eindproducten			x	

Effectief communiceren	O	T	V	G
Binnen afstudeerbedrijf			x	
Afstudeerdossier			x	

Reflectie	O	T	V	G
Inzicht in eigen functioneren			x	
Inzicht in eigen leerproces			x	

Toelichting per beoordelingscriterium

Aanpak
Student werkt conform planning in Plan van Aanpak en weet bij grote verandering het proces opnieuw in te richten

Beroepstaken op afgesproken niveau uitgevoerd?
De beroepstaken zijn conform afspraak uit gevoerd. Het uitvoeren van het testproces is nog wat onderbelicht.

Producten
De opgeleverde producten zijn van voldoende niveau. De onderbouwing van het tot stand komen van de producten en de bijbehorende verantwoording kan beter.

Effectief communiceren
Student heeft examinatoren met de juiste regelmaat op de hoogte gehouden. Uit zijn afstudeerdossier blijkt ook dat de student de stakeholders in de organisatie voldoende heeft betrokken bij zijn project

Reflectie
Uit zijn evaluatie van product en proces blijkt dat de student in voldoende mate de beschikking heeft over reflectievermogen

Advies

x	Positief **(bindend advies)**
	Verlengen **(vrijblijvend advies)**
	Negatief **(vrijblijvend advies)**

Besluit student

Aankruisen welke beslissing de student heeft genomen (alleen na vrijblijvend advies)

	Afstudeerdossier wordt op afgesproken datum ingeleverd	Inleverdatum: 1 juni 2012
	Afstudeerperiode wordt verlengd	Inleverdatum:
	Student stopt met afstudeeropdracht	

Naam begeleidend examinator: Arno Nederend

Naam tweede examinator: Nanny Jacobs

Datum: 18 mei 2012

Dit formulier wordt door de tweede examinator digitaal ingevuld, waarna de begeleidend examinator het per email verstuurt naar de student met een cc naar de coördinator van ICT & Media @ Work (A.M.Schipper@hhs.nl). Het formulier dient door de student te worden opgenomen in het afstudeerdossier.

S. Evaluatieformulier

DE HAAGSE HOGESCHOOL

Academie voor ICT & Media

Den Haag
Delft
Zoetermeer

Evaluatieformulier afstuderen
In te vullen door opdrachtgever c.q. bedrijfsmentor(en)

Student: Sander Benschop

Periode: 6 februari 2012 – 1 juni 2012

Bedrijf c.q. instelling: 42 BV

Bedrijfsmentor: Harko Ratsma

Plaats: Zoetermeer

Datum: 31-05-2012

1. **Heeft de student zich zelf snel en goed ingewerkt in het bedrijf en de uit te voeren afstudeeropdracht?**
   Sander heeft zich goed in de opdracht verdiept. Hij had al een ander project bij 42 gedaan, en wist dus zijn weg te vinden. Bij deze opdracht kon hij dus snel tot de kern komen.

2. **Hoe beoordeelt u de communicatieve vaardigheden van de student (in de samenwerking met collega's, in contacten met de opdrachtgever, bij mondelinge presentaties, schriftelijke rapportages)?**
   De communicatieve vaardigheden zijn voor een ontwikkelaar buitengewoon goed ontwikkeld. Presentaties zijn geen probleem. Ook de presentatie voor de gehele bedrijf, tijdens een KTM (Knowledge Transfer Meeting) ging prima. Lastige vragen van irritante collega's werden charmant gepareerd. Hij hield prima stand.

3. **Hoe heeft de student tijdens het uitvoeren van de opdracht gefunctioneerd?**

- **Qua verantwoordelijkheid** goed ~~/ voldoende / matig / onvoldoende~~
- **Qua zelfstandigheid** goed ~~/ voldoende / matig / onvoldoende~~
- **Qua planmatig werken** goed ~~/ voldoende / matig / onvoldoende~~
- **Qua creativiteit** ~~goed~~ / voldoende ~~/ matig / onvoldoende~~
- **Qua productiviteit** goed ~~/ voldoende / matig / onvoldoende~~
- **Qua samenwerken met collega's** goed / ~~voldoende / matig / onvoldoende~~
- **Qua draagvlakontwikkeling** ~~goed /~~ voldoende / ~~matig / onvoldoende~~
- **Qua inspelen op bedrijfscultuur** ~~goed~~ / voldoende / ~~matig / onvoldoende~~
- **Qua rekening houden met de specifieke context van het bedrijf** goed / ~~voldoende / matig / onvoldoende~~
- **Qua het op gang brengen van de nodige veranderingen** goed ~~/ voldoende / matig / onvoldoende~~

4. **Hoe beoordeelt u de kennis en kunde van de student in verhouding tot wat u verwacht van een bijna afgestudeerde?**

Sander is zelfstandiger dan ik van een student zou verwachten. Hij is eigenwijs in de goede zin van het woord. Mijn pogingen om hem ver weg te houden van eindgebruikers zijn jammerlijk gefaald, wat uitermate voor hem spreekt. Hij is ook geen software engineer pur sang, daarvoor communiceert hij te goed. Ik verwacht dat hij zich prima zal ontwikkelen op het snijvlak van business en techniek.

5. Hoe beoordeelt u de kwaliteit van de opgeleverde (tussen)producten?

De applicatie is goed bruikbaar, de technische uitdagingen die onderweg opdoken zijn alle prima opgelost. Ook bij de ontwikkeling van andere applicaties die we gaan ontwikkelen op deze stack zullen we baat hebben bij dit project. Tijdens het project heeft al kruisbestuiving plaatsgevonden met andere projecten.

6. Bent u tevreden over het opgeleverde (eind)product?

- **In hoeverre heeft u gekregen wat is afgesproken?**

We zijn begonnen met een iets ander uitgangspunt, een spectaculaire UI. Onderweg bleken er infrastructurele impediments te zijn die we eerst op moesten ruimen. Hoewel de uitkomst anders is dan initieel afgesproken, is de uiteindelijke waarde groter omdat we voor andere projecten zaken uitgezocht, ontworpen en geimplementeerd hebben.

- **In hoeverre voldoet het (eind)product aan uw verwachtingen?**

Omdat door de goede communicatie van Sander verwachtingen steeds bijgesteld zijn, voldoet het eindproduct volkomen. Het gekozen agile proces is door Sander ten volle benut, er is zo snel mogelijk feedback gevraagd en gegeven. Geen verrassingen dus.

- **Wat is de bruikbaarheid en onderhoudbaarheid hiervan?**

De bruikbaarheid is groot, er is nu realtime beschikbaarheids informatie die de sales effort direct kan sturen. De structuur van de code is zodanig dat eventuele aanpassingen snel doorgevoerd kunnen worden. Dit is in de praktijk bewezen omdat additionele eisen in een laat stadium snel doorgevoerd werden.

- **Wat gebeurt er met het opgeleverde (eind)product?**

Dat gaat in productie gebruikt worden door onze sales.

- **Kunt u direct met het opgeleverde product aan de slag?**

Ja

7. **Zijn er nog aspecten voor u van belang die nog niet aan de orde zijn geweest?**

Ik heb bijzonder prettig met Sander gewerkt. Ik heb mijn uiterste best gedaan om me als een echte klant op te stellen, met wisselende eisen, nieuwe inzichten en de vereiste mate van onredelijkheid. Sander blijft vriendelijk luisteren en weet mij uiteindelijk tevreden te stellen. Prima.

Sander kan soms nog wat onzeker overkomen, hij mag meer zijn successen claimen en zichzelf op de voorgrond stellen. Ik ben er echter van overtuigd dat dit goed gaat komen.

8. **Bent u bereid een volgende keer weer uw medewerking te verlenen aan het beschikbaar stellen van een afstudeerplaats** (graag met toelichting)**?**

Zonder enig probleem. Het is leuk, het heeft direct waarde voor het bedrijf en kan bovendien ook nog eens een nieuwe medewerker opleveren.