Afstudeerverslag

Ontwikkelen van een formulieren module binnen Toolbox WMS bij Eleven

Titel	Ontwikkelen van een formulieren module binnen Toolbox WMS bij Eleven
Student	Paul de Raaij (09081852)
Opdrachtgever	Eleven te Maasdijk
Datum	maandag 2 januari 2012 te Honselersdijk

1. Voorwoord

In de afgelopen vier maanden heb ik bij Eleven te Maasdijk mijn afstudeeropdracht in het kader van mijn deeltijdstudie Informatica aan de Haagse Hogeschool gelopen.

De afstudeeropdracht bestond uit het ontwikkelen van een formulierenmodule voor het Web Management Systeem Toolbox welke ontwikkeld is door Eleven. Deze nieuwe module draagt er zorg voor dat websitebeheerders zelf formulieren kunnen opstellen en toepassen binnen de eigen website voor een variëteit aan doelen.

In de voor u liggende scriptie beschrijf ik het verloop van het proces en geef ik een inzage in het uiteindelijke resultaat. Het document bevat technische termen die nader toegelicht worden in het hoofdstuk 'Woordenlijst'.

Mijn dank gaat uit naar Carlo Vollebregt en Peter Boon voor hun begeleiding en ondersteuning namens Eleven tijdens het uitvoeren van het project. Mede door hun feedback en ideeën heb ik het gewenste eindresultaat op kunnen leveren. Verder bedank ik Paul Smit, Kees Bakker en Rianne Béchet voor de begeleiding en ondersteuning namens de Haagse Hogeschool. Tot slot wil ik mijn familie, vrienden en kennissen bedanken voor hun steun en motiverende woorden tijdens het afstudeertraject.

Ik wens u veel leesplezier.

Paul de Raaij,
Honselersdijk, 2 januari 2012

Inhoudsopgave

1. Voorwoord 2
2. Inleiding 6
3. De afstudeeropdracht 7
   3.1 Probleemstelling 7
   3.2 Doelstelling 7
   3.3 De methode Scrum 8
4. Eleven 9
   4.1 Het bedrijf 9
   4.2 Bedrijfsstructuur Eleven 9
   4.3 Mijn plek binnen Eleven 10
   4.4 De projectorganisatie 11
5. Toolbox omgeving 12
   5.1 Spring Framework 12
   5.2 Hibernate ORM 13
   5.3 Ehcache 13
6. Projectmethode: Scrum 14
   6.1 Agile 14
      6.1.1 Iteraties 14
      6.1.2 Communicatie 14
      6.1.3 Voortgang 14
   6.2 De methode scrum 14
   6.3 Technieken en hulpmiddelen 15
      6.3.1 Daily Stand Up 15
      6.3.2 Sprint Planning meeting 16
      6.3.3 Sprint Retrospective meeting 16
      6.3.4 Story points 16
      6.3.5 Burndown chart 17
   6.4 Waarom dan Scrum? 18
7. Sprint 0 19
   7.1 Plan van aanpak 19
      7.1.1 Globale planning 19
      7.1.2 Definition of Done 19
   7.2 Requirements verzamelen 20
   7.3 Master Test Plan 21
8. Sprint 1 22

8.1 Sprint planning meeting 22
8.2 Ontwerpen domeinmodel 23
8.3 Ontwerpen datamodel 25
8.4 User stories uitwerken 26
8.5 De Toolbox user interface 27
8.6 Selenium tests 28
8.7 Wat ging er goed 31
8.8 Wat ging er niet goed 32
9. Sprint 2 33
9.1 Spring planning meeting 33
9.2 Versiebeheer van formulieren 33
9.2.1 Hibernate Envers 33
9.2.2 Globale werkwijze van Envers 34
9.2.3 Opslag van revisies 35
9.2.4 Implementatie van Envers 35
9.2.5 Ophalen en bepalen van revisies 36
9.3 Beheren van validatie en afhankelijkheden in het front-end 37
9.4 Afhandelen van ingevulde formulieren 39
9.4.1 Binden van data 39
9.4.2 Valideren formulier 39
9.4.3 Uitvoeren afhandelingsacties 40
9.4.4 Bedanktekst / Bedankpagina 40
9.5 Hallway testing 41
9.6 Wat ging er goed 43
9.7 Wat ging er niet goed 43
10. Sprint 3 44
10.1 Sprint planning meeting 44
10.2 Refactoring; leren van de testresultaten 44
10.3 Evolutie van de user interface 46
10.4 Koppeling met de CRM module 49
10.5 Advies Scrum bij Eleven 51
10.6 Wat ging er goed 52
10.7 Wat ging er niet goed 52
11. Evaluatie 53
11.1 Productevaluatie 53
11.2 Procesevaluatie 53

11.2.1 Wat ging er goed 53
11.2.2 Wat ging er niet goed 54
12. Verantwoording competenties 55
12.1 Opstellen gegevensmodel voor database [niveau 4] 55
12.2 Ontwerpen systeemdeel [niveau 3] 55
12.3 Bouwen applicatie [niveau 4] 55
12.4 Initiëren en plannen van het testproces [niveau 3] 55
12.5 Uitvoeren van en rapporteren over het testproces [niveau 3] 55
13. Referenties 56
13.1 Woordenlijst 56
13.2 Bibliografie 56
13.3 Bijlagen 57

2. Inleiding

Afsluitend onderdeel van iedere HBO opleiding is het afstuderen. Voor mij hield dat in dat ik in april 2011 op zoek mocht naar een afstudeeropdracht. Gelukkig heb ik bij mijn werkgever Eleven een leuke, uitdagende opdracht kunnen vinden voor mijn afstuderen.

In het verslag wat voor u ligt beschrijf ik de achtergronden van de opdracht en het bedrijf waar ik mijn opdracht heb uitgevoerd. Na een beschrijving van de software ontwikkelmethode Scrum in hoofdstuk 6 worden in de navolgende hoofdstukken iedere sprint nader toegelicht. Tot slot evalueer ik zowel het product als het proces. Het verslag sluit af met referenties naar gebruikte literatuur en definities van woorden.

De opbouw van dit verslag is als een procesverslag, het beschrijft het proces van de totstandkoming van de eindproducten. Opgeleverde producten en documenten zijn opgenomen als bijlage en zijn beschreven in hoofdstuk 13.

3. De afstudeeropdracht

3.1 Probleemstelling

Veel van onze klanten maken gebruik van ons eigen ontwikkeld Toolbox Web Management Systeem. Met behulp van Toolbox kunnen beheerders grote delen van hun website beheren. Dit kan variëren van een tekst pagina, een product in een catalogus of een vraag van een online enquête.

Als een beheerder een formulier wilt toevoegen aan de website dan biedt Toolbox daar nu geen mogelijkheid toe. De beheerder moet dan contact opnemen met Eleven zodat wij een formulier kunnen maken en implementeren.

Dit proces kost tijd en geld voor de klant en voor iedere wijziging is hij afhankelijk van Eleven. Daarnaast zal het altijd enige tijd duren voordat het formulier toegepast kan worden op de site.

3.2 Doelstelling

Om dit probleem te verhelpen is het doel van deze afstudeeropdracht om een nieuwe module voor Toolbox WMS te maken waarin website beheerders zelf formulieren kunnen aanmaken en deze kunnen gebruiken in hun website.

Deze nieuwe formulierenmodule zorgt ervoor dat een beheerder een formulier kan toevoegen, bewerken of verwijderen. Een formulier bestaat uit verschillende elementen. Dat kan een van de volgende elementen zijn:

- Tekstveld
- Uitgebreid tekstveld
- Enkelvoudige selectie
- Meervoudige selectie
- Keuzelijst
- Bestand (foto's, documenten, pdf's)
- Captcha (controlemethode om menselijke invoer te testen)
- Verzend knop

Ieder element heeft eigenschappen die specifiek zijn voor dat element. Zo kan men aan een tekstveld een veld lengte meegeven. Aan de selectie elementen zal men een aantal waarden moeten opgeven waaruit gekozen kan worden. Bijvoorbeeld de waarden man en vrouw voor het veld geslacht.

Een beheerder kan op ieder element validatieregels instellen. Zo kan er bepaald worden dat het aan een aantal minimaal of maximaal aantal karakters moet voldoen of aan een bepaald patroon. Patronen kunnen vooraf door Eleven gedefinieerd zijn bijvoorbeeld om een geldig e-mailadres of URL te valideren. Er kunnen ook eigen patronen opgegeven worden. Bijvoorbeeld voor een personeelsnummer die een uniek patroon bevat binnen een bedrijf.

Niet alleen de validatie kan bepaald worden, maar ook de afhankelijkheden tussen de diverse elementen. Zo kan de beheerder besluiten dat element C pas getoond mag worden als element A de waarde 'ja' heeft. Een praktisch voorbeeld is het veld 'Aanmelden voor nieuwsbrief' pas te tonen wanneer er ook een e-mailadres is ingevuld. Op deze manier hoeft een bezoeker van de site alleen maar relevante informatie op te geven en zal hij niet gestoord worden met onnodige velden en vragen.

Met een ingevuld formulier moet ook wat gebeuren. Een beheerder kan daarvoor diverse afhandeling acties opgeven. De mogelijke afhandeling acties zullen zijn:

- Opslaan in database (standaard geactiveerd)
- Mailen naar opgegeven e-mail adres
- Printen naar PDF

Een beheerder kan kiezen voor meerdere afhandeling acties per formulier en zo met de informatie in meerdere doelen voorzien.

De achterliggende gedachte van de module is dat het voor de beheerder gebruiksvriendelijk in elkaar moet steken en dat hij of zij zonder een uitgebreide training met de module aan de slag moet kunnen.

3.3 De methode Scrum

Een extra onderdeel in deze opdracht is een advies uitbrengen over de software ontwikkelmethode Scrum. De methode Scrum is een populaire methode om software te ontwikkelen daar zij zich eenvoudig kan aanpassen aan wisselende omstandigheden.

Eleven maakt al gebruik van deze methode in haar projecten, zij het op een heel simpel niveau en slechts enkele onderdelen. De opdrachtgever wil graag weten hoe ze deze methode beter kunnen inzetten en hoe ze dit kunnen implementeren in hun werkwijze.

Het doel is om een rapport op te leveren waarin de methode Scrum beschreven wordt. Het rapport zal laten zien welke onderdelen van Scrum ingezet zouden kunnen worden bij Eleven en op welke wijze ze dit kunnen toepassen. Hoe ze deze onderdelen het beste kunnen implementeren wordt in het rapport door middel van een advies beschreven.

4. Eleven

4.1 Het bedrijf

Eleven is een relatief jong bedrijf wat is ontstaan uit een samenwerking tussen de afdeling Webdevelopment van kabelmaatschappij CAIW Diensten en Studio11 internet solutions. Deze samenvoeging heeft plaatsgevonden in november 2009 (Eleven).

Eleven is werkzaam voor het midden- en kleinbedrijf en biedt een variëteit aan diensten voor haar klanten, zoals:

- Het bedenken en bouwen van (slimme) webapplicaties
- Het vindbaar en bruikbaar maken van die slimme applicaties
- IT-consultancy en projectmanagement

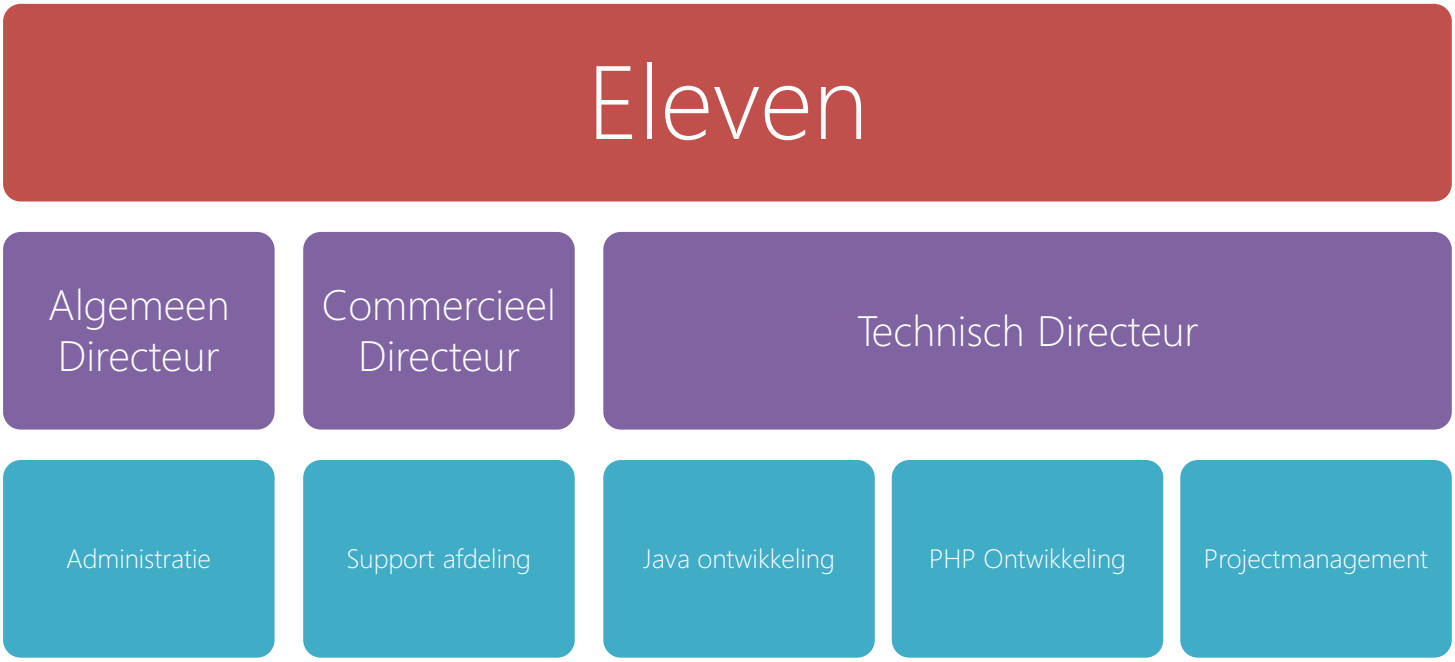
Dat doen zij voor een keur aan bedrijven, stichtingen en andere organisaties. Een greep uit onze klanten:

- 5xbeter
    - Een samenwerkingsverband van Koninklijke metaalunie, FME/CWM, FNV Bondgenoten, CNV Vakmensen en De Unie
- Kabelmaatschappij CAIW
- Flora Holland
- Van der Windt
- Aluminiumopmaat.nl
- Waterpop
- Het Hele Westland
- Fides

Binnen Eleven wordt er gewerkt met zowel de programmeertaal PHP als Java. Het eigen ontwikkelde Toolbox is gemaakt in Java en komt van oorsprong vanuit CAIW. De taal PHP wordt voornamelijk gebruikt voor de websites die gebruik maken van de open source pakketten Joomla en Magento.

4.2 Bedrijfsstructuur Eleven

Zoals gezegd is Eleven een jong bedrijf wat ontstaan is uit een samenvoeging van twee bedrijven. Er zijn op het moment van schrijven 14 medewerkers actief, waarvan het grootste deel ontwikkelaars zijn. Het bedrijf word geleid door drie directeuren die ieder verantwoordelijk zijn voor een deel van de organisatie. In een organogram ziet dat er als volgt uit.



4.3 Mijn plek binnen Eleven

Begin 2010 besloot ik na vijf jaar te stoppen met mijn eigen onderneming. Met mijn onderneming maakte ik websites en web applicaties voor kleine ondernemingen. Met veel plezier heb ik altijd dit bedrijf geleid, maar naar mate de tijd vorderde kwam ik er steeds meer achter dat ik meer een softwareontwikkelaar en softwarearchitect ben dan een ondernemer. Het contact met klanten en het maken van de applicaties was geweldig, maar het binnenhalen van nieuwe klanten en orders en de administratieve aspecten gingen mij steeds meer tegenstaan.

In mijn zoektocht naar een goed bedrijf voor mijn klanten kwam ik uiteindelijk terecht bij Eleven. Zij sloten perfect aan bij de waarden en gedachten die ik altijd voor mijn klanten heb gehanteerd en nog steeds heb. De overname kreeg zijn beslag in mei 2010 waarna ik van ondernemer naar medewerker van Eleven ging. Mijn functie was toen en op het moment van schrijven webdeveloper PHP/Java.

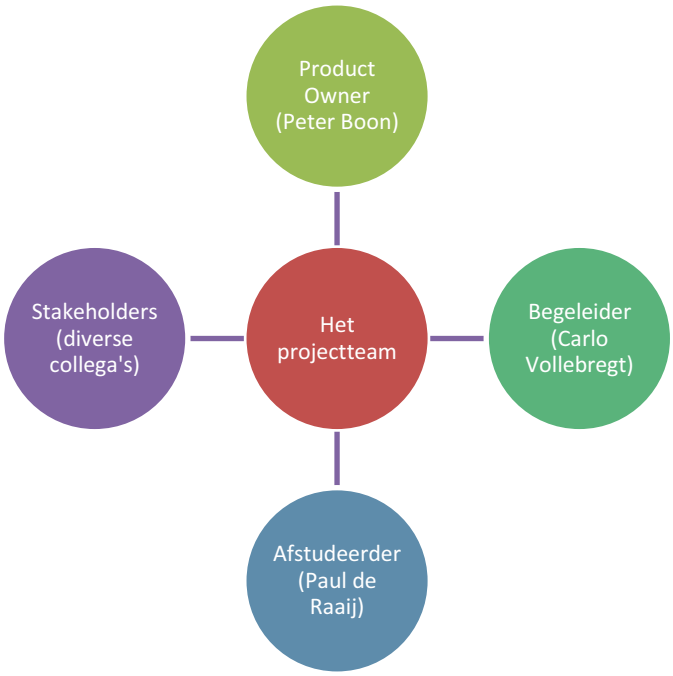
In die hoedanigheid ben ik voornamelijk verantwoordelijk voor de ontwikkeling van websites en web applicaties in zowel PHP als Java. De laatste maanden ben ik ook steeds vaker verantwoordelijk voor het maken van het technisch ontwerp en het voeren van het technisch management over projecten.

4.4 De projectorganisatie

Het afstudeerproject dat ik uitvoer is een intern project en heeft daarmee ook een projectorganisatie die uit de eigen gelederen komt. De opdrachtgever, en product owner, is de algemeen directeur van Eleven, Peter Boon. De technisch directeur Carlo Vollebregt zal mij tijdens mijn afstuderen begeleiden in het traject.

Tijdens het afstudeerproject zullen collega's mij ondersteunen in de technische werking van Toolbox of door een kleine test uit te voeren in het kader van de Hallway tests.

Het project team ziet er dan grafisch gezien als volgt uit:



5. Toolbox omgeving

Binnen Eleven werken wij met een zelf ontwikkeld Web Management Systeem getiteld Toolbox. Toolbox is ontwikkeld binnen de webdevelopment afdeling van CAIW en word door Eleven verder ontwikkeld en ingezet bij projecten.

De eerste ontwikkeling van Toolbox is gestart in 2007 en is een closed-source product. Begonnen als een nieuw content management systeem is Toolbox in de loop der jaren uitgegroeid tot een framework vol met basis functionaliteiten voor de ontwikkelaar.

Tevens is Toolbox gedurende de ontwikkeling voorzien van diverse modules die ingezet kunnen worden voor een website of een web applicatie. Een greep uit deze modules:

- Catalogus
- Nieuwsbrief
- Nieuws
- CRM
- Media bibliotheek
- Enquête

Toolbox is ontwikkeld in de objectgeoriënteerde programmeertaal Java. Naast dat het een Web Management Systeem voor klanten is, acteert het onder andere ook als een framework voor ontwikkelaars.

Klant specifieke wijzingen of functionaliteiten aan een website of applicatie worden door de ontwikkelaar op basis van Toolbox gemaakt. Daarvoor beschikt het over twee basis libraries getiteld 'Cool' en ' Coolkernel'.

Beide libraries zijn ontwikkeld door de webdevelopment afdeling van Caiw en vormen de basis van ieder project dat in de afgelopen jaren bij CAIW en Eleven zijn ontwikkeld. Deze libraries bevatten een flink aantal externe libraries en eigen code die een set van generieke functionaliteit geeft aan de ontwikkelaar.

Een functionaliteit van Cool is het bieden van een tussenlaag tussen Hibernate en de applicatie specifieke code voor acties omtrent de database. Het biedt functionaliteiten om geagendeerde taken uit te voeren, dit gebruiken we bijvoorbeeld om op een vast tijdstip op een dag een rapportage te genereren.

De libraries Cool en Coolkernel zijn de bron van Toolbox en vormen de tussenlaag tussen een groot aantal third party libraries die we binnen Toolbox gebruiken. Belangrijke onderdelen die Cool en Coolkernel ons bieden zijn Spring Framework, Hibernate ORM en EhCache.

5.1 Spring Framework

Spring Framework bestaat uit een groot aantal API's en ideeën die een uitbreiding zijn op de aanwezige functionaliteiten van Java. Het forceert de ontwikkelaar om zijn applicatie functioneel gezien op te delen in een aantal tiers (lagen). Door die opdeling krijgt iedere laag zijn eigen verantwoordelijkheid en lopen deze niet door elkaar heen waardoor de applicatie lastig en duur te onderhouden is.

Twee belangrijke zaken die het Spring Framework onze Cool library, en daarmee ook Toolbox, biedt, is het beschikbaar stellen van een Model-View-Controller architectuur.

Hierdoor is het eenvoudig om de business logica te scheiden van de presentatie. De applicatie is hierdoor eenvoudiger in het onderhoud en zal qua code eenvoudiger voor de ontwikkelaar te begrijpen zijn.
Een ander belangrijk onderdeel hierin is het principe van Inversion of Control, oftewel Dependency Injection. Het idee van dit principe is dat een container geen eisen mag stellen aan de implementatie van de business objecten.

De container is een object die de technische infrastructuur biedt aan een business object. Het is zijn verantwoordelijkheid dat alles wat een business object nodig heeft ook bij dat object aanwezig komt.
Een business object is een entiteit die een onderdeel van de business case representeert. Dit kan een klant zijn in een winkel, een order in een bestelbedrijf of een caravan in een caravanstalling.

Business objecten beschikken over attributen die een bepaalde eigenschap van het object voorstellen. Zo zal een klant over het attribuut "naam" beschikken en zal een caravan een attribuut "type" bevatten.
Een eigenschap van een business object is dat zij niet binnen 1 laag van de applicatie blijven, maar door de verschillende lagen heen zwerven. Hier komt dan ook het voordeel van Inversion of Control (IoC) kijken.
Wat IoC als voordeel biedt is dat het ieder business object als een generieke simpele klasse ziet en hanteert, namelijk een JavaBean. Eigenschap van een simpele JavaBean is dat het niets van de rest van het systeem weet. Hierdoor is een business object, oftewel JavaBean, op ieder willekeurige plek te gebruiken, zolang de container maar aanspreekbaar is.

Het is de verantwoordelijkheid van de container om te voldoen aan de wensen van een business objecten.

5.2 Hibernate ORM

Hibernate is een Object Relational Mapping library voor Java en inmiddels ook .NET. Het biedt een framework om een objectgeoriënteerd domein model in kaart te brengen in een relationele database.

Door de entiteiten te configureren door middel van annotaties kan Hibernate automatisch de database tabellen opbouwen. Daarnaast biedt Hibernate functionaliteiten om relaties vast te leggen en kan het omgaan met eigen gedefinieerde typen.

Binnen Toolbox maken we hevig gebruik van Hibernate. Het is onze tussenlaag tussen onze code en de database. Binnen Cool zit een extra tussenlaag die een groot aantal basisfunctionaliteiten beschrijft waar wij ons binnen Toolbox weer gebruik van kunnen maken.

5.3 Ehcache

Een ander belangrijke third party library binnen onze libraries is Ehcache. Ehcache is een generieke oplossing om caching toe te passen. In 2003 is Greg Luck gestart met de ontwikkeling van Ehcache.

Door gebruik te maken van deze library kunnen we de performance van Toolbox verbeteren. Door entiteiten die veel gebruikt worden in het geheugen te cachen halen we significante verbeteringen. Niet alleen entiteiten maar ook queries die op de database worden uitgevoerd worden door Ehache in de cache opgeslagen indien gewenst.

6. Projectmethode: Scrum

6.1 Agile

Voor dit project heb ik in samenspraak met Eleven besloten om gebruik te maken van een agile software ontwikkelingsmethode. Het woord agile betekent letterlijk vertaald behendig en lenig. Precies de omschrijvingen die ik zoek bij de wijze waarop ik dit project wilde uitvoeren. Daarnaast is dat ook de gedachtegang die Eleven hanteert en toepast bij het uitvoeren van projecten.

De agile methode is een conceptueel raamwerk wat ontstaan is als tegenhanger voor de traditionele projectmethoden. Agile is ontstaan in februari 2001. Dit concept is ontstaan tijdens een informele bijeenkomst van ontwikkelaars waarin het agile manifesto is opgesteld. Dit manifesto beschrijft twaalf principes die stellen hoe goede software gemaakt dient te worden. (manifesto)

Een agile methode bestaat uit diverse kenmerken.

6.1.1 Iteraties

Vrijwel alle agile methoden proberen de risico's te verminderen door te werken in korte tijdsperiode, genoemd timeboxes. Iedere iteratie is een eigen miniproject en zal altijd een werkende applicatie opleveren. Er zal aan het eind van iedere iteratie bepaald worden of het ook daadwerkelijk in productie wordt genomen of niet. Na iedere iteratie zal het project team de project prioriteiten heroverwegen.

6.1.2 Communicatie

In agile methoden is er veel aandacht voor communicatie. In plaats van verslaglegging ligt de nadruk op direct en veel contact. Bij voorkeur is ook de klant nauw betrokken bij het project door een product manager op de locatie van het projectteam te huisvesten en bij de bouw te betrekken.

6.1.3 Voortgang

In tegenstelling tot de traditionele methode wordt de voortgang van een project niet gemeten in rapportage documenten, maar door de status van het product of prototypes te bepalen.

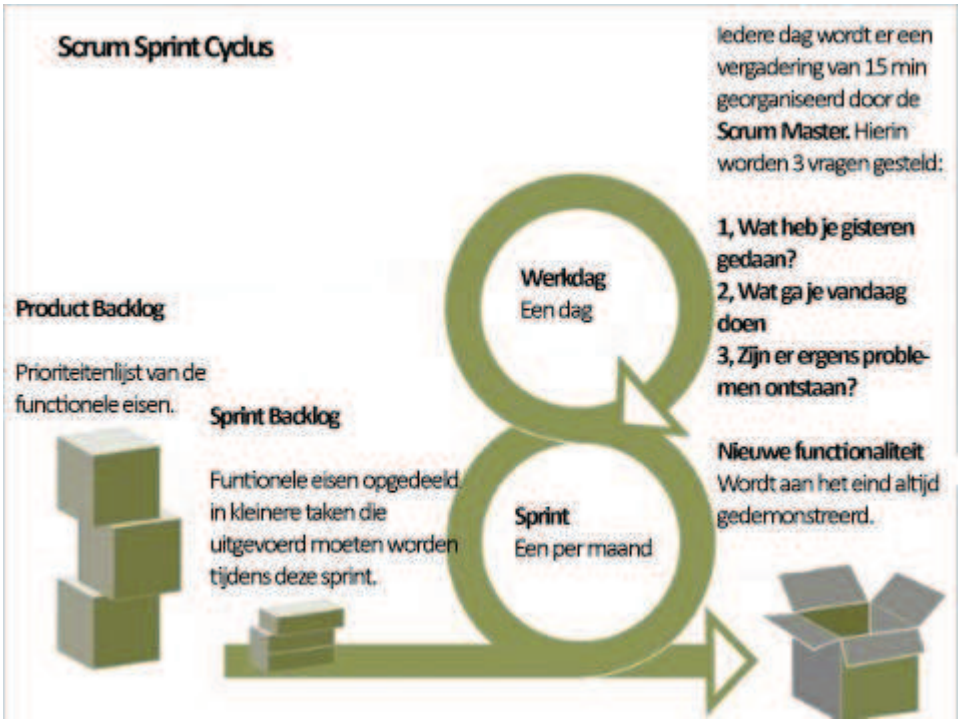
6.2 De methode scrum

De agile methode die ik voor dit project gebruik is de methode Scrum. De methode Scrum voldoet aan de gestelde kenmerken zoals eerder beschreven. Daarnaast is vanuit Eleven het verzoek geweest om deze methode te gebruiken zodat zij kunnen zien hoe ze deze methode kunnen inzetten in hun eigen projecten.

De term Scrum komt uit de rugbysport alwaar een grote groep spelers schouder aan schouder proberen de bal aan de overkant te brengen. De Scrum methode is in 1995 geformaliseerd als ontwikkelmethode nadat het in 1986 al is gepubliceerd in de Harvard Business Review. (Wikipedia-Scrum)

Doelstellingen van de Scrum methode zijn onder andere het verhogen van de effectiviteit van het team en het bewaken van de voortgang.

Doordat Scrum zich focust op veel communicatie en contact zorgt het voor een verhoogde teamspirit binnen het projectteam. Problemen zijn nu bij iedereen binnen het team bekend en zo kan het team sneller tot oplossingen komen.



Scrum kent een cyclus die zich tot het oneindige kan herhalen. Belangrijk element in de Scrum methode is de sprint. De sprint is de periode waarin het werk daadwerkelijk word uitgevoerd. Een sprint is altijd getimeboxed en kan variëren van een week tot zes weken. De keuze van de duur ligt in handen van het ontwikkelteam in overleg met de product owner.

In een sprint wordt door het team gewerkt aan de applicatie. Vereiste is dat het team aan het einde van de sprint altijd een werkende applicatie oplevert. Of deze versie ook daadwerkelijk in productie wordt genomen is een keuze van team en de product owner. Hoe dan ook mag er aan het eind van de sprint geen halve applicatie opgeleverd worden. User stories die nog niet afgerond zijn worden niet opgeleverd aan het eind van de sprint en zullen in de volgende sprint afgerond worden.

6.3 Technieken en hulpmiddelen

De Scrum methode kent een aantal technieken en hulpmiddelen om tot een gedegen projectverloop te komen.

6.3.1 Daily Stand Up

Een van de technieken die Scrum aandraagt is de Daily Stand Up. Dit is een dagelijkse vergadering die bij voorkeur in de ochtend gehouden wordt. Deze vergadering heeft als doel om het team op de hoogte te houden van elkaars voortgang. Daartoe zijn er drie vragen die ieder teamlid moet beantwoorden tijdens de Daily Stand Up:

1. Wat heb je gister gedaan
2. Wat ga je vandaag doen
3. Voorzie je problemen in het project of je werkzaamheden

De vergadering is iedere dag op hetzelfde tijdstip en bij voorkeur ook locatie. De projectleden worden aangemoedigd om de vergadering bij te wonen, maar het tijdstip zal niet verschuiven vanwege de afwezigheid van een van de medewerkers.

Om alle deelnemers er aan te herinneren dat de vergadering kort en tot de noodzaak te houden dienen alle deelnemers te staan tijdens de vergadering, vandaar ook de naam: The Daily Standup.

6.3.2 Sprint Planning meeting

Voor de start van een sprint houdt het project team in samenspraak met de Product Owner een sprint planning meeting. In deze vergadering bepaalt het team samen met de Product Owner welke user stories uitgevoerd gaan worden in de komende sprint.

Vaak worden deze vergaderingen in twee delen opgesplitst. Het eerste deel wordt gebruikt om de user stories te identificeren die opgepakt moeten worden. In het tweede deel is de Product Owner niet aanwezig en stippelt het team het juiste plan uit voor de komende sprint.

6.3.3 Sprint Retrospective meeting

De Sprint Retrospective Meeting is de vergadering waarin alle teamleden reflecteren op de afgelopen sprint. Twee vragen staan hierin centraal: (Agile-Development-Site)

1. Wat ging er goed in deze sprint?
2. Wat kan er beter in de volgende sprint?

Het voornaamste doel van deze sprint is om te leren hoe komende sprints beter kunnen verlopen. Deze meeting is dan in essentie ook alleen voor het team, maar optioneel kan de Product Owner ook aanwezig zijn bij deze bijeenkomsten.

Voor iedere deelnemer aan de bijeenkomst is het noodzakelijk dat ze begrijpen dat het gaat om het proces en de productiviteit te verbeteren. Het is niet de bedoeling om elkaar onderuit te halen.

Wat er ook uit de Sprint Retrospective Meeting te voorschijn komt, het is de taak van de Scrum Master om deze punten beet te pakken en zorgen voor eventuele wijzigingen.

6.3.4 Story points

Een Story point is een relatieve en abstracte eenheid van complexiteit van een user story. De relativiteit komt voort uit het feit dat een user story van 2, tweemaal zo complex is als een user story van 1.

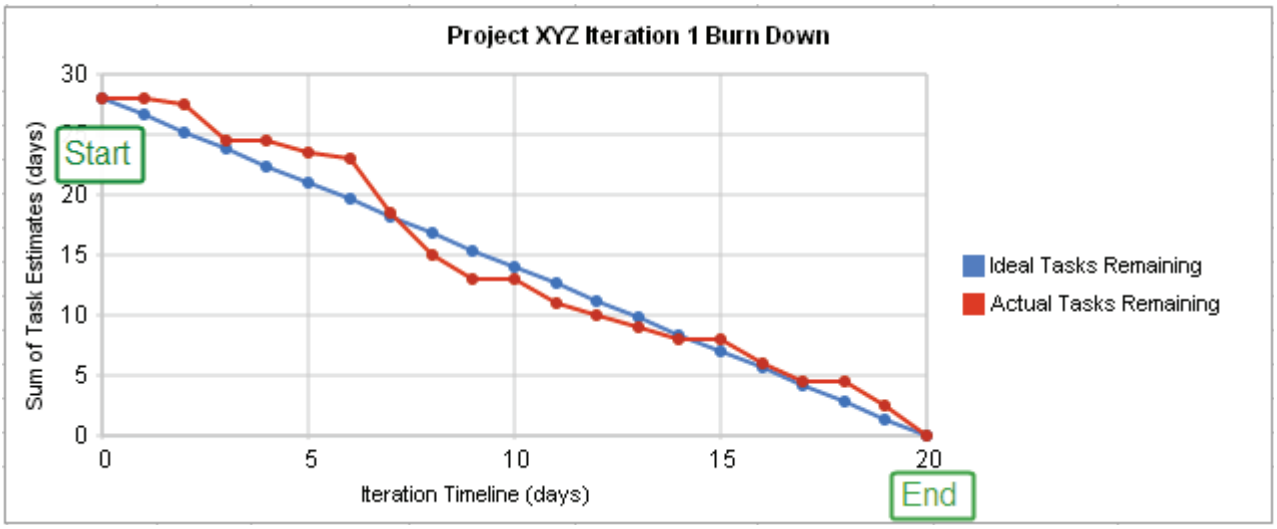
Het voordeel met het werken van een relatieve puntenschaal in tegenstelling tot een ureninschatting is dat het inschatten van uren zich richt op het beantwoorden van de vraag: “Hoe lang gaat het duren?”. Over het algemeen een wilde gok op basis van eerdere ervaringen.

Het inschatten van punten richt zich meer op de grootte en complexiteit van een user story. Hierdoor kunnen user stories op gelijkwaardige basis met elkaar vergeleken worden. Een goede schaal om te gebruiken is de Fibonacci reeks, 1-2-3-5-8-13-21-34.

Het is een arbitraire eenheid aangezien er geen vaste afspraken zijn wat een bepaalde waarde aan complexiteit representeert. Deze afspraken worden binnen een team gemaakt en kunnen verschillen per project.

6.3.5 Burndown chart

De burndown chart is een grafische weergave van de voortgang van de sprint. Het laat zien hoeveel werk er nog uitgevoerd moet worden en hoeveel werk er al verzet is.



De grafiek bestaat doorgaans uit twee assen en twee lijnen waarvan er één linear is. De x-as is de tijdlijn van het project. Aangezien de grafiek dagelijks wordt bijgewerkt is het verstandig om deze x-as ook in de eenheid van dag op te zetten.

De y-as toont het werk wat gedaan moet worden. Hier kan het team kiezen voor een eenheid. Dit kan het werk in dagen, functiepunten of uren zijn. De methode laat het project team vrij in deze keuze.

Lijn 1, getiteld ideal tasks remaining, is lineair en laat zien wat de ideale burndown van het werk is. De andere lijn, getiteld: actual tasks remaining, is de representatie van het werkelijk uitgevoerde werk. Zodra een user story is afgerond neemt het werk te doen af en zal de lijn dichter bij de lijn met het ideale taak verloop komen.

Als de actuele lijn zich boven de ideale lijn bevindt zal er geanalyseerd moeten worden waarom er minder werk is uitgevoerd dan er idealiter gedaan had moeten zijn. Het kan zijn omdat user stories zich in de afrondende fase zitten en op korte termijn afgerond worden. Het kan zijn dat er problemen zijn die opgelost moet worden.

In ieder geval laat de burndown chart zien wat de status van het project is en of er ingegrepen moet worden. Dit kan door het user stories door te schuiven naar de volgende sprint of door meer resources beschikbaar te maken.

6.4 Waarom dan Scrum?

De meest belangrijke reden is omdat dit een specifieke eis van Eleven is voor deze afstudeeropdracht omdat zij willen leren wat deze methode voor hun kan betekenen. Waarom kiezen ze dan voor Scrum?

Doordat de Scrum methode zich sterk richt op communicatie en het monitoren van de voortgang van het project zal het eindresultaat sneller en met hogere kwaliteit opgeleverd worden. Problemen en onduidelijkheden worden sneller waargenomen en verholpen door de diverse overlegmomenten.

Voor de projectleider, accountmanager en directie geeft de methode een beter inzicht in de projectvoortgang. De burndown chart geeft in één oogopslag weer wat de hoeveelheid werk is die nog gedaan moet worden en of het gestelde einddoel wel realistisch is.

Met behulp van deze methode zal het team meer betrokken zijn bij het project. Dat is het gevolg van het feit ze meer betrokken zijn bij het maken van beslissingen, keuzes in technische vraagstukken en door hun eigen inschattingen over de benodigde tijd van een user story. Hetzelfde geldt voor stakeholders die vertegenwoordigd worden door een Product Owner. Ze zijn meer betrokken bij de keuzes van het werk wat uitgevoerd gaat worden in een sprint. Daarnaast hebben ze ook meer invloed op het werk en de uiteindelijke applicatie.

Door gebruik te maken van de methode Scrum hoopt Eleven de projecten soepeler laten verlopen. Dit moet ervoor zorgen dat de producten met een hogere kwaliteit opgeleverd worden en nog beter voldoen aan de verwachtingen van de klant.

Dit project was een goede pilot voor Eleven om deze vorm van Scrum te proberen omdat in dit geval alle rollen intern vertegenwoordigd worden en daardoor goed kennis gemaakt wordt van de inhoud en verantwoordelijkheden van alle rollen. Andere reden is het feit dat dit project de ruimte biedt om de methode dieper te onderzoeken zonder druk van een klant om een project af te ronden tegen een bepaald budget.

Wel moet gerealiseerd worden dat de uitvoer van het project een eenmansproject is. Ondanks de aanwezigheid van een Product Owner is er relatief weinig communicatie omdat er maar één persoon, weliswaar met steun van een begeleider, het project ontwikkeld.

7. Sprint 0

Het project is gestart met een korte eerste sprint waarin de juiste vereisten verzameld en opgezet zijn. Belangrijkste taken in deze sprint waren het maken van het plan van aanpak, het verzamelen van de requirements en het opstellen van het test plan. Deze sprint 0, waarin nog niet aan het uiteindelijke product werd gewerkt, was ingepland voor een duur van twee weken. De start van deze sprint was 1 augustus 2011 en liep top 13 augustus 2011.

7.1 Plan van aanpak

Als eerste stap in deze sprint heb ik een plan van aanpak opgesteld om voor mijzelf duidelijk te krijgen hoe ik dit afstudeerproject ging aanvliegen. Belangrijke facetten in het plan van aanpak was de globale planning en de opbouw van de projectorganisatie.

7.1.1 Globale planning

Belangrijk onderdeel in het plan van aanpak is de globale planning. Deze planning geeft aan hoe de sprints verdeeld zouden worden en hoeveel werk er ook daadwerkelijk verzet kan gaan worden in het afstudeertraject.

Er is besloten om voor dit afstudeerproject gebruik te maken van de software ontwikkelmethode Scrum omdat Eleven graag wilt weten wat deze methode voor hun kan betekenen. Dat heeft als gevolg dat het project in iteraties word uitgevoerd en er bij de start van iedere iteratie wordt pas bepaald wat er wordt uitgevoerd.

Om die reden staat er in het plan van aanpak alleen een globale planning waarin de timeboxes van de sprints bepaald is. De globale planning is als volgt.

Start datum	Eind datum	Activiteit
1 augustus 2011	13 augustus 2011	Sprint 0
15 augustus 2011	30 september 2011	Sprint 1
3 oktober 2011	11 november 2011	Sprint 2
14 november 2011	23 december 2011	Sprint 3

7.1.2 Definition of Done

Een ander document wat ik in deze sprint heb opgesteld is de Definition of Done. De Definition of Done is in feite een kleine checklist die de ontwikkelaar kan gebruiken om te controleren of de user story voldoet aan de eisen die het projectteam stelt. De activiteiten op de Definition of Done zijn waardevolle activiteiten om de kwaliteit van de software te controleren of te verhogen. Activiteiten kunnen bijvoorbeeld zijn:

- Schrijven van functionele tests
- De code is gedocumenteerd
- De code is beoordeeld door een collega

De Definition of Done zorgt ervoor dat er een eenduidige afspraak is aan welke eisen een user story moet voldoen om daadwerkelijk als af beschouwd te worden. De Definition of Done heb ik in samenspraak met mijn begeleider opgesteld en ziet er als volgt uit

- ☐ Alle unit tests hebben een coverage van minimaal 60%
- ☐ Frontend functionaliteit heeft minimaal 1 selenium test
- ☐ Alle unit en functional tests slagen
- ☐ Alle java code beschikt over Javadoc documentatie
- ☐ De HTML is W3C compliant
- ☐ De complete user story is succesvol gebouwd door de CI server

7.2 Requirements verzamelen

In de voorbereiding op dit afstudeerproject heb ik enkele malen met mijn begeleider en de product owner gesproken. In deze gesprekken hebben we het globaal over het gewenste eindproduct gehad. Deze gesprekken hebben bij lange na niet genoeg informatie gegeven om een goed beeld te hebben van het eindproduct.

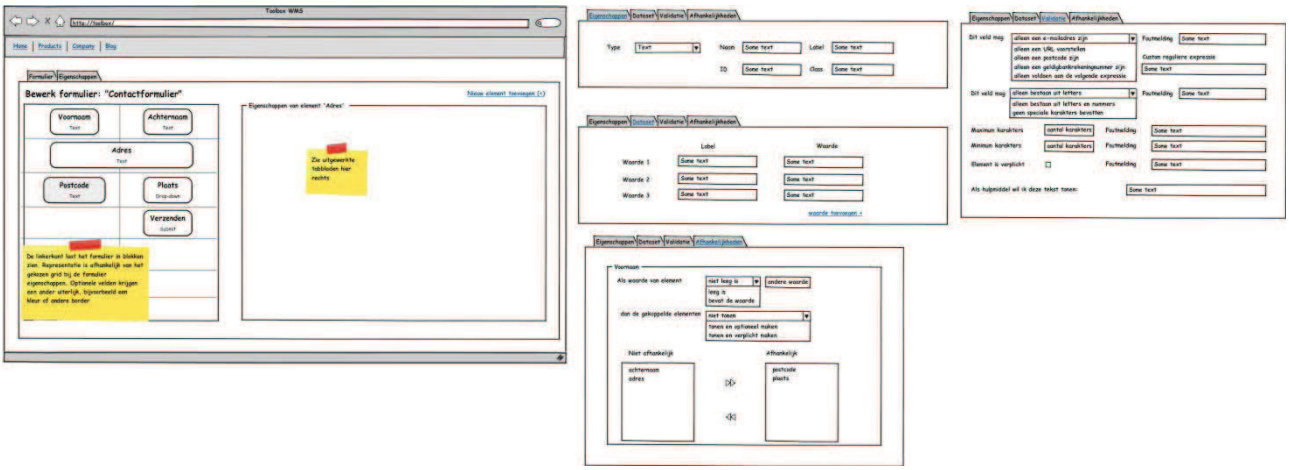
In deze sprint hebben we dan ook eerst een sessie gehouden waarin we uitgebreid over de nieuwe formulierenmodule hebben gesproken. Mijn begeleider en product owner hebben duidelijk aangegeven wat ze van de formulierenmodule verwachten en hoe zij het eindproduct voor zich zien.

Mijn taak in deze vergadering was voornamelijk het notuleren en beschrijven van de vereisten en opmerkingen. Eveneens heb ik veel mee gedacht over het eindproduct en voornamelijk vanuit gebruikersperspectief bekeken hoe zij eenvoudig met deze module te werk zouden kunnen gaan.

Deze notulen heb ik daarna verwerkt in het prduct backlog. In het product backlog staan alle user stories die bedacht zijn. Het hoeft niet per se zo te zijn dat alle user stories ontwikkeld worden. Het backlog kan ook hersenspinsels van de Product Owner bevatten die ter registratie zijn genoteerd.

Van alle ideeën, wensen en functionaliteiten die zijn geopperd heb ik alles verwerkt tot user stories en toegevoegd aan het product backlog.

Ter validatie heb ik dit product backlog daarna nogmaals besproken met mijn begeleider en product owner om er zeker van te zijn dat we over hetzelfde spreken en hetzelfde beeld van het eindproduct hebben. Voor extra validatie heb ik ook een mockup gemaakt die een grafische weergave van de mogelijkheden van de toekomstige formulierenmodule liet zien.



Deze mockup is ook bijgesloten als bijlage 6.

Het voordeel van het werken met een mockup is dat het gewenste eindresultaat meer begint te spreken en dat het eenvoudiger word om over de functionaliteit te discussiëren. Prettige bijkomstigheid is ook dat het mogelijk word om over het gedrag van de interface te brainstormen. Zo kwamen wij er in onze sessie goed achter wat door middel van drag and drop zou kunnen werken. Ook konden we bespreken hoe bepaalde schermen geopend moesten worden.

Voor alle Scrum activiteiten heb ik gebruik gemaakt van een online web applicatie getiteld Pango Scrum. Pango Scrum is een open source web applicatie waarin je producten en sprints kun definiëren. Van iedere sprint kun je vervolgens bepalen welke items uit het product backlog je gaat oppakken. Daarnaast genereert het ook een burndown chart en laat het op eenvoudige wijze zien wat de status van het project is.

7.3 Master Test Plan

Tijdens het project zal ik de principes van Agile Testing volgen. De ideologie achter deze methode van testen is om nieuw ontwikkelde code zo snel mogelijk te testen.

Door zo snel mogelijk de applicatie te testen blijf je dicht bij een mogelijke fout en zijn de kosten en de complexiteit om de fout op te lossen zo laag mogelijk.

Achter deze filosofie kan ik me volledig scharen. In deze voorbereidingsprint heb ik een Master Test Plan opgesteld om te identificeren welke verschillende test methode ik ga gebruiken tijdens de ontwikkeling van de nieuwe formulierenmodule.

Technieken waarvoor ik heb gekozen om te gebruiken voor het testen van de applicatie zijn:

- Mockup's
- Hallway testing
- Exploratory testing
- Unit testing
- GUI testing
- User acceptance test

De verantwoording en beschrijving van deze testen worden in het bijgesloten test plan nader omschreven.

De gekozen testsoorten zijn niet willekeurig gekozen, maar zijn gekozen omdat zij van toepassing zijn op een aantal kwaliteitsattributen uit de ISO 9126 standaard. Ze sluiten aan bij het gedachtegoed wat Eleven heeft met Toolbox en de nieuw te ontwikkelen formulierenmodule.

De ISO 9126 standaard biedt een aantal kwaliteitskenmerken waaraan software getoetst kan worden. Het kadert begrippen af zodat het eenvoudiger is om overeenstemming te bereiken over de kaders van een kwaliteitskenmerk. (Wiki-ISO_9126)

In samenspraak met de Product Owner en mijn begeleider hebben wij besloten om ons tijdens de ontwikkeling en testen van de nieuwe module te richten op de volgende kwaliteitsattributen:

- Understandability
- Learnability
- Maturity

Wij hebben voor deze kwaliteitsattributen gekozen omdat ze aansluiten bij de eisen die Eleven stelt aan de nieuwe module. De focus moet liggen op gebruiksvriendelijkheid van de module zodat zij eenvoudig door de beheerders gebruikt kunnen worden zonder een uitvoerige training en ondersteuning. Voornamelijk de attributen understandability en learnability hebben hier betrekking op.

Het attribuut maturity is gekozen omdat Eleven van de applicatie verwacht dat als er iets misgaat niet gelijk de hel losbarst. Fouten kunnen natuurlijk altijd voorkomen, maar dat mag niet de applicatie lam leggen. Een eventuele foutmelding moet netjes afgevangen worden en indien nodig een melding geven aan de gebruiker.

8. Sprint 1

Na de afwikkeling van sprint 0 ben ik verder gegaan met sprint 1. Deze sprint liep van 15 augustus 2011 tot 30 september 2011. Deze sprint was de eerste sprint waar daadwerkelijk ontwikkeling van het systeem is gestart.

8.1 Sprint planning meeting

De sprint begon met het houden van een sprint planning meeting. Deze meeting is een belangrijk onderdeel van de Agile methode Scrum.

In de Sprint planning meeting bepaalt het team in samenspraak met de product owner welke user stories er worden uitgevoerd in de komende sprint. In mijn geval heb ik deze meeting gehouden met de product owner en mijn begeleider.

Belangrijk was om logische user stories uit te kiezen die aan de basis van het eindproduct staan. Deze user stories moesten een werkend product opleveren en voldoen aan de wensen en eisen van de product owner.

Mijn taak in deze meeting was de rol van teamleider en Scrum Master. Dat betekent dat ik de meeting faciliteerde en zorgde voor het organiseren van de vergadering. Daarnaast was mijn rol ook die van ontwikkelaar en daarbij nauw betrokken bij de keuzes van de user stories. Mijn voornaamste afwegingen bij het kiezen van de user stories lagen voornamelijk bij de mogelijkheid om de user story al zo vroeg in het stadium te ontwikkelen.

Zou de user story niet te veel relaties met andere onderdelen hebben die beter eerst ontwikkeld kunnen worden voordat het gewenste user story opgepakt kan worden. Deze afweging maakte ik dan niet zo zeer vanwege gemakzucht of technische uitdaging, maar om de uren benodigd voor de ontwikkeling zo laag mogelijk te houden.

De user stories die we in deze meeting hebben uitgekozen voor de komende sprint zijn geplaatst in het sprint backlog. Dit register bepaalt de werkzaamheden voor de komende sprint en zal gebruikt worden om de "burndown chart" op te bouwen.

De gekozen user stories waren:

Opzetten van de toolboxmodule
Opzetten selenium testframework
Opzetten JUnit test framework
De cms pagina moet het fornulier kunnen opbouwen en tonen aan de gebruiker
Een beheerder moet een formulier kunnen toevoegen
Een beheerder moet een bestaand formulier kunnen bewerken
Een beheerder moet een formulier kunnen verwijderen
Een beheerder moet een element kunnen toevoegen aan een formulier
Een beheerder moet een element van een formulier kunnen bewerken
Een beheerder moet een element van het formulier kunnen verwijderen
Een beheerder moet de positie/volgorde van een element kunnen wijzigen
Een beheerder moet de eigenschappen van een element kunnen bewerken
Een beheerder moet een formulier kunnen invoegen op een willekeurig...
Een beheerder moet kunnen bepalen wat voor grid er gebruikt word
Een beheerder moet een bedank tekst kunnen opgeven
Afhandelingsactie: Ingevuld formulier kunnen opslaan in database

Een beheerder moet in Toolbox een overzicht van aanwezige formulieren inzien
Een beheerder moet alle ingevulde formulieren kunnen bekijken
Ondersteunen element: Text, Hidden, Password & Textarea input
Ondersteunen element: Radio/Select/Checkbox
Ondersteunen element: Submit & Reset
Een beheerder moet de eigenschappen van een formulier kunnen wijzigen

8.2 Ontwerpen domeinmodel

Aan het begin van deze sprint heb ik veel tijd besteed aan het ontwerp van het systeem. De gekozen user stories representeerden al een flink spectrum van het uiteindelijke systeem, dus nam het ontwerp ook veel tijd in beslag.

Ik heb dan ook de tijd genomen om goed alle elementen van het systeem te identificeren en ze de juiste plek binnen het systeem te geven. Belangrijke uitgangpunten voor mij waren het SOLID principe. SOLID is een acroniem wat staat voor de volgende vijf object-georiënteerde ontwerp principes:

- Single Responsibility Principe
- Open/Closed Principle
- Liskov Substitution Principle
- Interface Segregation Principle
- Dependency Inversion Principle

Het SOLID principe is dus een verzameling van vijf ontwerp principes die door ze tijdens het ontwerpen van een object-georiënteerde applicatie te volgen een goed OO ontwerp op zou moeten leveren. Door een applicatie te ontwikkelen op basis een correct OO ontwerp zal de applicatie eenvoudiger in het onderhoud zijn en kwalitatief van hogere kwaliteit zijn.

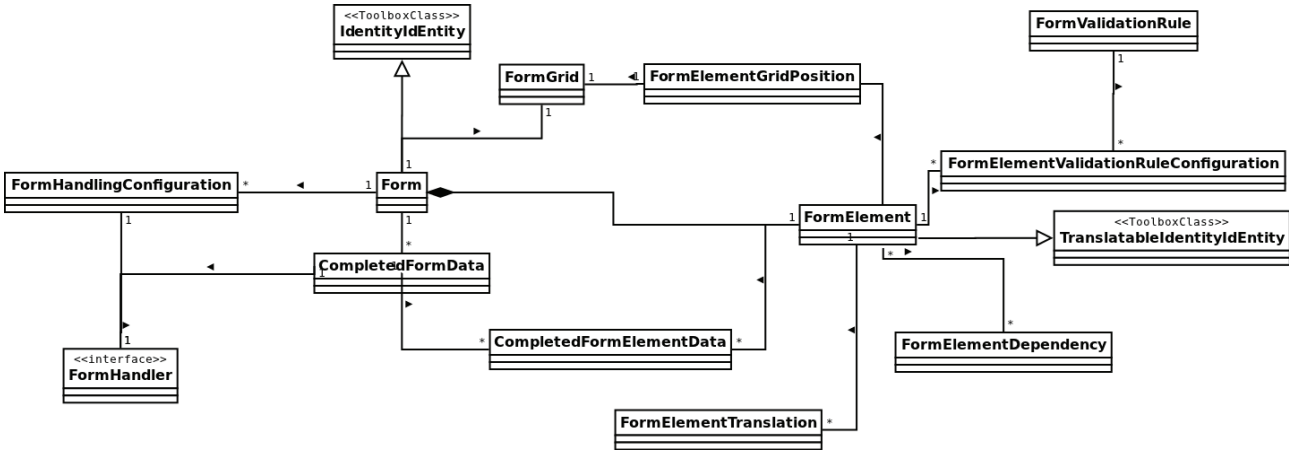
Door het ontwerp te laten voldoen aan deze vijf software ontwikkeling principes is het waarschijnlijker dat het systeem eenvoudiger in het onderhoud en uitbreid baar is. (SOLID)

Ook deze SOLID principes zijn een onderdeel van de Agile methodiek.

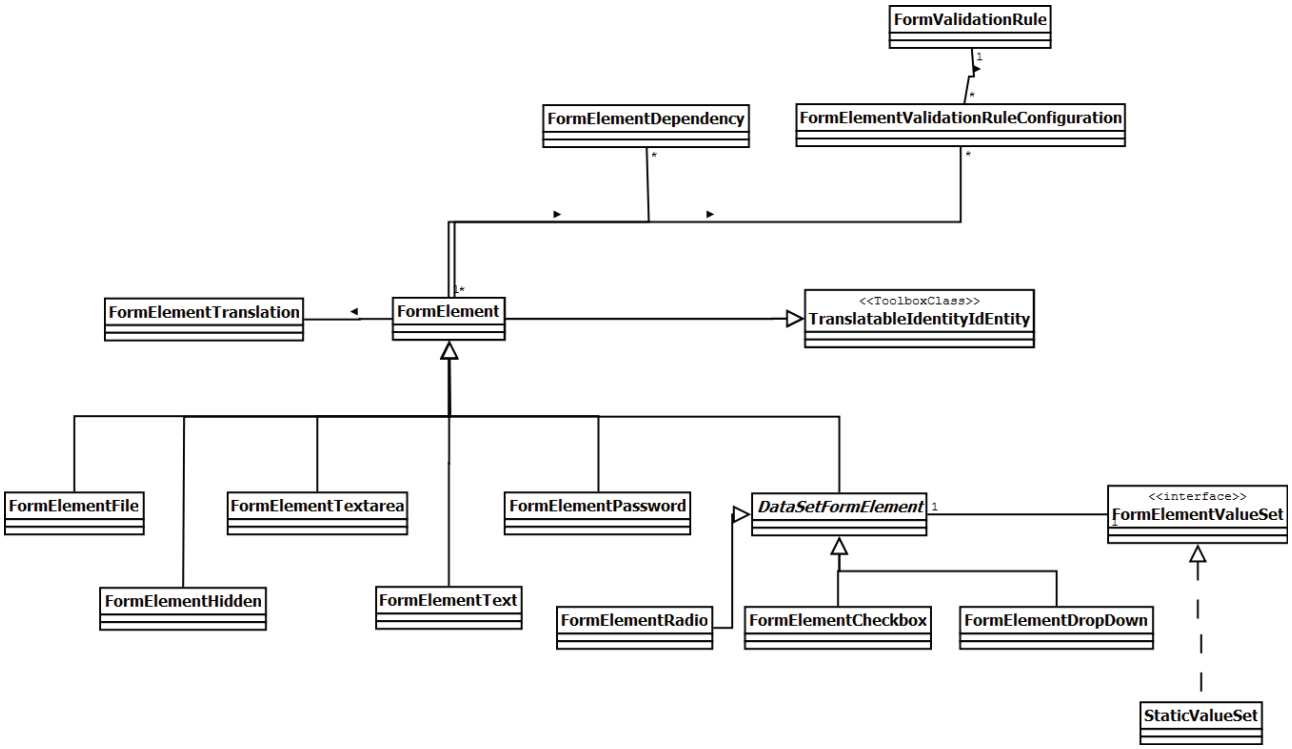
Met deze principes in acht genomen ben ik begonnen met het maken van het ontwerp. Extra moeilijkheidsgraad hierin was het rekening houden met de mogelijkheden van Toolbox. Hoewel het prettig is om op een bestaand systeem te bouwen, neem het ook enkele beperkingen mee.

Zo zijn er binnen Toolbox enkele basis klassen beschikbaar waarvan afgeleid moet worden om de integratie met andere componenten mogelijk te houden. In mijn geval betekent dat modellen afgeleid moeten worden van de IdentityIdEntity of TranslatableIdentityIdEntity class. Zoals de naam al weggeeft is laatst genoemde voor entiteiten die vertaalbaar moeten zijn.

Het domeinmodel voor deze sprint ziet er als volgt uit:



Belangrijkste elementen in het ontwerp zijn de klassen Form en FormElement. Deze twee elementen brengen een groot deel van de functionaliteit samen en vormen de koppelfactoren tussen de andere elementen.

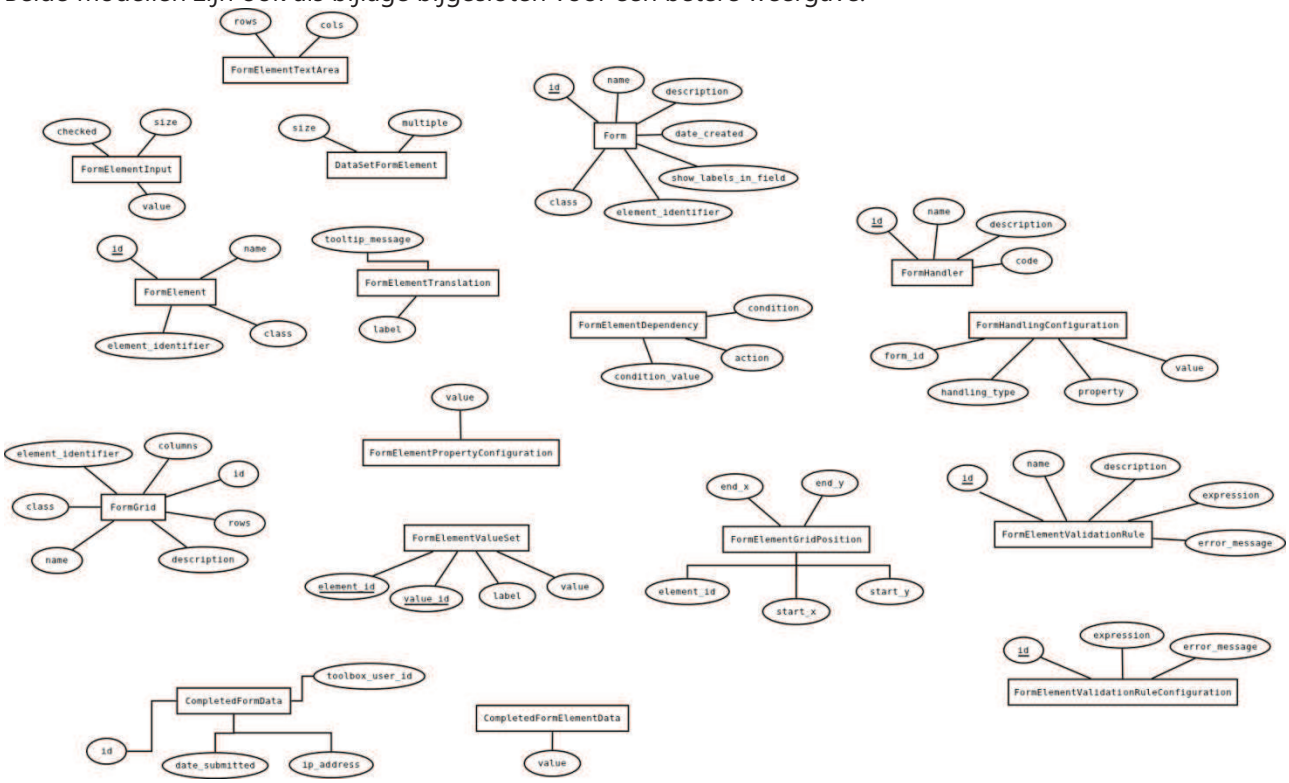


Het ontwerp heb ik twee keer besproken met mijn begeleider Carlo Vollebregt en collega Dennis van Holsteijn. Door met zijn drieën over het ontwerp te praten en te discussiëren hebben we een domeinmodel neergezet wat naar onze mening goed toepasbaar is op de nieuwe module en ook in de toekomst overeind zal blijven staan bij uitbreidingen en onderhoudswerkzaamheden.

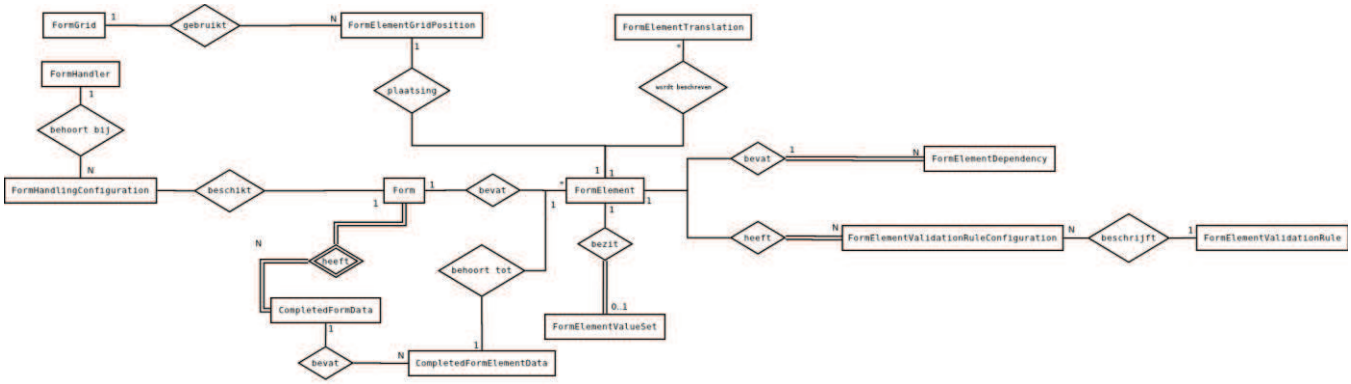
8.3 Ontwerpen datamodel

Nu het domein van de applicatie duidelijk is, was de volgende taak het bepalen wat er van iedere entiteit opgeslagen moest worden. Hiervoor heb ik een ERD model ontworpen die dit grafisch representeert. Om het model enigszins overzichtelijk te houden heb ik de relaties gescheiden van het beschrijven van de attributen van iedere entiteit.

Beide modellen zijn ook als bijlage bijgesloten voor een betere weergave.



Bovenstaand model laat de diverse entiteiten zien die we kennen binnen de applicatie en in de database opgeslagen moeten worden. Van ieder entiteit is beschreven de attributen die in de database gebruikt worden. De attributen die vluchtig zijn, de zogeheten transient attributen, staan niet in dit model beschreven.



Bovenstaand diagram laat de relaties in de database zien tussen de diverse entiteiten. Hierbij is te zien dat er gebruik gemaakt wordt diverse lijnen om de relaties aan te geven. De dubbele lijnen in het model geven aan dat de relatie totaal is en dat de relatie minimaal 1 keer moet voorkomen.

Het datamodel heeft mij en mijn begeleider geholpen om een goed beeld van de applicatie te krijgen en of we voor het bouwen van de applicatie op de goede weg waren.

8.4 User stories uitwerken

Met het ontwerp van het domeinmodel gereed kon er gestart worden met de daadwerkelijke ontwikkeling van het systeem.

Na het opzetten van de nieuwe module binnen de Toolbox- en ontwikkelomgeving was het zaak om stap voor stap aan de slag te gaan met de user stories.

Binnen Eleven proberen we zoveel mogelijk gebruik te maken van Test Driven Development. Deze methodologie houdt in dat we volgens het paradigma werken om eerst een unit test te schrijven, te verifiëren dat deze faalt om hem vervolgens succesvol te laten verlopen. Door volgens het TDD paradigma te werken is zal de uiteindelijke code duidelijker van structuur zijn en ook daadwerkelijk maar 1 verantwoordelijkheid te hebben.

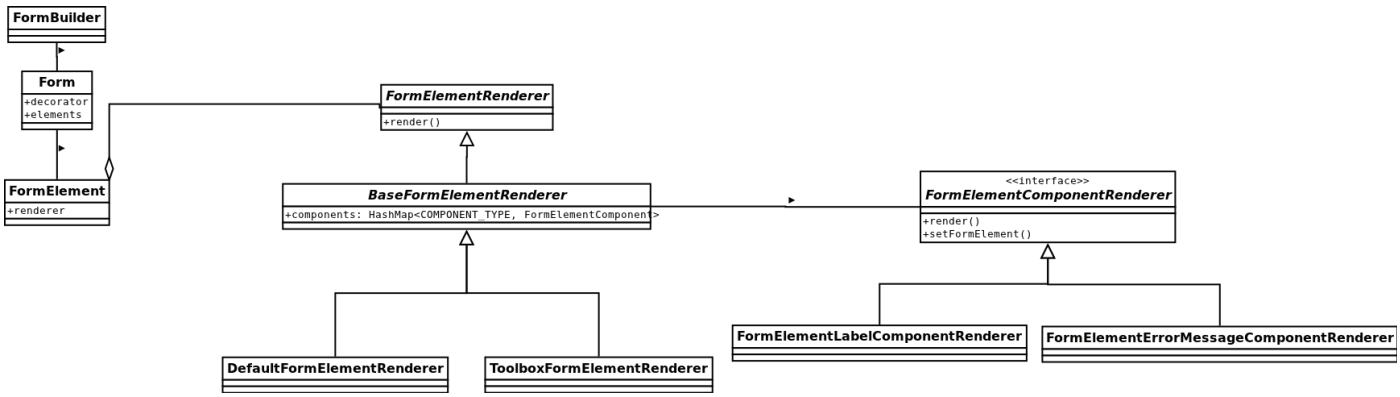
Ook tijdens de ontwikkeling van de eerste sprint heb ik geprobeerd zoveel mogelijk volgens TDD te werken. Bij sommige user stories was dit niet te doen omdat deze zo specifiek Toolbox user functionaliteit aanspraken dat ik te lang zou bezig zijn om deze testen te schrijven. De generatie van de formulieren en de elementen aan de voorkant zijn wel volgens het TDD principe ontwikkeld.

Het implementeren van de FormElement entiteiten binnen Toolbox was geen gemakkelijke opgave. Doordat we een strak model hebben ontworpen met een aantal abstracte hoofdklassen die nog eens door andere klassen geovererft worden, zaten we behoorlijk aan de grenzen van de huidige mogelijkheden binnen Cool. Niet dat het niet mogelijk was, maar het was simpelweg nog niet eerder gedaan. Aan mij de taak om dit wel voor elkaar te krijgen. Er heeft dan ook wel wat tijd en uitzoekwerk gezeten in het aanpassen van de Cool library zodat deze nieuwe constructies ondersteund werden.

Het aanpassen van de Cool library heb ik vooral met begeleiding van mijn technische begeleider gedaan. Dit omdat Cool een flinke library is met een behoorlijke historie. Er zijn veel onderdelen betrokken bij sommige functionaliteiten die ik moest aanpassen. Mijn begeleider is een van de architecten van dit systeem en kon mij dan ook goed helpen om de functionaliteiten en de code in het juiste plaatje te plaatsen. Dit zorgde ervoor dat wijzigingen die ik deed ook weinig impact hadden op overige functionaliteiten en implementaties van Cool.

Omdat aanpassingen in onze Cool library impact hebben op al onze projecten heb ik ook een aantal regressietesten gedraaid met zeven verschillende projecten om te controleren of de aanpassingen geen applicaties braken.

Eenmaal bij de user stories aangekomen die het weergeven van het formulier in de website behelzen, ben ik eerst naar de tekentafel gegaan om het model goed te tekenen.



Belangrijke eis in het ontwerp was dat de renderer class die het element tekent door middel van configuratie gewijzigd kan worden. Renderen is het genereren van de HTML die benodigd is om het formulier weer te geven. Bij iedere aanvraag van het formulier wordt het formulier real time en opnieuw gegenereerd.

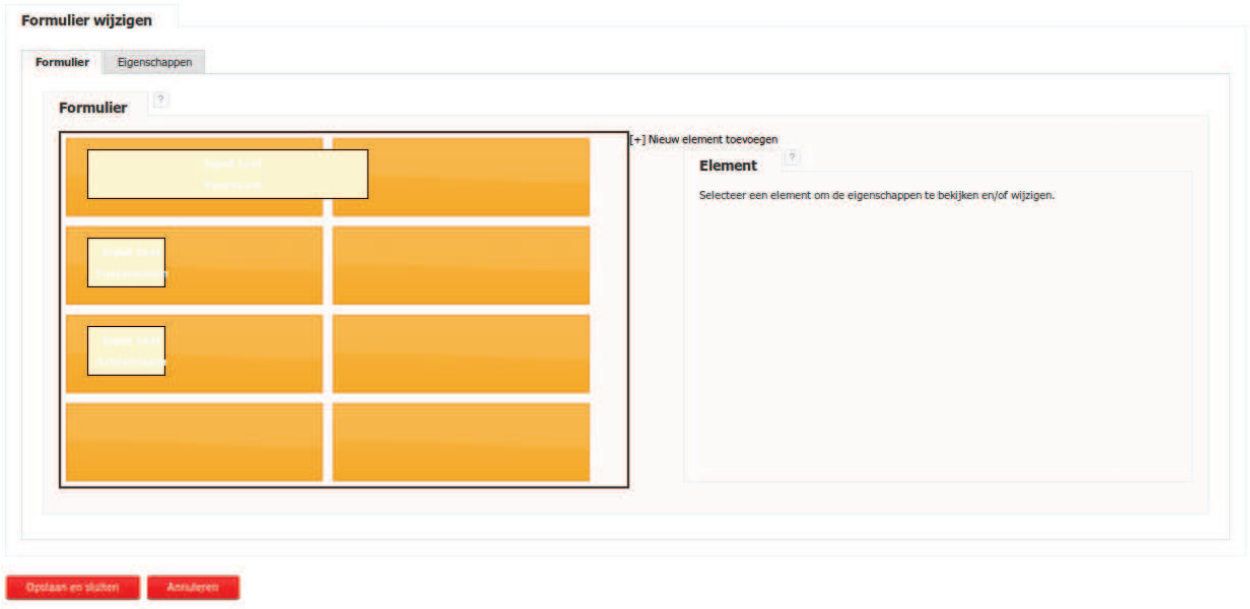
Ieder element bestaat, gezien vanuit een abstract niveau, uit een aantal componenten. Deze componenten kunnen geïdentificeerd worden als:

- Label
- Element
- Tooltip
- Foutmelding
- Wrapper

Ook deze componenten hebben hun eigen renderer. Zo blijft er voor iedere website of applicatie de mogelijkheid om invloed te houden op de weergave van de elementen.

8.5 De Toolbox user interface

Het belangrijkste scherm van de module is het bewerkscherm van het formulier. In dit scherm zal een beheerder de meeste tijd besteden om het formulier op te stellen en te wijzigen. Juist in dit scherm zullen de gekozen kwaliteitsattributen tot uiting moeten komen. Voor de opbouw van dit scherm hebben we in deze sprint de mockup als leidraad genomen.



TOOLBOX

Bovenstaande screenshot geeft een indruk van de user interface ten tijde van sprint 1. Net zoals de mockup bevat het scherm over twee hoofdelementen. Links is dat het grid waarop de element geplaatst kunnen worden. Aan de rechterkant is dat het tabbladen scherm waar de eigenschappen van ieder element gewijzigd kunnen worden. Tenslotte is het geheel verpakt in eigen tabblad zodat men de eigenschappen van het formulier in een ander tabblad kunnen wijzigen.

De structuur van het gebruik van tabbladen ligt in de lijn zoals Toolbox ook in andere modules opgezet is. Het voldoet daarmee ook aan de verwachting van een Toolbox gebruiker.

8.6 Selenium tests

Een van de gebruikte testtechnieken was het testen van de GUI. Hiervoor heb ik gebruik gemaakt van de open-source applicatie Selenium. Selenium is een test framework die gebruikt kan worden in een variëteit van ontwikkeltalen.

Met Selenium kun je testen uitvoeren alsof je in een browser zit, omdat Selenium ook daadwerkelijk een browser opstart. Hierdoor kun je dynamiek op een pagina ook daadwerkelijk testen.
Zo is het mogelijk om een test te schrijven of het inlogscherm ook daadwerkelijk tevoorschijn knop komt als de inlogknop wordt ingedrukt of dat als men een ongeldig e-mail adres invult in het e-mailadres het validatiebericht ook daadwerkelijk schrijft.

De bedoeling is dat de testen geschreven dan wel begrepen kunnen worden door de product owner. Daarvoor schrijven we de testen zoveel mogelijk in normaal Engels en stoppen we alle technische details weg in een lagere laag.

Deze lagere laag noemen we ook wel een domain-specific language oftewel DSL. Deze DSL heeft alleen betrekking op de functionaliteiten van de formulierenmodule en heeft dan ook alleen op dit domein betrekking.

Domain-specific languages zijn niet nieuw en worden veelvuldig gebruikt, denk hierbij bijvoorbeeld aan SQL en HTML. Het tegenovergestelde van domain-specific languages zijn generic-purpose languages. Deze talen kunnen gebruikt worden om een breed spectrum van applicaties te ontwikkelen. Voorbeelden van dit soort talen zijn Java en C. (Wiki-DSL)

Om het schrijven van de Selenium tests zo eenvoudig en duidelijk mogelijk te maken voor een persoon die geen technische achtergrond heeft, maar wel kennis van het probleem domein heb ik een DSL geschreven. Van deze DSL wordt gebruik gemaakt in de Selenium testen zodat een test beschrijft wat er gebeurt zonder enige technische implementatie.

Ter illustratie zijn er op de volgende pagina een aantal voorbeeld methodes van de DSL gepresenteerd.

Een voorbeeld van het gebruik van deze DSL in een test is in onderstaande code listing zichtbaar.

```
public class AcceptanceTestFormRendering extends FormWebTestDSL {
    1.
    2.   @Test
    3.   public void testCorrectFormRendering() {
    4.      openUrl("/site/nl/projecten/websites");
    5.      hasFormWithId("first-form");
    6.      hasFormElementBelongingToFormWithId("firstname", "first-form");
    7.      hasFormElementBelongingToFormWithName("Tussennaam", "first-form");
    8.      hasFormElementBelongingToFormWithName("Opmerkingen", "first-form");
    9.    }
    10.       }
```

De basisklasse FormWebTestDSL verbergt de implementatie van de Selenium code die voor de juiste functionaliteit zorgt. De DSL zorgt ervoor dat de ontwikkelaar geen weet heeft van deze lastige code, maar eenvoudig kan bereiken wat hij functioneel wilt.

De DSL maakt gebruik van XPath om de juiste elementen op te sporen. XPath is een techniek die op basis van gegeven criteria elementen uit een bron bestand kan halen zolang de bron maar op basis van een XML-achtige is opgesteld. In ons geval gaat het om een HTML bron die van de XML taal is afgeleid.

XPath kan gebruik maken van criteria die controleren of een element van een bepaald type is, of dat hij een bepaalde class bevat of dat de identifier van het element ‘addres’ is.

Alhoewel XPath een flexibele en functioneel uitgebreide query taal is, heeft het wel als nadeel dat het relatief langzaam is en dat het zoeken van de juiste elementen in een grote bron een tijdrovende operatie is.

De methode is de basisklasse zijn voorzien van Javadoc documentatie zodat de ontwikkelaar door middel van het commentaar en de IDE tijdens het schrijven van de tests de bedoeling van de tests eenvoudig kan inzien.

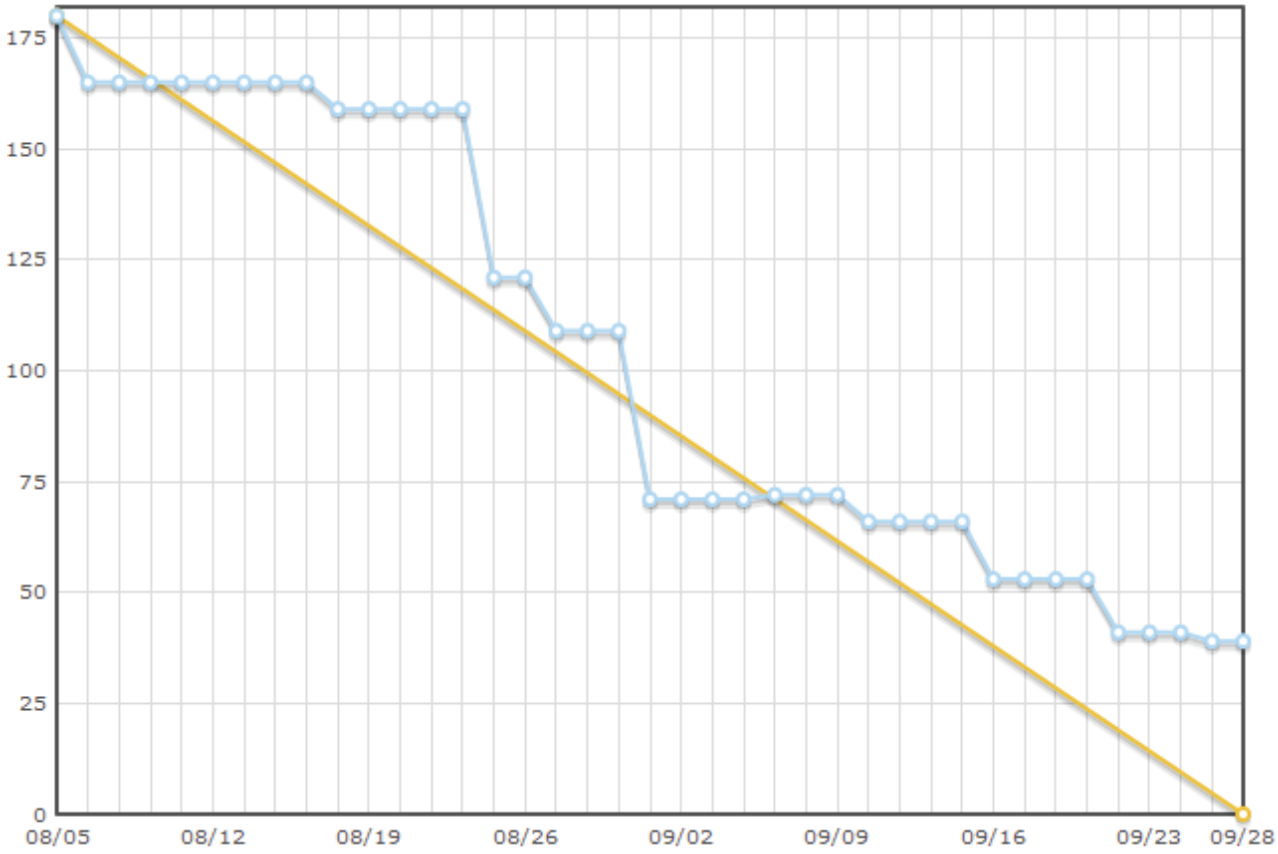
Korte impressie FormWebTestDSL

```
1.  /**
2.   * This class acts as a domain specific language to make the implementation so easy that a
3.   * non-programmer should be able to understand the tests at once
4.   *
5.   * @author pderaaij
6.   *
7.   */
8.  public class FormWebTestDSL extends WebTest {
9.
10.     /**
11.      * Searches for an element with the given id within the form with the given id
12.      *
13.      * @param id The id of the element which should be contained by the form
14.      * @param formId The id of the form which should contain the element
15.      */
16.     public void hasFormElementBelongingToFormWithId(String id, String formId) {
17.         assertTrue(selenium.isElementPresent("//form[@id='" + formId + "']//*[@id='" + id + "']"));
18.     }
19.
20.     /**
21.      * Searches for an element with the given name within the form with the given id
22.      *
23.      * @param id The name of the element which should be contained by the form
24.      * @param formId The id of the form which should contain the element
25.      */
26.     public void hasFormElementBelongingToFormWithName(String name, String formId) {
27.         assertTrue(selenium.isElementPresent("//form[@id='" + formId + "']//*[@name='" + name + "']")
    );
28.     }
29.
30.     /**
31.      * Looks on the page if an form element is present with the given id
32.      *
33.      * @param id The element identifier
34.      */
35.     public void hasFormWithId(String id) {
36.         assertTrue(selenium.isElementPresent("//form[@id='" + id + "']"));
37.     }
38.
39.     @Before
40.     public void setUp() throws Exception {
41.         // TODO make the URL server dependent (so we can execute this test class on the coffee)
42.         // TODO make it possible to specify the browser, for cross browser testing
43.         setUp("http://localhost:8080", "*firefox");
44.     }
```

8.7 Wat ging er goed

Over de eerste sprint ben ik redelijk tevreden. Het ontwerpen van de module ging voorspoedig en door de samenspraak met mijn collega's ben ik dan ook tevreden over het model wat we neergezet hebben. Helaas moesten we door de soms moeizame implementatie van de module in Toolbox een aantal user stories doorschuiven naar de volgende sprint. Dit omdat er wat tijd verloren ging aan de implementatie die ik niet kon gebruiken voor het uitwerken van de user stories.

Onderstaande burndown chart laat een grillig verloop van de Sprint zien. Dat komt deels omdat ik niet vijf dagen in de week achterelkaar aan het project kon werken. Daarnaast is goed te zien dat we in het begin veel aandacht aan het ontwerp van het domein- en datamodel besteed hebben en dat we daardoor geen user stories konden afronden. In de weken daarna maak je ruime sprongen omdat er a) grote user stories worden afgerond en b) omdat veel voorbereidend werk gedaan is en je vlot door kan met het project.



Zoals gezegd heb ik redelijk wat wijzigingen aan Cool moeten aanbrengen die in de planning onvoorzien waren. Ik heb daarom de user story omtrent het versiebeheer moeten doorschuiven naar de volgende sprint en dat verklaart ook waarom er in de laatste week nog flink wat punten openstaan.

De ontwikkeling van de eerste set user stories ging ook goed. De problemen met de implementatie van het model binnen Toolbox waren voorzien, hoewel ook een enkele keer frustrerend.

Een aantal keren ben ik in aanraking gekomen met werkzaamheden waar ik normaal gesproken niet zo veel mee doet. Een voorbeeld daarvan is het opzetten van de Selenium tests. Het is erg leuk om gebruik te maken van deze nieuwe technieken en ze onder de knie te krijgen.

8.8 Wat ging er niet goed

Na afloop van de sprint hebben we in de Sprint Retrospective Meeting niet alleen besproken wat er goed ging, maar ook wat er beter moest.

Zelf was ik niet tevreden over de mate waarop ik het project ook daadwerkelijke aanstuurde. Dit werd bevestigd door mijn begeleider. Dit kwam voornamelijk omdat de favoriete rol waar ik in dook die was van software architect of software ontwikkelaar. Hierbij vergat ik regelmatig de rol van projectleider die het project moet managen en een faciliterende rol heeft.

Om dit te verbeteren ga ik voor mijzelf een vast moment in de week blokken waarin ik mijzelf dwing om het project vanuit een projectmanager te bekijken en hoog boven de details te hangen. Dit doe ik aan de hand van en checklist waarin ik een aantal vragen omtrent het project moet beantwoorden. Een greep van punten uit deze checklist zijn:

- Hoeveel uur aan werk staat er nog open
- Kan ik de geplande tijd voor de formulierenmodule ook nog daadwerkelijk besteden
- Hoe verloopt de trend op de burndown chart
- Is de status van het project besproken met de Product Owner
- Is de voortgang besproken met de technisch begeleider

Het afstuderen is onderdeel van mijn deeltijd opleiding Informatica. Normaal gesproken werk ik 40 uur in de week bij Eleven en voor het afstuderen had ik met mijn werkgever een overeenkomst dat ik enkele dagen in de week aan het afstudeerproject kon werken.

Dan is het heel moeilijk om op de dagen dat je werkt aan het afstudeerproject de focus te houden op het afstudeerproject. Doordat je op kantoor werkzaam bent is het zo eenvoudig om gestoord te worden door collega's met vragen of spoedklusjes.

Deze sprint ben ik hier te makkelijk mee omgegaan en heb ik te vaak het afstuderen los gelaten om dan maar die vraag of spoedklus voor mijn collega op te pakken. In de komende sprints moet ik egoïstischer worden en meer voor het afstuderen kiezen, wil ik het binnen de gestelde termijn halen.

9. Sprint 2

In sprint 2 is het systeem verder ontwikkeld en uitgebreid met nieuwe functionaliteiten. Een belangrijke user story die in deze sprint is opgepakt is de toepassing van het juiste design van de module in Toolbox. Deze sprint liep van 3 oktober 2011 tot 11 november 2011.

9.1 Spring planning meeting

In deze Scrum planning hebben we besloten om de user stories uit sprint 1 die waren doorgeschoven als eerste in de sprint backlog geplaatst. Daarnaast besloten we om de user stories die het versiebeheer en het nieuwe design beschrijven op te pakken.

De sprint backlog zag er als volgt uit:

Een beheerder moet middels versiebeheer de juiste versie kunnen selecteren
Een beheerder moet een of meerdere validatieregels kunnen instellen op een element
Een beheerder moet voor een element een helptekst/tooltip kunnen op een element
Een beheerder moet voor iedere validatieregel een error message kunnen opgeven
Een bezoeker die een formulier invult moet realtime gevalideerd worden
Een beheerder moet afhankelijkheden tussen de elementen aan kunnen geven
Een beheerder moet één of meer afhandelingsacties kunnen registreren op een formulier
Afhandelingsactie: Ingevuld formulier kunnen mailen
Ondersteunen element: Datepicker/Calendar
Ondersteunen element: Captcha
Implementeren van nieuw design
Overzicht ingevulde formulieren inzien via formulierenoverzicht

9.2 Versiebeheer van formulieren

Een van de wensen van de product owner was het hebben van versiebeheer op de formulieren. Voor de gebruiker moest het eenvoudig zijn om eventuele wijzigingen ongedaan te maken en terug te gaan naar een vorige versie van het formulier.

Dit versiebeheer moest actief zijn op alle verschillende onderdelen van een formulier. Dus niet alleen de verschillende elementen, maar ook de eigenschappen, validaties en afhankelijkheden van ieder element. Indirect gevolg is dat bij het ophalen van ingevulde formulieren de juiste versie van het formulier erbij gepakt moet worden.

Het monitoren van entiteiten of er een nieuwe versie opgeslagen dient te worden noemen we ook wel Auditing. Auditing betekent het evalueren van een object of het valide en betrouwbaar is.

9.2.1 Hibernate Envers

Binnen Hibernate is er al een sub project actief die Auditing technieken biedt aan een ontwikkelaar, namelijk Hibernate Envers. Het voordeel is dat deze library actief werkt binnen de lagen van Hibernate en dat het een bewezen techniek is. De library is al in gebruik bij een flink aantal andere projecten en heeft een actieve community die mee ontwikkelt aan de library en ondersteuning biedt.

Om te controleren of Envers voor de formulierenmodule de juiste library was voor onze wensen en eisen heb ik een test project geschreven om te zien of de library voldoet aan onze wensen. Dit testproject is gebaseerd op de volgende requirements:

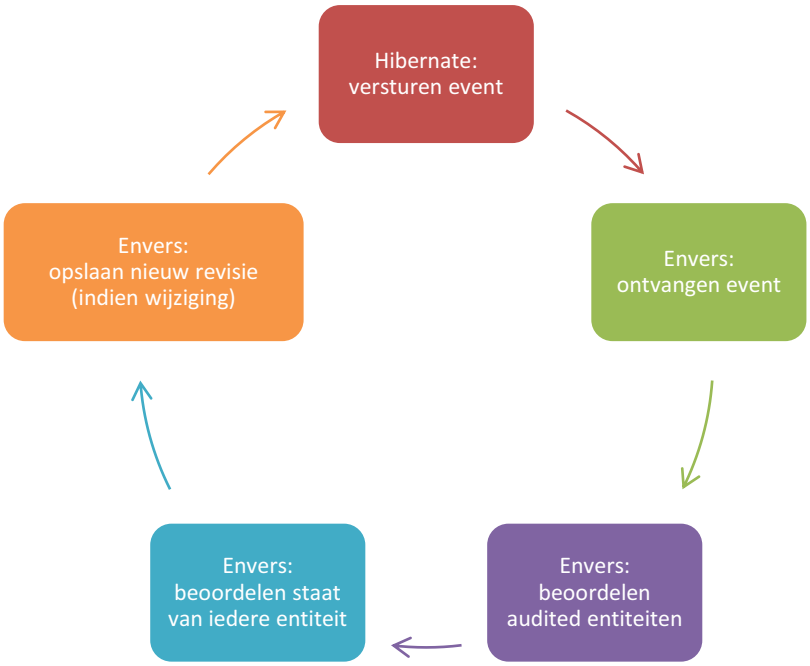
- De library kan een entiteit volledig auditen
- Het is mogelijk om bepaalde attributen te negeren voor auditing
- De library ondersteunt one-to-many en many-to-one relaties.
- Als de root entitieit wordt geladen, worden ook alle subentiteiten geladen.

Het testproject is een simpel project dat heeft aangetoond dat aan alle eisen die we stelden aan Envers is voldaan. In overleg met mijn technisch begeleider hebben we dan ook besloten om van Envers gebruik te maken als auditing library.

9.2.2 Globale werkwijze van Envers

Envers werkt door in te haken op events die de Hibernate library afvuurt op het moment dat een entiteit wordt opgeslagen, opgehaald of een set van entiteiten word gereconstrueerd. Aan deze events koppelt Envers luisteraars die actief worden zodra een event wordt ontvangen.

Als voorbeeld kijken we wat Envers doet met een entiteit als het opgeslagen wordt. Het event bevat de entiteiten waar een persist actie is uitgevoerd. Envers beoordeelt of deze entiteiten zijn geannoteerd met @Audited wat aangeeft dat we willen dat Envers iets met deze entiteit doet. De entiteiten die beschikken over de juiste annotatie worden vervolgens beoordeeld of de entiteit in deze transactie gewijzigd is ten opzichte van zijn laatst opgeslagen staat. Als er een wijziging plaats heeft gevonden op de entiteit, dan slaat Envers automatisch een nieuwe revisie op van de entiteit in zijn eigen Audit database table. In schema ziet dat er als volgt uit.



9.2.3 Opslag van revisies

Alle revisies van een entiteit moeten door Envers opgeslagen kunnen worden. Om die revisie informatie op te slaan maakt Envers automatisch een extra tabel aan voor iedere entiteit. Deze tabellen worden binnen Envers Audit tabellen genoemd en hebben exact dezelfde structuur als dat de originele tabel heeft. Het enige wat Envers aan deze tabellen toevoegt zijn de kolommen 'REV' en 'REVTYPE'. De kolom 'REV' wordt gebruikt om de revisie identificatie van de opgeslagen revisie op te slaan. De kolom 'REVTYPE' bevat een waarde die aangeeft of deze revisie een INSERT of een UPDATE is. Visueel gezien ziet dat er als volgt uit:

Form
Id
Name
Description
elementClass
elementIdentifier

Form_AUD
Id
Name
Description
elementClass
elementIdentifier
REV
REVTYPE

9.2.4 Implementatie van Envers

De implementatie van Envers heeft wel wat tijd gevergd. Het auditing principe is niet alleen voor mij nieuw, maar ook voor mijn collega's bij Eleven. Het is dan op sommige momenten ook een flinke zoektocht geweest om tot de juiste oplossing en implementatie te komen.

Zo was het een probleem om op de juiste wijze de EventListeners van Envers te registreren aan de lifecycle van Hibernate. Toolbox beschikt namelijk over een eigen implementatie van de Hibernae Lifecycle waardoor we niet de configuratie zoals beschreven in de documentatie konden gebruiken. Dit vergde van mij een flinke zoektocht door de diepste lagen van Toolbox om de juiste registratiepunten te vinden.

Ander groot pijnpunt was om er voor te zorgen dat de auditing alleen maar betrekking had op de entiteiten die behoren tot formulierenmodule. We wilden er voor zorgen dat Envers alleen weet had van entiteiten uit de formulierenmodule zodat het geen effect had op alle Toolbox implementaties zonder de formulierenmodule.

In een aantal entiteiten gaf dit enkele problemen omdat zij directe relaties hebben met entiteiten uit Toolbox. Zo heeft de entiteit FormElementTranslation een attribuut van het type Language. Deze Language entiteit komt uit de Toolbox package en willen we niet betrekken in de Envers configuratie van de formulieren module.

We willen voorkomen dat we Toolbox entiteiten auditeren zodat we niet onnodig informatie opslaan die we functioneel gezien toch niet zullen gebruiken. Daarnaast heeft iedere entiteit die we auditen gevolg voor de performance van de applicatie.

Vrijwel alle problemen die ik hierin tegenkwam waren redelijk snel en zoals gewenst op te lossen. Helaas kon ik in het geval van de Language entiteit niet voorkomen dat deze toch altijd en in ieder project geaudit moet worden. Dit betekent dat ieder Toolbox project nu een actieve Envers configuratie heeft en iedere persist actie van Hibernate nu door Envers beoordeeld word, onafhankelijk de formulierenmodule nu wel of niet aanwezig is.

Dit heeft te maken met de wijze waarop wij translations gebruiken binnen Toolbox en haar modules. Hierbij maken wij gebruik van een Map waarin de key gevormd word door Language. Deze key is een elementair onderdeel in de database om de Map op te slaan.

De achterliggende technische reden is dat Envers zijn informatie volledig uit de Audit tabellen haalt. De key in de Map worden bepaald door middel van een one to many relatie tussen FormElementTranslation en de Language entiteit. Als, zoals bij ons het geval was, de key waarde niet geauditeerd wordt dan zal de SQL query die gegenereerd wordt door Envers altijd een lege set opleveren.

9.2.5 Ophalen en bepalen van revisies

Om van een formulier te bepalen welke revisies er zijn en deze op te halen heb ik een speciale service klasse geschreven. In deze klasse is alle Envers specifieke code weggestopt zodat het voor een ontwikkelaar eenvoudig is om versie informatie van een formulier op te halen.

Een van de functies in deze service klasse is het ophalen van de verschillende revisies van een formulier:

```
1.  /**
2.   * Get a list of all @link{DefaultRevisionEntity} which represents all versions of this
       form
3.   *
4.   * Limited to 25 results max
5.   * @author Paul de Raaij <paul@eleven.nl>
6.   * @param form
7.   * @return
8.   */
9.  @SuppressWarnings("unchecked")
10. public List<DefaultRevisionEntity> findVersionsOfForm(Form form) {
11.         AuditReader
        auditReader = AuditReaderFactory.get(sessionFactory.getCurrentSession());
12.
13.         List<DefaultRevisionEntity> revisionInfos = new ArrayList<DefaultRevisionEntity>();
14.         List<Number> revisions = auditReader.getRevisions(Form.class, form.getId());
15.
16.         for (Number number : revisions) {
17.                 revisionInfos.add(auditReader.findRevision(DefaultRevisionEntity.class,
        number));
18.         }
19.
20.         for (int formElementId : form.getElements().keySet()) {
21.                 List<Number> elementRevisions = auditReader.getRevisions(FormElement.class,
        formElementId);
22.                 List<Number> newRevisionNumbers = ListUtils.removeAll(elementRevisions,
23.                         ListUtils.intersection(elementRevisions, revisions));
24.
25.                 for (Number number : newRevisionNumbers) {
26.                         revisionInfos.add(auditReader.findRevision(DefaultRevisionEntity.cl
        ass, number));
27.                 }
28.         }
29.
30.         Collections.sort(revisionInfos, new RevisionEntityComparator());
31.
32.         if (revisionInfos.size() > 25) {
```

```
33.                 return revisionInfos.subList(revisionInfos.size() - 25,
        revisionInfos.size());
34.         }
35.
36.         return revisionInfos;
37. }
```

Alle specifieke code om Envers aan te roepen en de juiste informatie op te halen is voor de ontwikkelaar op deze manier netjes weggestopt. Het enige wat een ontwikkelaar nu hoeft te doen is een implementatie zoals hieronder.

```
1. @Autowired //Spring methode om een instantie van onderstaande class te injecten
2. private FormVersioningService formVersioningService;
3.
4. @Autowired
5. private FormDao formDao;
6.
7. Form form = formDao.find(2); //instantie van formulier uit DB opvragen
8. formVersioningService.findVersionsOfForm(form); // revisies ophalen
```

9.3 Beheren van validatie en afhankelijkheden in het front-end

Ieder formulier dat getoond word aan de voorkant van de site beschikt over nogal wat dynamiek. Zo kan ieder element beschikken over één of meerdere validatieregels en of heeft het element een afhankelijkheid met een ander element.

Voor het mogelijk maken van deze dynamiek maken we hevig gebruik van javascript. Door middel van de javascript kunnen we de gebruiker direct een foutmelding geven als het element incorrect ins ingevuld. Daarnaast kunnen we elementen verbergen of juist tonen als dat gewenst wordt door een ander element.

Om het wiel niet opnieuw uit te vinden maak ik voor de validatie van de elementen gebruik van de jQuery validate plugin. Deze plugin maakt het mogelijk om validatieregels op te geven voor een element. De plugin beschikt over een waslijst aan standard validatieregels die al veelvuldig gebruikt worden. Dit versimpelt de implementatie voor ons en geeft enige garantie over de juistheid van de validaties.

Om de afhankelijkheden te laten functioneren heb ik wel een eigen plugin geschreven om dit mogelijk te maken. Hiervoor konden we namelijk geen vergelijkbaar materiaal voor vinden die ook maar enigszins deed wat we wilden.

Om al deze dynamiek te registreren in het formulier konden we twee kanten op. Een mogelijkheid was het registreren van alle gewenste acties op de jQuery validate plugin en de zelf geschreven dependency plugin met het tekenen van het formulier.

Nadeel van deze methode is dat de registratie verspreid raakt over diverse plekken in de element renderers. Zo zal de afhankelijkheden in de renderer van het element beschreven worden, maar de validatie van de elementen weer tijdens het renderen van het formulier zelf.

Dit zou een rommeltje worden, daarnaast zou je als iemand een nieuwe renderer wilt schrijven voor de formulierenmodule veel kennis moeten hebben van de plugins om ze juist te configureren.

Daarom heb ik gekozen voor de andere optie en dat is het maken van een aparte JavaScript manager waaraan alle acties geregistreerd worden. Dit zorgt ervoor dat het registreren van de acties in de renderers eenvoudig wordt en vanuit één plek kan gebeuren. Daarnaast schermt het ook de implementatie af en is het relatief simpel om de validatie plugin in de toekomst te wijzigen voor een andere implementatie. Het zorgt er dus voor dat binnen de applicatie de diverse verantwoordelijkheden decoupled blijven van elkaar.

Het registreren van een validatieregel kan daardoor heel eenvoudig via een JavaScript declaratie zoals:

```
1.  <script type="text/javascript">
2.  formManager.addValidationRule('11', 'pattern', '^(([^<>()[\\]\\.,;:\\s@\"]+(\\.[^<>()[
        \\]\\.,;:\\s@\"]+)*)|(\".+\"))@((\\[[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.[0-
        9]{1,3}\\])|(([a-zA-Z\\-0-9]+\\.)+[a-zA-Z]{2,}))$', 'Vul een geldig e-mailadres
        is');
3.  </script>
```

Het toekennen van meerdere validatieregels is ook mogelijk.

```
1.  <script type="text/javascript">
2.  formManager.addValidationRule('1', 'minlength', '5', 'Vul minimaal 5 karakters in');
3.  formManager.addValidationRule('1', 'maxlength', '12', 'Vul maximaal 12 karakters in');
4.  formManager.addValidationRule('1', 'required', 'true', 'Dit is een verplicht veld');
5.  </script>
```

Ook de afhankelijkheden worden op deze manier geregistreerd.

```
1.  <script type="text/javascript">
2.  formManager.addDependency('2', '1', 'HIDE', 'CONTAINS_VALUE', 'test');
3.  </script>
```

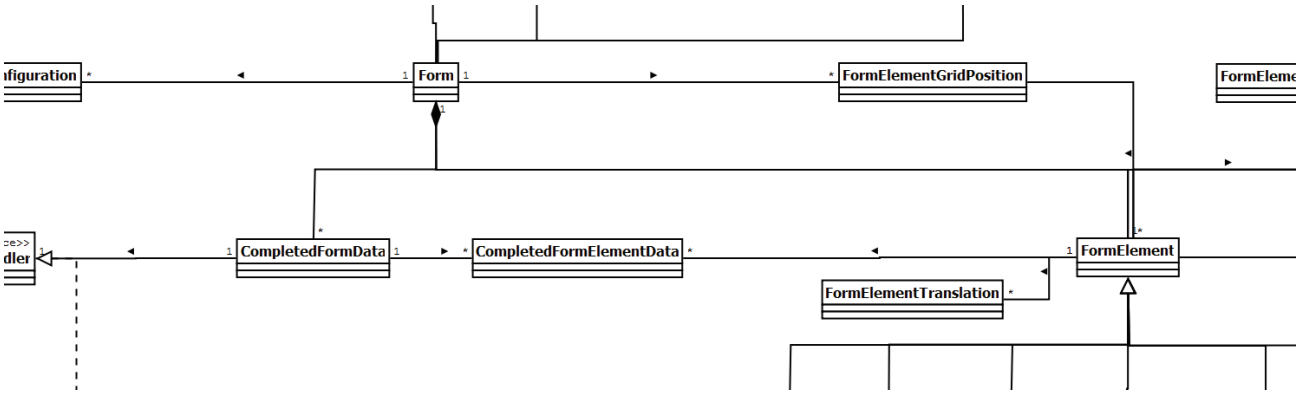
Overigens worden deze javascript statements in de renderers dynamisch opgebouwd. De voorbeeld code als hier getoond is het resultaat van deze dynamische constructie.

9.4 Afhandelen van ingevulde formulieren

Zodra een bezoeker een formulier invult moeten er door de applicatie nogal wat acties uitgevoerd worden. Na een druk op de verstuur knop door de bezoeker gebeurt er het volgende:

9.4.1 Binden van data

Stap 1 in deze cyclus is het prepareren van de invoer die door de bezoeker zijn ingevuld in de juiste entiteiten. Het gemaakte datamodel verwacht namelijk dat we de uiteindelijke invoer opslaan in de entiteit CompletedFormElementData, zoals te zien in onderstaande uitsnede van het domeinmodel.



Het koppelen van de informatie die ingevuld is aan de juiste entiteiten is een techniek die ook wel data binding wordt genoemd. Het komt er op neer dat je doormiddel van databinding de informatie tussen twee informatiebronnen kan synchroniseren. In ons geval gaat het dan om het formulier element en het ‘value’ attribuut van CompletedFormElementData.

Spring Framework biedt ons automatisch de juiste data binding functionaliteit. Dit werkt out of the box en heeft maar een vereiste. De applicatie moet er zorg voor dragen dat de formulier elementen beschikken over de juiste (technische) naam. Door deze naamgeving kan Spring de juiste informatie koppelen aan de juiste elementen.

9.4.2 Valideren formulier

Als extra controle valideren we het formulier nogmaals. Dit gebeurt al in het front-end van de website, maar hier zijn we afhankelijk van de javascript functionaliteiten in de browser. Dit kan bewust of onbewust uitgeschakeld zijn in de browser van de bezoeker waardoor potentieel incorrecte formulieren worden ingezonden.

De validatie die we bij het verwerken van het formulieren uitvoeren is er dan ook een ter extra controle. Dit geeft een stukje extra zekerheid dat de informatie die ingezonden wordt ook echt correct is.

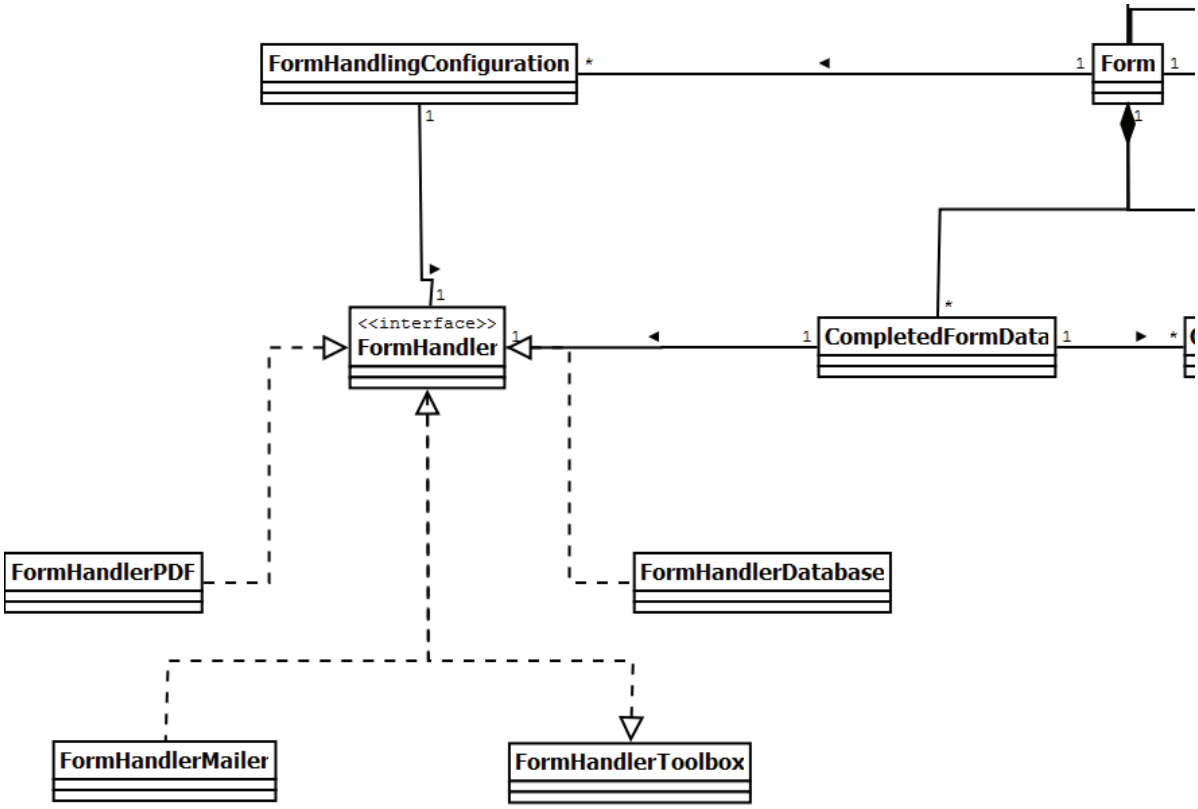
Als de informatie niet valide blijkt te zijn wordt de bezoeker teruggezonden naar het formulier met de juiste foutmelding en alle informatie die hij ingevuld heeft.

9.4.3 Uitvoeren afhandelingsacties

Nu we weten dat we correcte informatie hebben kunnen we iets met de informatie doen. De beheerder kan diverse afhandeling acties opgeven aan het formulier en in deze stap worden deze uitgevoerd.

Standaard word alle informatie opgeslagen in de database, dat is een keuze die wij gemaakt hebben in overleg met de product owner. De afweging om de informatie op te slaan is gemaakt zodat de klant altijd de informatie beschikbaar heeft.

Vertrouwt de beheerder alleen op de e-mail dan kan de informatie verloren gaan doordat de opgegeven e-mailbox niet beschikbaar was of iets dergelijks. Nu bieden wij de informatie altijd aan en zal de data nooit verloren zijn.



Voor iedere afhandeling actie die een beheerder kan opgeven is er een speciale entiteit FormHandlingConfiguration. Deze entiteit beschikt over de configuratie voor die afhandeling. In het geval van een e-mail actie is dit bijvoorbeeld het e-mail adres waar de informatie heen gezonden moet worden.

In deze stap worden alle configuraties opgehaald die bij het formulier horen en één voor één uitgevoerd.

9.4.4 Bedanktekst / Bedankpagina

Als alle stappen succesvol zijn verlopen dan moeten we dat ook de bezoeker laten weten. In deze laatste stap word of de bedanktekst getoond of de bezoeker naar een speciale pagina in de website geleid. Dit is afhankelijk van de configuratie van het formulier en ligt in handen van de beheerder.

9.5 Hallway testing

In deze fase van het project heb ik regelmatig gebruik gemaakt van Hallway testing om de gebruiksvriendelijkheid van de module te testen.

Hallway testing is een vorm van testen waarbij een willekeurig persoon van de gang wordt 'geplukt' en achter de applicatie wordt gezet met een korte opdracht. De testpersoon moet proberen deze opdracht te vervullen zonder enige hulp van de ontwikkelaar.

Het doel is om op zoek te gaan naar 'brick walls'. Problemen zo ernstig dat een gebruiker niet verder kan met de opdracht en zich op deze manier vastloopt in de applicatie. Daarnaast is het voor de makers ook erg leerzaam omdat ze direct zien hoe de oplossing gebruikt zal worden. De testpersoon bewandelt over het algemeen andere paden, die de ontwikkelaar door zijn ervaring niet zal bewandelen. (WIKI-Hallway)

Binnen dit project en deze sprint heb ik deze vorm van testen twee keer toegepast met een willekeurig persoon. Deze persoon was een stagiair van de Haagse Hogeschool die zijn afstudeerstage bij Eleven uitvoert.

Hij was een goede kandidaat voor de testsessies omdat hij geen kennis van Toolbox en de nieuwe formulierenmodule had. De opdracht die hij moest uitvoeren zou dan ook een echte ontdekkingstocht worden waarbij we veel konden leren over het gebruik van de nieuwe module.

De opdracht voor de testpersoon was als volgt geformuleerd:

```
Maken van formulier bestaande uit:
      Voornaam (tekstelement, mag niet leeg zijn, minimaal 2 karakters)
      Achternaam (tekstelement, mag niet leeg zijn, minimaal 2 karakters)
      Geslacht (radio element, verplicht; waarde man | vrouw)
      E-mail (tekstveld, geldig e-mail adres)
      Aanmelden voor nieuwsbrief( alleen zichtbaar als e-mail waarde bevat )
      Opmerkingen veld (tekstveld, verplicht veld)
      Verzendknop


Resultaat formulier moet gemaild worden naar webdevelopers@eleven.nl

Bedank tekst:
Bedankt voor je vraag, we nemen zo spoedig mogelijk contact met je op.
```

In deze opdracht zitten alle onderdelen die er ten tijde van deze sprint ontwikkeld zijn en zorgt ervoor dat de testpersoon door alle schermen heen moet klikken.

De test is de eerste maal uitgevoerd begin november. Tijdens het uitvoeren van de test lette ik zelf voornamelijk op het eenvoud en de snelheid waarmee de testpersoon de opdracht kon uitvoeren. Onderwijl maakte ik aantekeningen over het gebruik.

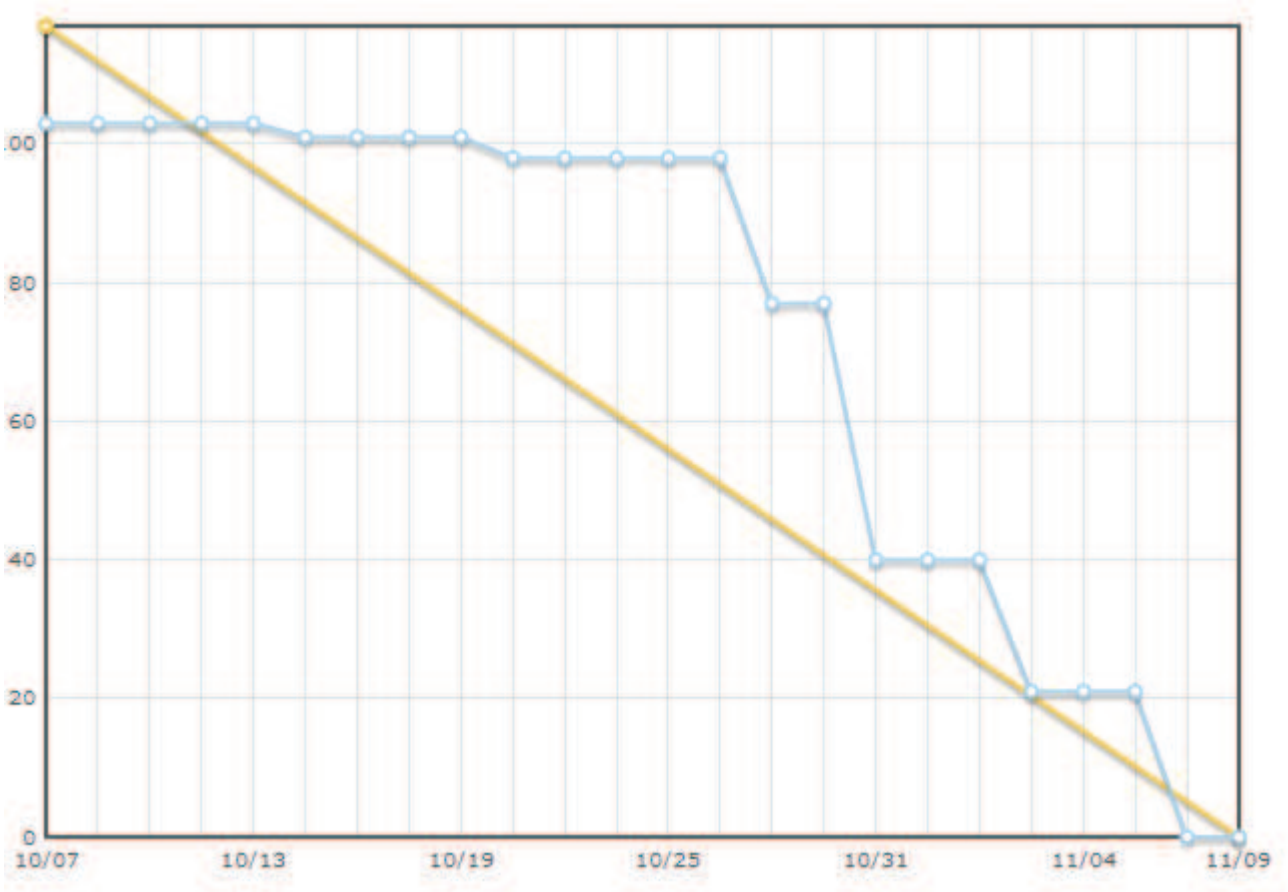
De sessie was erg leerzaam. Je ziet goed dat de testpersoon een heel andere benadering heeft van de opdracht en de applicatie dan dat je die zelf heeft. Het resultaat van de sessie was als volgt:

```
Uitvoer opdracht:
- Grid beter toonbaarder maken, bedoeling niet 1 keer duidelijk
- Bedoeling label duidelijker maken
- Velden css wel zichtbaar maken?
- Veld grootte bedoeling is niet duidelijk, men verwacht dat het maximale
karakters is in het veld
- Labels mist verplicht indicatie (*)
- Afhankelijkheid veld mag leeg zijn, nu altijd verplicht
- Veplichte velden van formulier eigenschappen duidelijk zichtbaar maken

Opmerkingen:
- Validatie waarschuwingen zijn nu maar in 1 taal
- Selectieveld afhankelijkheden alleen elementen uit eigen formulier tonen
(filter toevoegen)
- restrictie op karakters in naam veld
- Bij text input veldgrootte niet verplicht maken
- Multivalue set aanpassen, scherm breekt nu
- Validatie op Textarea element breekt
- Hoe kan je een rij of kolom toevoegen?
```

9.6 Wat ging er goed

Over het verloop van deze sprint ben ik tevreden. Voor mijzelf denk ik dat ik de mij gestelde doelen omtrent het project management goed heb opgepakt en daar in deze sprint beter mee om ben gegaan. Dit werd ook bevestigd door mijn begeleider en de product owner in de Sprint Review Meeting.



Het ontwikkelen van de user stories liep voorspoedig ondanks enige tegenslagen in het implementeren van de stories binnen Toolbox. Wederom liep ik regelmatig tegen het feit aan dat we met deze module tegen de randen van Toolbox zaten. Het aanpassen van Toolbox om het wel mogelijk te maken is geen probleem aangezien het eigen ontwikkelde software is, maar het is in sommige gevallen wel een tijdrovende klus.

De hallway testen zijn bijzonder leerzaam geweest. Het is ontzettend leuk om te zien hoe een persoon zonder enige ervaring met jouw applicatie omgaan. Het is niet alleen een reflectie op de applicatie, maar ook op je eigen gedachtegoed van hoe een applicatie moet werken.

9.7 Wat ging er niet goed

In deze sprint heb ik al aardig wat aandacht besteed aan het testen, maar naar mijn gevoel zou dit nog een stuk actiever en beter kunnen. Door meer zinnige test cases te schrijven die ook echt logica testen in plaats van het verhogen van de code coverage.

Daarnaast moet ik er zorg voordragen dat een user story bij het opleveren in één keer voldoet aan de definition of done in plaats van het in meerdere stappen bereiken. Door mijn enthousiasme richt ik me teveel op het ontwikkelen van nieuwe functionaliteiten in plaats van het compleet afronden van een functionaliteit.

10. Sprint 3

In sprint 3 is het systeem verder ontwikkeld en uitgebreid met nieuwe functionaliteiten. Deze sprint liep van 14 november 2011 tot en met 23 december 2011.

10.1 Sprint planning meeting

Ook voor deze laatste sprint van dit afstudeerproject is er wederom gestart met een sprint planning meeting. Voor deze sprint waren er 130 punten aan story points te verdelen. In overleg met de Product Owner en mijn begeleider is er gekozen voor de volgende user stories uit de product backlog.

Dynamisch kunnen wijzigen van het aantal rijen en kolommen
Een beheerder moet een formulier kunnen kopiëren
Een beheerder moet aan kunnen geven welke gebruikers het formulier moeten invullen
Een beheerder moet kunnen zien welke gebruikers het formulier nog niet ingevuld hebben
Een beheerder moet gebruikers op de hoogte kunnen stellen dat ze het formulier moeten invullen
Een beheerder moet kunnen opgeven naar welke bedank pagina er gestuurd moet worden na het invullen
Een beheerder moet de statistieken van een formulier kunnen inzien
Een beheerder moet kunnen zien op welke pagina's een formulier gebruikt is
Een beheerder moet de ingevulde formulieren kunnen exporteren naar Excel
Ondersteunen element: File
Opslag van bestanden die in formulier geüpload worden
Een beheerder moet een preview van het formulier kunnen bekijken

In deze sprint heb ik ook veel tijd besteed aan het opstellen van de diverse rapportages. Naast dit afstudeerverslag is dat ook het Scrum rapport betreffende het advies omtrent de methode Scrum.

10.2 Refactoring; leren van de testresultaten

Gedurende het project leer je iedere keer van elke test die uitgevoerd wordt. Eén van de eerste dingen die ons op viel in de hallway tests betrof het werken met het grid. We kwamen er achter dat het werken met het grid zoals wij bedacht hadden niet functioneerde tijdens het werkelijke gebruik van de module.

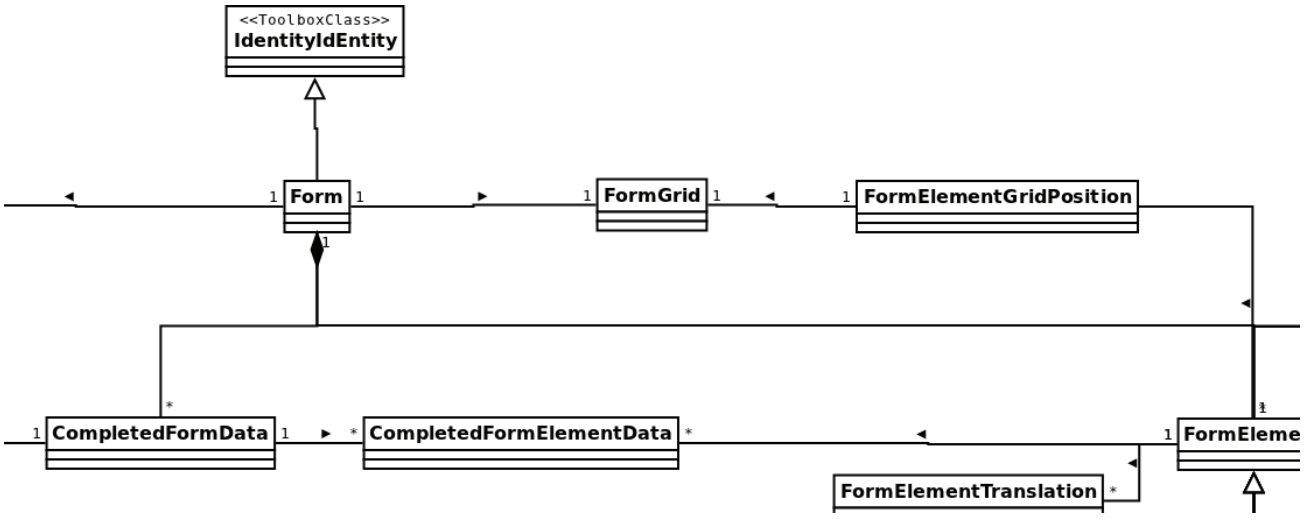
In de oorspronkelijke situatie moest de beheerder bij het aanmaken van het formulier bepalen hoe groot het grid moest zijn. Hij of zij moet bijvoorbeeld vooraf vastleggen dat er gebruik gemaakt zou worden van een 4 x 2 grid, bestaande uit vier rijen en twee kolommen.

Tijdens de testen bleek dat het regelmatig voor kan komen dat er toch nog even een rij of kolom bij moet en daar was in de oorspronkelijk situatie geen rekening mee gehouden. Een beheerder zou dan een nieuw formulier moeten maken. Niet bepaald gebruiksvriendelijk te noemen.

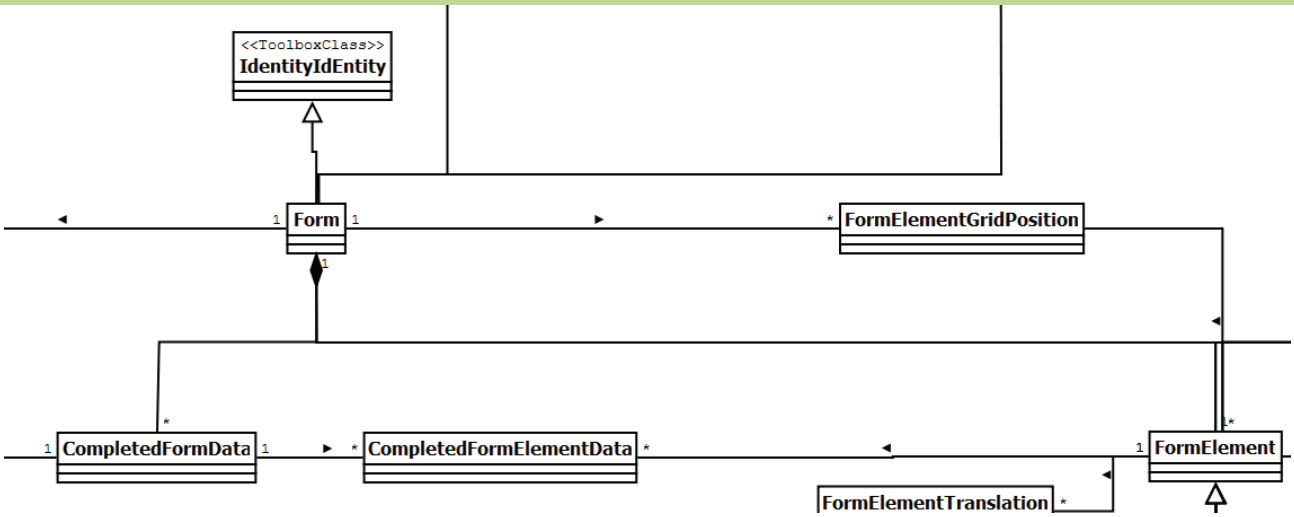
Een van de belangrijkste user stories in deze sprint was dan ook het dynamisch kunnen toevoegen van een rij of een kolom aan het formulier.

Om dit mogelijk te maken heb ik eerst kritisch naar het huidige model gekeken om te zien hoe deze wijziging het beste geïmplementeerd kan worden.

Situatie voor refactoring



Situatie na refactoring



Zoals de bovenstaande domeinmodellen laten zien zijn de wijzigingen minimaal van aard. Dit is een inherent gevolg van het toepassen van SOLID design principes. De verantwoordelijkheden zijn op een juiste manier gescheiden waardoor wijzigingen minimale impact hebben op overige onderdelen in het systeem.

De feitelijke wijziging is het verwijderen van de entiteit FormGrid en het verleggen van de relatie tussen Form en FormGrid naar Form en FormElementGridPosition.

Alhoewel de wijzing op het niveau van de entiteiten en business logica relatief weinig impact heeft, de wijzigingen voor de eigen geschreven javascript plugin zijn stukken groter.

De plugin moest gaan beschikken over extra functionaliteiten om de rijen en kolommen toe te kunnen voegen. Dat betekent niet alleen dat we extra cellen moeten tekenen, maar ook de representatie van iedere cell opnieuw moeten tekenen.

Het grid verandert namelijk niet van formaat, alleen de cellen worden kleiner bij het toevoegen van een kolom. Daarmee automatisch de inhoud van iedere cell ook. Dit was dan ook een goed moment om de plugin wat slimmer op te zetten zodat de plugin zelf ook in andere contexten te gebruiken is en niet meer hard gekoppeld aan de elementen binnen de formulierenmodule.

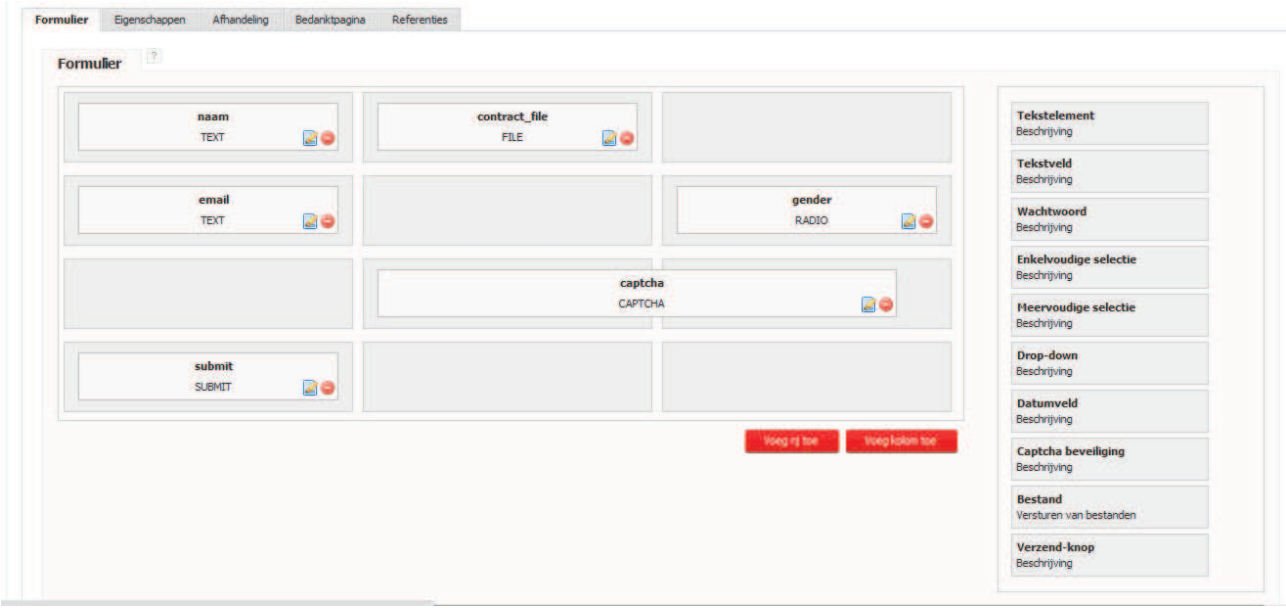
10.3 Evolutie van de user interface

Zoals in het vorige hoofdstuk al aangegeven leer je iedere keer weer van de testresultaten en je eigen gebruik van het systeem. Met iedere functionaliteit die je toevoegt pas je vaak ook iets aan de user interface aan waardoor je iedere keer moet overwegen of deze dan nog wel zo gebruiksvriendelijk en intuïtief is.

De hallway testen die uitgevoerd worden met personen zonder enige voorkennis zijn dan ook bijzonder leerzaam. Als ontwikkelaar kies je toch vaak dezelfde paden van element A naar element B. Tijdens de diverse hallway testen kwamen een aantal pijnpunten naar voren in de user interface. Deze pijnpunten zijn:

- Het dubbelklikken om de elementeigenschappen te openen is onduidelijk
- Het grid is te smal en word iel bij een groot aantal kolommen
- De extra dialogen om nieuwe waardes in dataset toe te voegen zorgen voor teveel klikken van de gebruiker

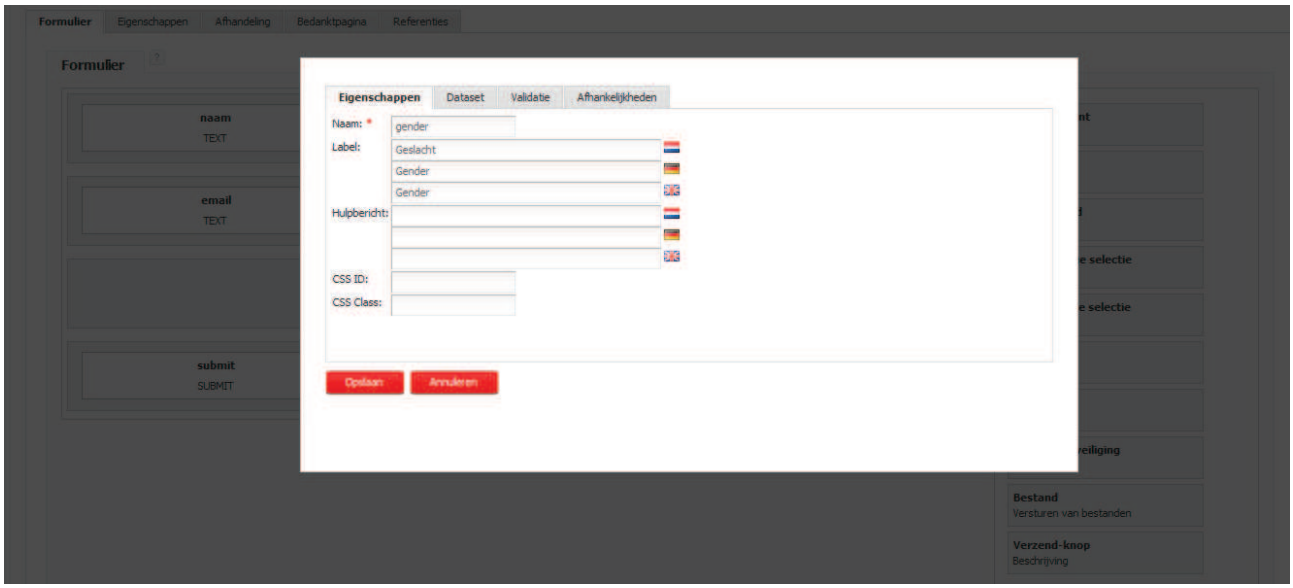
In deze sprint hebben we dan ook kritisch gekeken hoe we deze problemen konden oplossen zodat de user interface intuïtiever voor de beheerder te gebruiken is.



In het uiteindelijke resultaat hebben we het eerste punt opgelost door de elementen te voorzien van iconen voor het bewerken en verwijderen van elementen. Hiermee konden we het dubbelklikken verwijderen van de elementen. Voor de beheerder is het nu gelijk inzichtelijk waar de bewerk en verwijder functionaliteiten zitten.

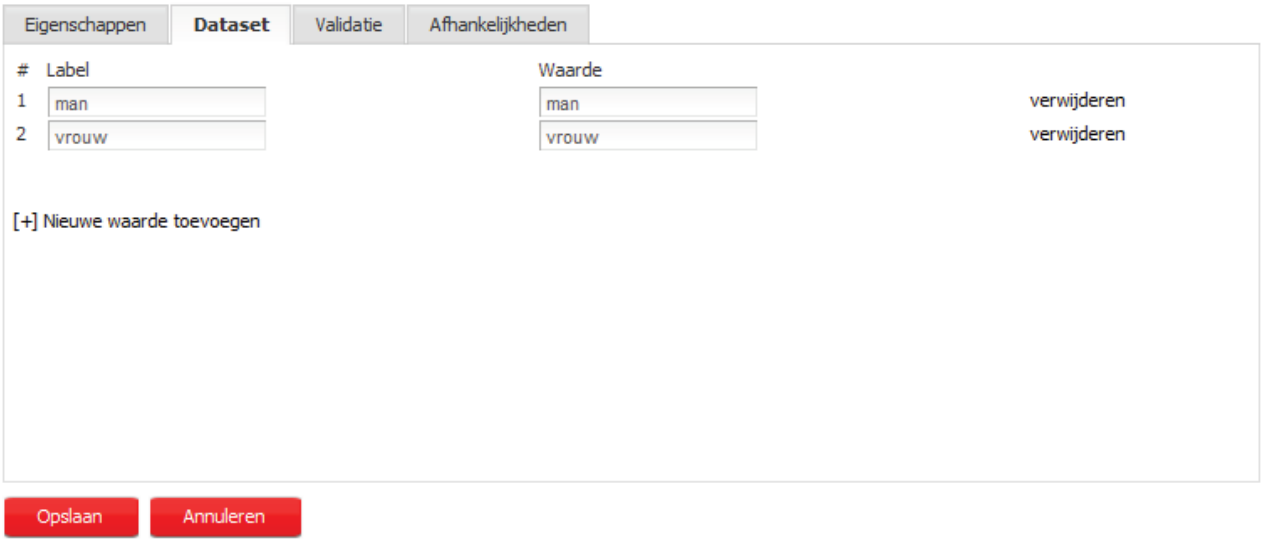
Het probleem van het te smalle grid hebben we opgelost door het bewerken van de element eigenschappen weg te stoppen in een dialoog die boven de pagina komt te zweven. Omdat we wel wilden dat een beheerder eenvoudig nieuwe elementen op het grid kan plaatsen hebben we aan de rechterkant ruimte gehouden voor een lijst van mogelijke elementen. Door middel van drag and drop kan een beheerder deze elementen op het grid plaatsen.

Wanneer de beheerder een element wilt bewerken krijgt hij of zij een dialoog die dezelfde tabbladen bevat zoals ze in de vorige sprints in de rechterkant stonden. Het voordeel is nu wel dat deze tabbladen meer ruimte krijgen en daardoor rustiger overkomen op de gebruiker. Daarnaast focust het de beheerder op de taak die hij op dat moment wilt uitvoeren zoals te zien in onderstaande afbeelding.

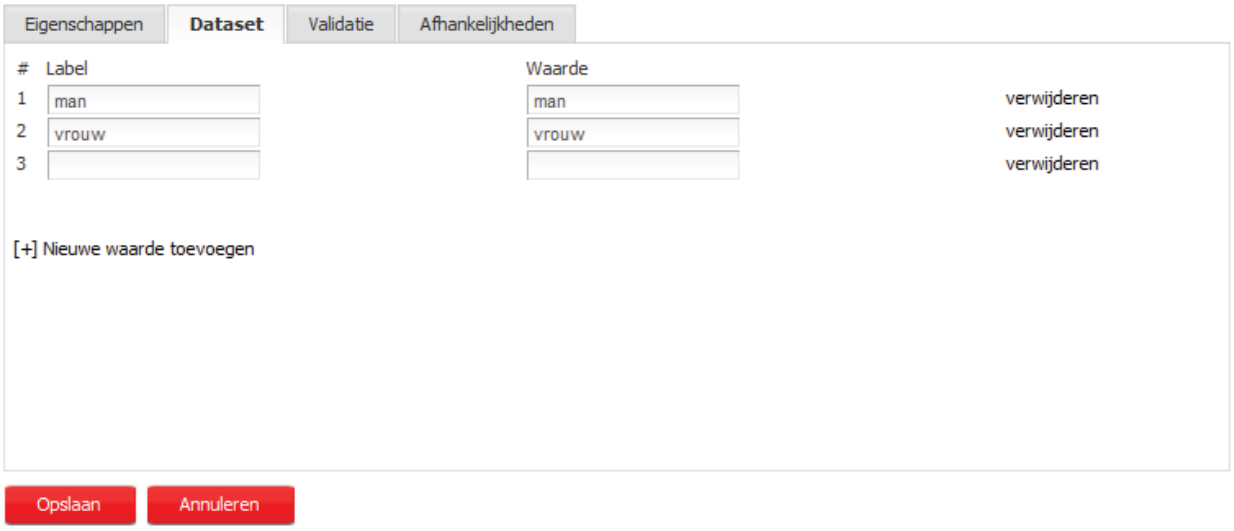


Het laatste probleem hebben we opgelost door de extra dialogen weg te halen en op dynamische wijze een nieuwe rij toe te voegen aan het formulier.

Bij start van bewerken



Na toevoegen van nieuwe waarde



Het voordeel van deze werkwijze is dat de invoer voor de beheerder veel sneller kan verlopen en dat het weghalen van de dialogen behoorlijk wat ergernis bij een gebruiker zal weghalen.

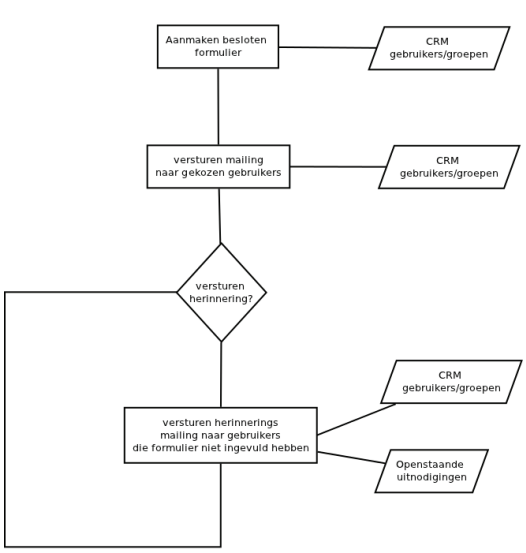
10.4 Koppeling met de CRM module

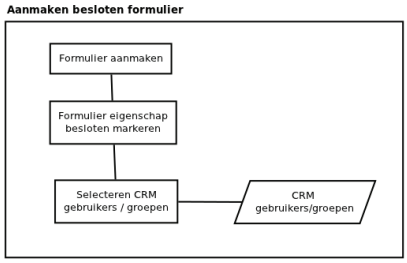
Een wens van de Product Owner is het kunnen koppelen van de formulierenmodule met de CRM module die binnen Toolbox actief is. Door deze koppeling kan een beheerder aangeven dat een formulier door een bepaalde groep van mensen ingevuld moet worden. De koppeling moet vervolgens ook registreren wie het formulier wel of niet heeft ingevuld.

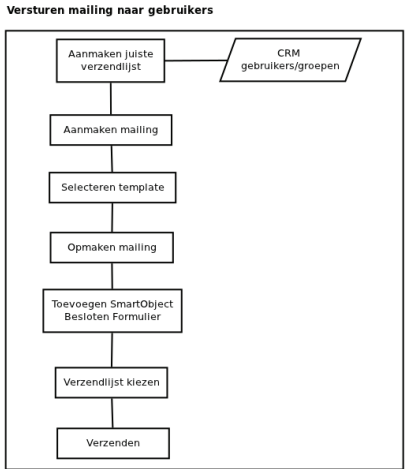
De beheerder moet de gebruikers via de formulierenmodule kunnen uitnodigen om het formulier in te voeren. Niet alleen voor de eerste uitnodiging, maar ook om een herinnering te kunnen versturen als iemand hem nog niet ingevoerd heeft.

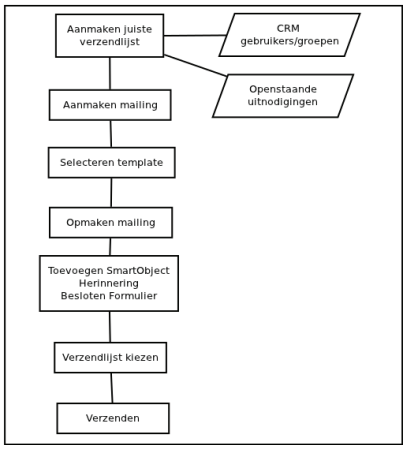
Deze wensen zijn goed uit te voeren binnen de formulierenmodule, maar vergen wel wat implementatie werkzaamheden en wijzigingen in de CRM module. Eerste taak was dan ook om eerst de benodigde wijzigingen te inventariseren voordat we verder konden met de implementatie.

Met de Product Owner heb ik eerst bepaald wat de workflow van de mogelijk processen in de gegeven wensen is. Door de workflow te identificeren en welke informatie in het proces nodig werd het duidelijk waar ik moet gaan kijken voor implementatiemogelijkheden.









Een wens die naar boven kwam tijdens het maken van deze inventarisatie is dat het versturen van de notificaties en herinneringen via de mailing module van Toolbox moet gebeuren. Voordeel van deze wens is dat de mails die verstuurd worden naar de CRM gebruikers door de beheerder opgemaakt kunnen worden en eenvoudig voorzien van extra informatie.

Technisch gezien is het voordeel dat we binnen de formulierenmodule nu geen voorzieningen hoeven te treffen om e-mails op te maken en te versturen. We kunnen vrij eenvoudig een aansluiting op de mailing module maken om het versturen te regelen.

Na de functionele inventarisatie heb ik een technische inventarisatie gemaakt welke wijzigingen we uit moeten voeren om de wensen mogelijk te maken.

Formulierenmodule
- Bepalen of mailing module aanwezig is
- Aanpassen edit scherm
- Uitlezen CRM gebruikers
- Bij opslaan koppeling tussen CRM gebruiker en formulier leggen
- Genereren identificatiecode
- Maken Smart Object 'Besloten Formulier'
- Controle inbouwen of formulier wel op een pagina ingevoegd is (invoegen smart object)
- Maken Smart object 'Herinnering besloten formulier'
- Controle inbouwen of formulier wel op een pagina ingevoegd is (invoegen smart object)
- Ophalen gebruikers die formulier nog niet ingevuld hebben
- Bij tonen van besloten formulier controleren of identificatie in URL aanwezig is
- Nee; dan formulier niet renderen
- Ja: formulier renderen
- Bij opslaan formulier flag zetten dat gebruiker formulier heeft ingevuld
- Maken overzicht genodigden en wie formulier wel of niet heeft ingevuld
- Wellicht koppeling maken naar de gestuurde mailings?

CRM Module
- Introduceren van groepen
- Aanpassen van hevellijst zodat gefilterd kan worden op naam/kenmerk
- Het e-mail adres vast toevoegen als attribuut van een relatie
- Mogelijk maken uitlezen van CRM gebruikers / Groepen

Al met al hebben de wensen niet alleen impact op de nieuwe formulierenmodule, maar ook op de CRM module. Om deze reden is dan ook besloten om de ontwikkeling van deze user stories stop te zetten. De Product Owner wil even de tijd nemen om de impact op de CRM module te overwegen en een zakelijke beslissing nemen om de ontwikkeling wel of niet voort te zetten.

10.5 Advies Scrum bij Eleven

Een ander onderdeel van de afstudeeropdracht is het schrijven van een advies aan Eleven over hoe zij beter en meer gebruik kunnen maken van de software ontwikkelmethode Scrum.

Tijdens de uitvoering van de afstudeeropdracht heb ik getracht zoveel mogelijk onderdelen van Scrum te gebruiken die zinnig waren de uitvoer van het project. Een flink aantal van de onderdelen waar ik gebruik van heb gemaakt werden binnen Eleven nog niet toegepast. Dit project was dan ook gelijk een mooie testcase om deze onderdelen toe te passen.

Nieuwe onderdelen die ik toegepast heb van de methode Scrum waren:

- (formele) Product Owner
- Sprint planning meeting
- Sprint retrospective meeting
- Burndown Chart
- Planning poker
- Puntenschaal
- Definition of Done

Om deze onderdelen goed toe te passen heb ik een flinke studie uitgevoerd naar de achtergronden en bedoelingen van deze onderdelen. In de literatuurlijst van het bijgesloten “Scrum bij Eleven” rapport staan alle geraadpleegde bronnen.

Aangezien Scrum een empirische methode is heb ik ook veel kennis genomen van de meningen en ervaringen van mensen die sinds kort of al jaren gebruik maken van de methode Scrum. Wat daarbij opvalt is dat iedereen zijn eigen draai geeft aan de diverse onderdelen en dat is dan misschien ook wel belangrijkste conclusie die getrokken moet worden.

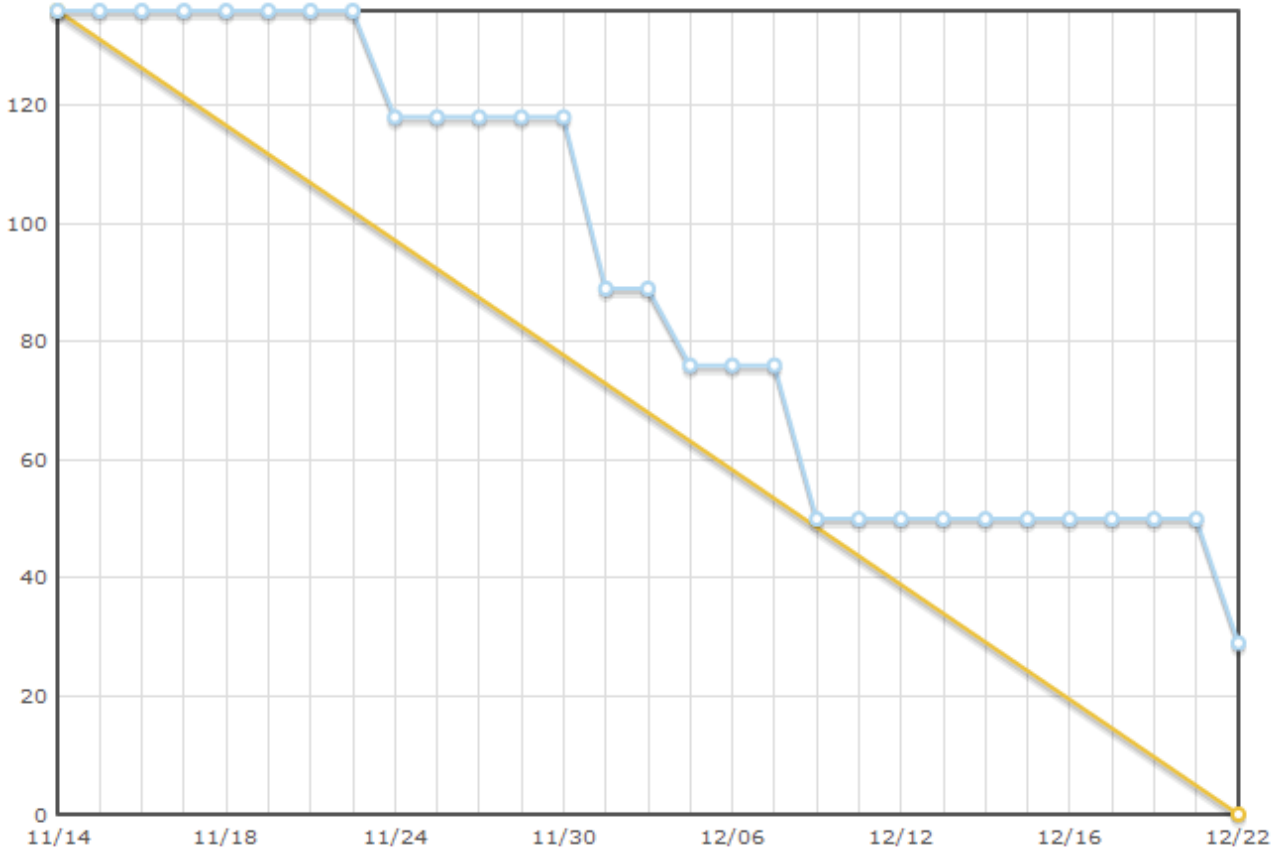
Scrum geeft puur een richtlijn hoe de onderdelen uitgevoerd moet worden, maar geeft de ruimte aan het team om te spelen met de wijze van gebruik. Op deze manier zal de methode ook eerder voor honderd procent gedragen worden door de organisatie.

In het adviesrapport heb ik naast het beschrijven van de diverse onderdelen die Scrum rijk is, ook gelijk een advies gegeven over de onderdelen die mij voor Eleven van toepassing zijn. De onderdelen die ik uitgekozen heb kunnen de organisatie op korte termijn helpen om projecten gestroomlijnder en volgens de Scrum methode uit te voeren. Om de implementatie van de methode te versnellen heb ik het rapport afgesloten met een advies over hoe de methodes en in welke volgorde zijn geïmplementeerd kunnen worden.

10.6 Wat ging er goed

Het was fijn om te zien dat de refactoring zoals uitgevoerd in dit project betrekkelijk eenvoudig verliep door het goede ontwerp wat we gemaakt hebben aan het begin van het project. De vele uren die we daarin gestoken hebben bewezen zich tijdens de refactoring.

Het schrijven van het adviesrapport was een leuke uitdaging. Niet zozeer omdat advies geven moeilijk is, maar omdat ik heb geprobeerd het vanuit directieperspectief te beschrijven en aan te tonen hoe het projecten soepeler kan laten lopen.



Het verloop van de ontwikkeling ging naar wens, ook al laat de burn down chart zien dat er in de eerste weken weinig afgerond is. Dat komt voornamelijk omdat ik in het begin grote user stories heb opgepakt, die door kleine complicaties niet direct afgerond konden worden.

Door het on hold zetten van de CRM user stories bleef er op het eind nog punten over die niet ontwikkeld zijn.

10.7 Wat ging er niet goed

Wat lastig was in deze sprint is dat ik ook vrij veel tijd moest besteden aan het opstellen en afronden van de extra documentatie die behoren tot het afstuderen. In deze fase ben je wat tijd kwijt aan alle administratie rondom het afstuderen en het schrijven van dit afstudeerverslag. Hier heb ik bij het plannen van deze sprint te weinig rekening mee gehouden waardoor ik veel extra tijd heb moeten vrijmaken voor het afstuderen. Niet dat ik daar moeite mee heb, maar het vraagt nogal wat van je lichaam en sociaal leven.

11. Evaluatie

11.1 Productevaluatie

Het opgeleverde product voldoet aan de gestelde eisen zoals opgesteld in het afstudeerplan. Het is voor een gebruiker mogelijk om een formulier te maken waarin men de keuze heeft uit de diverse beschikbare HTML elementen. Op ieder element is het mogelijk om validatie- en afhankelijkhedenregels in te stellen. Hierdoor heeft een beheerder behoorlijke controle op het formulier en de informatie die hij of zij uiteindelijk ontvangt.

Het opgeleverde product is een oplossing voor de gestelde probleemstelling. Een beheerder van de website is met deze module niet meer afhankelijk van Eleven om formulieren toe te voegen of wijzigingen aan een formulieren aan te brengen.

De mogelijkheden van de nieuwe module omtrent de afhandeling van ingevulde formulieren geeft een beheerder meer mogelijkheden dan met een huidige klant specifieke implementatie mogelijk is.

Voor Eleven verhoogt deze nieuwe module de business value van Toolbox WMS. Het biedt een extra verkoopargument in verkoopgesprekken en zal de waardering voor Toolbox bij huidige klanten verhogen.

Het Scrum rapport geeft Eleven een goed inzicht in de methode Scrum en hoe ze dit kunnen implementeren in de organisatie. Het laat zien welke onderdelen interessant zijn om het verloop van de projecten soepeler te maken.

11.2 Procesevaluatie

11.2.1 Wat ging er goed

Het uitvoeren van de opdracht is voorspoedig verlopen en ik kan dan ook terugkijken op een goed verlopen project. In de afgelopen maanden heb ik kunnen laten zien dat ik een project van start tot eind kan uitvoeren en managen. Natuurlijk heb ik nog een heleboel te leren, maar dankzij dit project en de begeleiding vanuit Eleven ben ik al stukken wijzer geworden en completer als Software Developer.

Tijdens mijn afstuderen heb ik mij kunnen verdiepen in methodieken en technieken die al een tijd op mijn wensenlijst stonden. Onderdelen van die wensenlijst waren het testen van de GUI door middel van Selenium, de principes van SOLID design en een verdere verdieping van de projectmethode Scrum.

Het werken met Scrum is goed gegaan ondanks je eigenlijk alleen zelf ontwikkelt aan de applicatie. Je merkt dat de contactmomenten die in Scrum ingebakken zijn goed zijn om de Product Owner en begeleider op de hoogte te houden van de voortgang. Ondanks dat dit in sprint 1 nog wat moeizaam ging, ben ik van mening dat ik dit in de overige sprints goed heb opgepakt.

Met het gebruiken van de, voor Eleven onbekende, Scrum onderdelen in dit project heb ik de organisatie laten zien wat voor nut deze onderdelen kunnen hebben voor de organisatie tijdens een project. De extra contactmomenten, maar ook de momenten waar de organisatie op zichzelf moet reflecteren om er van te leren kunnen zeer waardevol voor de organisatie zijn.

11.2.2 Wat ging er niet goed

Tijdens de start van het project heb ik mij onvoldoende gerealiseerd dat ik nog best wel wat werk zou hebben aan het geschikt maken van Toolbox voor het ontworpen domeinmodel. Hierdoor ontpopte zich nog wel eens een kleine frustratie na een lange zoektocht binnen Toolbox om de juiste wijzing door te voeren. Vooral in de eerste sprint uitte zich dat door middel van het doorschuiven van enkele user stories naar de volgende sprint.

Het leven van een deeltijdstudent is pittig te noemen. Naast een fulltime baan is het ook nog eens een flink aantal uren besteden aan studie en studie gerelateerde projecten. Tijdens het afstuderen zijn deze uren alleen nog maar eens groter geworden. Het is dan ook zaak om goed je dagelijkse leven goed te plannen. Hier had ik slimmer mee om kunnen gaan. Tijdens het afstuderen heb ik teveel werkzaamheden aangenomen voor maatschappelijke organisaties waardoor ik die tijd niet in het afstuderen kon besteden.

De goedkeuring van de opdracht heeft veel tijd gekost en zorgde voor enige onzekerheid tijdens het afstuderen. Hierbij hielp het ook niet dat één van de begeleiders tijdens deze fase werd vervangen door welke reden dan ook. De communicatie hierin was ook niet bijzonder sterk te noemen waardoor er onduidelijkheid en onzekerheid optrad.

Zelf had ik dit ook beter kunnen voorkomen door meer tijd te steken in het beschrijven van de opdracht en het uitwerken van het afstudeerplan. Veel afspraken of functionaliteiten zijn wel mondeling besproken maar niet of onvoldoende op papier gezet.

12. Verantwoording competenties

12.1 Opstellen gegevensmodel voor database [niveau 4]

Voor de applicatie is een gegevensmodel opgesteld om grafisch weer te geven wat er in de database opgeslagen zou worden. Bij het opstellen van het afstudeerplan is dit onderdeel op niveau 4 ingeschat omdat er door het gebruik van versiebeheer een behoorlijke complexiteit om de hoek zou komen kijken.

Het gebruik van Envers heeft dit uit handen genomen waardoor ons eigen datamodel een stuk eenvoudiger is geworden. Dit datamodel is beschreven in de gemaakte ERD diagrammen.

12.2 Ontwerpen systeemdeel [niveau 3]

Om de applicatie te kunnen ontwikkelen heb ik voor de ontwikkeling ook goed nagedacht over het ontwerp van de applicatie. Hiertoe heb ik diverse ontwerpen gemaakt met behulp van UML. Ik heb een domeinmodel opgesteld door middel van een klassendiagram. Ook zijn hierbij de gekozen kwaliteitsattributen in acht genomen tijdens het ontwerp.

Tijdens het ontwerp en de ontwikkeling van de applicatie bleek dat het niet nodig was om use case diagrammen, sequence diagrammen en state diagrammen te ontwikkelen. Scrum geeft aan dat deze UML diagrammen ook zo min mogelijk gebruikt dienen te worden. Doordat de methode snel op veranderingen inspeelt is het onderhouden van deze diagrammen een tijdrovende bezigheid of blijven de diagrammen niet valide met de werkelijkheid.

12.3 Bouwen applicatie [niveau 4]

Om de probleemstelling op te lossen heb ik een applicatie gebouwd op basis van Toolbox met behulp van Java. De code die ik heb geschreven is getest door middel van whitebox en blackbox testen.

12.4 Initiëren en plannen van het testproces [niveau 3]

Deze competentie heb ik bewezen door het schrijven van het Master Test Plan. Ander bewijs is de keuze en verantwoording van het gebruik van de kwaliteitsattributen. Daarnaast heb ik ook de omgeving voor de functionele Selenium tests opgezet en werkend gemaakt.

Voor de Hallway tests heb ook test casussen geschreven die als bewijs voor deze competentie kunnen dienen. Ook het schrijven van de DSL behoort tot deze competentie.

12.5 Uitvoeren van en rapporteren over het testproces [niveau 3]

Voor de applicatie zijn er diverse functionele tests en unit tests geschreven om de kwaliteit van de code te beoordelen en te verantwoorden. Rapportages van deze testen zijn aanwezig door middel van de rapportages gegenereerd door de continious integration server.

De uitgevoerde Hallway tests zijn door mij uitgevoerd en door middel van notulen gerapporteerd.

13. Referenties

13.1 Woordenlijst

Term	Definitie
Element	Een element is een onderdeel van een formulier en kan een veld zijn wat door de bezoeker ingevuld moet worden of een actie bevat. Het kan dus een HTML tekst input zijn of een submit button, maar een ook seperator.
Label	Is een tekstblok wat de betekenis van het element omschrijft.
Grid	Een raster waarop de elementen uitgelijnd worden.
Beheerder	De beheerder is de persoon die de gehele of onderdelen van de website beheert. De beheerder heeft toegang tot Toolbox WMS.
Bezoeker	De bezoeker is de persoon die de website bezoekt en indien nodig een actie uitvoert op een onderdeel van de website
Toolbox	Het Web Management Systeem van Eleven
Sprint	Een tijdsperiode van vaste duur waarin het ontwikkelteam werkt aan de applicatie.
Product Backlog	Een register van gewenste functionaliteiten geprioriteerd bij de bedrijfswaarde van ieder item.
Sprint Backlog	Een register van items uit het Product Backlog die in de sprint ontwikkeld gaan worden.
Agile Testing	Het beoefenen van software testen volgens de principes van de agile software ontwikkelmethode.
Annotaties	Speciale metadata die een specifieke functionaliteit van een class beschrijft, bijvoorbeeld @Entity
Domain-specific Language (DSL)	Een speciale versie van een programmeertaal gespecialiseerd voor een specifiek probleemdomein, bijvoorbeeld SQL of HTML.

13.2 Bibliografie

Agile-Development-Site. (sd). *Agile Development Site*. Opgehaald van https://sites.google.com/site/agiledevelopmentsite/process/sprint-retrospective

Eleven. (sd). *Eleven*. Opgehaald van www.eleven.nl: http://www.eleven.nl/site/nl/contact/organisatie

manifesto, A. (sd). *Agile manifesto*. Opgehaald van http://agilemanifesto.org/

Wiki-DSL. (sd). Opgehaald van Wikipedia: http://en.wikipedia.org/wiki/Domain-specific_language

WIKI-Hallway. (sd). Opgehaald van http://en.wikipedia.org/wiki/Usability_testing#Hallway_testing

Wiki-ISO_9126. (sd). Opgehaald van http://en.wikipedia.org/wiki/ISO/IEC_9126

Wikipedia-Scrum. (sd). *Wikipedia*. Opgehaald van http://en.wikipedia.org/wiki/Scrum_(development)

Wiki-SOLID. (sd). Opgehaald van Wikepedia: http://en.wikipedia.org/wiki/SOLID_(object-oriented_design)

13.3 Bijlagen

1. Afstudeerplan
2. Plan van aanpak
3. Testplan
4. Mockup applicatie sprint 0
5. Datamodel entiteiten na sprint 1
6. Datamodel relaties na sprint 1
7. Domeinmodel applicatie na sprint 3
8. Rapport: Scrum bij Eleven

Afstudeerplan

Ontwikkelen van een dynamische formulieren module binnen Toolbox WMS bij Eleven

Titel	Afstudeerplan afstuderen Paul de Raaij
Student	Paul de Raaij (09081852)
Opdrachtgever	Eleven te Maasdijk
Datum	maandag 2 januari 2012 te Honselersdijk

Inhoudsopgave

1. Het Bedrijf 3
2. Probleemstelling 3
3. Doelstelling van de afstudeeropdracht 4
4. Resultaat 4
5. Uit te voeren werkzaamheden, inclusief een globale fasering, mijlpalen en bijbehorende activiteiten 5
6. Op te leveren (tussen)producten 7
7. Te demonstreren competenties en wijze waarop 8

1. Het Bedrijf

Eleven is een jong bedrijf dat anderhalf jaar geleden is ontstaan uit een fusie tussen twee web-development bedrijven. Ze zijn voornamelijk actief voor het midden- en kleinbedrijf in de tuinbouw en zakelijk dienstverlening sector.

Op dit moment zijn er 14 medewerkers actief bij Eleven. Hiervan zijn 9 medewerkers werkzaam als ontwikkelaar, ofwel als Backend of Front-end developer. Ze beschikken over zowel een Java-ontwikkelstraat, als een PHP-ontwikkelstraat.

Eleven richt zich puur en alleen op de technische ontwikkeling van websites en web applicaties. Voor het ontwerp en communicatie advies werkt men met een groot aantal partners.

De afstudeeropdracht zal plaatsvinden binnen de Java tak van Eleven. Bij deze afdeling zijn 6 ontwikkelaars actief die Toolbox beheren en verder ontwikkelen. Ook voert deze afdeling de implementatie van nieuwe websites uit.

2. Probleemstelling

De websites die Eleven oplevert in de programmeertaal Java zijn gemaakt op de eigen ontwikkelde Toolbox. Toolbox is een webmanagement systeem waar een website in zijn totaliteit beheerd kan worden door een websitebeheerder.

Toolbox bestaat uit diverse modules die een klant apart kan afnemen. Er is een basis module waarin de basis van het framework is opgenomen. De modules die nu gekozen kunnen worden zijn de volgende:

- CMS
- Catalogus
- Mailing
- CRM

Daarnaast kunnen er voor ieder project specifieke wijzigingen uitgevoerd worden aan Toolbox.

Toolbox is niet alleen een modulair systeem, maar ook een framework. Een geheel van softwarecomponenten en afspraken die verenigd zijn binnen Toolbox. Dit framework is de basis voor iedere module en specifieke toevoeging voor een klant.

Groot gemis is dat nu voor ieder formulier dat gevraagd wordt door de klant, het werk moet worden uitgevoerd door een ontwikkelaar van Eleven. Soms is het een heel nieuw formulier, maar vaker is het een extra veld erbij of een waarde toevoegen aan een drop-down formulier.

Dit kost tijd en zal ingepland moeten worden. Het werk wat door de ontwikkelaar gedaan moet worden is vrij triviaal en repeterend. Voor de klant is dit vervelend omdat de aanpassing vaak langer duurt dan nodig.

Een formulier bestaat uit verschillende typen velden, validatie en een aantal afhandeling acties. Vrijwel altijd wordt er niet meer dan dit gevraagd. Het is dus goed mogelijk om dit beheersbaar te maken voor de beheerder.

Een afhandelingsactie is een taak die uitgevoerd moet worden na het versturen van een formulier door de websitebezoeker. Bijvoorbeeld het sturen van een e-mail naar de beheerder of het opslaan van de ingevulde waarden in de database.

Ander probleem is dat binnen Eleven projecten via een niet gestandaardiseerde wijze verlopen. Gaandeweg de tijd zijn er eigen gewoonten en procedures ontwikkeld. Ook is er geleend vanuit andere methodes.

Dit is lastig voor nieuwe medewerkers en sommige klanten omdat zij niet bekend zijn met onze projectmethodiek. Het duurt dan ook even voordat een nieuwe medewerker of klant efficiënt meedraait in een project.

3. Doelstelling van de afstudeeropdracht

Om de probleemstelling op te lossen ga ik gedurende de afstudeeropdracht een nieuwe module voor Toolbox maken. Met deze nieuwe module kan een beheerder op dynamische wijze zelf formulieren maken, de inhoud van deze formulieren bepalen en aangeven hoe ieder formulier afgehandeld moet worden.

Het voordeel voor de klanten is dat ze zelf formulieren kunnen aanmaken of wijzigen, zonder dat ze Eleven hoeven in te schakelen. Dit zal het gebruikersgemak voor de beheerder verhogen en extra mogelijkheden bieden.

Voor Eleven biedt deze nieuwe formulierenmodule een extra verkoopargument. Daarnaast zal deze module veel triviaal werk uit handen nemen van de ontwikkelaar en druk van de planning wegnemen. Dit omdat dit soort werkzaamheden vaak op korte termijn uitgevoerd moet worden.

Gezien de mogelijkheden van Toolbox WMS zou deze module goed als basis kunnen dienen voor toekomstige modules of op maat gemaakte uitbreidingen. Denk hierbij aan een reserveringsmodule voor kantoorruimte, hotels of congressen.

Om het probleem omtrent de projectmethodiek aan te pakken wil ik met dit project laten zien dat de methodiek Scrum goed past binnen Eleven. Dat we de houvasten en methodieken waarop Scrum gebaseerd is goed in te passen zijn bij Eleven en dat de methodiek zonder al te veel moeite in te passen is in het bedrijf.

4. Resultaat

Het eindproduct is een nieuw module binnen Toolbox WMS waarin een beheerder zelf formulieren kan toevoegen. De structuur van de formulieren wordt door de beheerder zelf bepaald. Het zijn dus geen standaard formulieren die toegevoegd worden.

Bij ieder formulier kan een gebruiker velden toevoegen of verwijderen. Een veld kan in ieder geval een van de volgende HTML typen zijn:

- Input field
- Textarea
- Radio button
- Checkboxes
- Drop-down
- File (voor afbeeldingen of documenten)
- Submit/Reset button

De beheerder kan voor ieder veld een set aan waarden opgeven die voor dat element mogelijk zijn. Dit zal vooral gebruikt worden bij radio buttons, checkboxes en drop down. Op deze manier zijn man/vrouw keuzes mogelijk.

Naast deze standaard HTML input velden zullen er ook Toolbox specifieke elementen in komen die interactie met Toolbox zullen hebben. Dit kan bijvoorbeeld een element zijn om alle medewerkers te tonen in een drop-down.

Een gebruiker selecteert dan dit element en het systeem zal daar een drop-down van maken met als waardes alle gebruikers binnen Toolbox.

Het kan zijn dat een element dan extra configuratie instellingen heeft. Bijvoorbeeld in het voorbeeld van de gebruikers, om alleen de gebruikers met de rol 'Medewerker' te tonen.

Ieder veld kan ook afhankelijk zijn van andere velden. Zo kan bijvoorbeeld veld B alleen getoond worden als Veld A de waarde 'Gehuwd' heeft. Deze afhankelijk heden kan de gebruiker zelf bepalen. Ze bepalen dan wat het afhankelijke veld is en welke waarde die moet hebben om getoond, dan wel verborgen te worden.

De beheerder bepaalt ook welke validatie er op een veld word toegepast. Men kan kiezen uit minimaal de volgende opties:

- Veld verplicht
- Geldig e-mailadres
- Geldig postcode
- Geldig telefoonnummer
- Geldig URL
- Alleen getallen
- Alleen getallen en letters

De beheerder kan daarnaast opgeven wat er met het formulier moet gebeuren als het ingevuld is. De beheerder kan uit de volgende opties kiezen (meerdere mogelijk):

- Ingevuld formulier mailen
- Ingevuld formulier in PDF rapport opslaan
- Ingevuld formulier opslaan in database

De ontwikkelaars van Eleven moeten op eenvoudige wijze kunnen integreren met de nieuwe formulierenmodule. Zo zou de module als basis kunnen fungeren voor toekomstige modules, bijvoorbeeld een reserveringsmodule.

De module zal sowieso integreren met de CMS module zodat de beheerder een formulier eenvoudig kunnen toevoegen aan een CMS pagina. Tevens kan een beheerder het bedankt bericht na een ingevuld formulier ook door laten sturen naar een CMS pagina.

Op de formulieren zal ook versiebeheer worden toegepast. Een gebruiker kan zo altijd eenvoudig terug springen naar een vorige staat. Dit heeft als gevolg dat de opslag van de ingevulde formulieren dus onafhankelijk moet zijn van de formulier definitie omdat anders de ingevulde waarden nergens meer op slaan.

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

Tijdens het afstuderen zal ik gebruik maken van de agile methode SCRUM. Op dit moment wordt binnen Eleven alleen de daily stand up meeting uit de SCRUM methode gebruikt. Tijdens dit afstudeerproject wil ik Eleven laten zien wat SCRUM nog meer te bieden heeft.

Gedurende het project ga ik gebruik maken van de onderdelen Product Backlog, Sprint Backlog en de sprint review meeting. Deze onderdelen worden nu nog niet gebruikt binnen Eleven en na dit project hoop ik Eleven een prettige kennismaking gegeven te hebben om de zinvolle elementen van deze onderdelen te

gebruiken in de toekomst.

Deze nieuwe onderdelen van Scrum die we nu nog niet gebruiken, vragen veel betrokkenheid van de betreffende collega's. Op deze manier laat ik mijn collega's goed kennis maken met de nieuwe onderdelen. Het is van belang dat mijn collega's hiervoor openstaan en ik zal ze ook vragen om tijd vrij te maken voor deze activiteiten.

Aan het eind van het afstudeertraject ligt er een rapportage die beschrijft hoe deze onderdelen binnen het afstuderen gebruikt zijn en met welke hulpmiddelen.

Het rapport over het gebruik van Scrum binnen dit project laat een implementatie van de nieuwe onderdelen zien en zal tenslotte ook een eenvoudig advies geven hoe deze methodes zijn in te zetten binnen Eleven en toekomstige projecten.

De eerste stap, ook wel Sprint 0 genoemd, zal zijn om de requirements op te stellen en te verzamelen in het Product Backlog. Iedere requirement zal opgesteld worden als een user story. De requirements worden beschreven vanuit de wens van een gebruiker. Voor deze fase reserveer ik twee dagen.

Het project zal in 3 sprints van 18 dagen worden opgesplitst. Bij de start van iedere sprint zal er een Sprint Planning Meeting zijn waarin samen met de Product Owner het werk wordt geselecteerd voor die sprint. Aan de start van een sprint zullen de gekozen user stories opgedeeld worden in taken en een geschatte tijdsduur krijgen. Dit gebeurt zonder inspraak en aanwezigheid van de Product Owner.

De Product Owner van Toolbox en de nieuw te bouwen module is de algemeen directeur van het bedrijf, Peter Boon.

Aan het eind van de sprint zal er een Sprint Review Meeting gehouden worden waarin het werk aan de Product Owner wordt getoond en wordt gereflecteerd of wat er wel en niet is gebeurd. Ook zal hier feedback gegeven worden op het gemaakte werk.

Binnen het afstudeerproject zal mijn functie uit meerdere rollen bestaan aangezien ik dit project alleen uitvoer. Naast de project manager zal ik ook de lead developer en architect zijn.

De reviews op code en architectuur zullen uitgevoerd worden door de bedrijfsmentor, Carlo Vollebregt. Mocht ik tegen problemen aanlopen kan ik ook bij hem terecht.

Daarnaast zullen mijn collega's uit interesse en enthousiasme ook regelmatig vragen wat de status van het project is en gevraagd of ongevraagd hun advies geven. Binnen Eleven is het gebruikelijk om aan het eind van de week een kleine presentatie te geven van een project waar je aan gewerkt hebt. Ook dit project zal regelmatig hierin voorbij komen.

Wat betreft de requirements en interface van de module zal ik voornamelijk contact hebben met de Product Owner, Peter Boon.

Het testen zal ook binnen de Agile omgeving plaatsvinden. Aangezien dit nieuw is voor mij zal ik deze kennis gedurende het afstudeerproject op moeten doen. Hiervoor heb ik diverse boeken over Agile Testing besteld.

Om een beeld te geven van het werk hieronder een kort overzicht met de user stories die in grote lijnen aanwezig zijn in het project.

User stories	Geschat benodigde werkdagen
NON-USER-STORY Opstellen project plan	5
NON-USER-STORY Opstellen test rapportages	10
De beheerder moet een formulier kunnen aanmaken, bewerken en verwijderen.	5
De beheerder moet velden kunnen toevoegen/bewerken en verwijderen aan een formulier.	10
De beheerder moet een validatie kunnen toevoegen/bewerken en verwijderen aan een veld.	8
De beheerder moet een afhandelingsactie kunnen toevoegen/bewerken/verwijderen	10
De beheerder moet een formulier kunnen integreren in een CMS pagina.	3
Een bezoeker van de website moet een of meerdere bestanden kunnen uploaden.	5
De beheerder moet een bedank pagina voor een formulier kunnen opgeven.	3
Eleven moet de module eenvoudig kunnen integreren en inschakelen binnen het project	2
Eleven moet programmatisch afhandelingsacties kunnen toevoegen	5
NON-USER-STORY Integratietest & Overdracht	3
totaal	69

Naast deze werkzaamheden voor het project zal er ook tijd besteed worden aan het opstellen van de producten omtrent het afstuderen.

De globale planning is als bijlage bijgevoegd.

6. Op te leveren (tussen)producten

Door de Agile methode zal het Product Backlog, ieder Sprint Backlog en de burn down chart opgeleverd worden als product.

De formulieren module zal het ultieme eindproduct voor Eleven zijn. Tijdens de ontwikkeling daarvan zullen enkele ontwerp documenten opgeleverd worden. Deze worden nader bepaald tijdens de sprint meetings, maar zal in ieder geval bevatten:

- Domeinmodel (UML)
- Sequence diagram (UML)
- Deployment diagram (UML)
- Schermschetsen GUI
- Entity Relationship Diagram

Naast deze producten voor het eindproduct, zijn er ook nog een aantal producten die opgeleverd worden voor het afstuderen.

Het eerste product wat voor het afstuderen opgeleverd zal worden is het projectplan. Dit projectplan beschrijft het project, de gekozen aanpak, risico's en beheersmaatregelen, projectorganisatie, planning en kwaliteitsbewaking.

Omdat ik tijdens het afstuderen aan Eleven wil laten zien wat de methode SCRUM nog meer te bieden heeft, zal er ook een rapport opgeleverd worden wat laat zien hoe ik de methode SCRUM tijdens het afstudeerproject heb gebruikt.

In het adviesrapport ' Scrum bij Eleven' beschrijf ik niet alleen hoe ik SCRUM heb ingezet, maar ook welke tools en methodieken daarvoor heb gebruikt.

Als einddocument zal het afstudeerverslag opgeleverd worden. In dit verslag staat de gang van zaken tijdens het afstuderen beschreven. Het laat zien welke werkzaamheden ik heb uitgevoerd, welke moeilijkheden en keuzes ik heb moeten maken en hoe er tot een bepaalde keuze is gekomen.

7. Te demonstreren competenties en wijze waarop

2.1 Opstellen gegevensmodel voor database [niveau 4]

Voor de module zal een gegevensmodel opgesteld moeten worden om de structuur van het formulier op te slaan. Daarnaast zullen ook alle resultaten van ingevulde formulieren opgeslagen moeten worden, indien voor deze afhandelingsactie gekozen is.

De formulieren zullen ook beschikken over versiebeheer en dit brengt een uitdaging met zich mee om zo compact mogelijk de gegevens te bewaren. De context en dynamiek van de formulieren mag daarbij niet in het gedrang komen.

Deze competentie word bewezen in het product Entity-Relationship Diagram

3.2 Ontwerpen systeemdeel [niveau 3]

Voor de module zal een technisch ontwerp gemaakt moeten worden met behulp van UML. Hierin zullen verschillende design patterns verwerkt worden. Het uitgangspunt een is object georiënteerd ontwerp zijn waarin de SOLID[1] principes gevolgd worden.

De producten die opgeleverd zullen worden voor deze competentie zijn Use case diagrammen, klassendiagrammen, sequence diagrammen, state diagrammen (als nodig).

Daarnaast zullen er ook scherm schetsen en ontwerpen gemaakt moeten worden voor de GUI. Zowel van de backend als de front-end.

Alhoewel de module binnen Toolbox gebouwd wordt zullen er enkele architectuur keuzes gemaakt moeten worden betreffende multitier architecture, event-driven architecture.

3.3 Bouwen applicatie [niveau 4]

De module zal in Java ontwikkeld worden binnen het Toolbox framework.

Binnen Toolbox word er nu al gebruik gemaakt van een aantal libraries waar ik binnen dit project niet omheen kan. Een voorbeeld daarvan is Hibernate die in combinatie met Ehcache gebruikt wordt als persistence library.

Dit biedt mij als ontwikkelaar enkele voordelen, maar zal er ook voor zorgen dat ik bepaalde keuzes moet afstemmen op de gewoonte van de library.

Deze competentie word aangetoond in de opgeleverde applicatiecode. Gebruikte libraries: Eclipse als IDE, Hibernate als persistence library. Ehcache als caching library, Selenium voor GUI testing, SVN als versioning control system en Maven als build management tool.

3.4 Initiëren en plannen van het testproces [niveau 3]

Om te bepalen hoe de module getest moet worden zal er een Mastertestplan opgesteld worden die de verschillende testen definieert. Daarnaast zal er voor iedere testmethode een detailtestplan opgesteld worden.

De testsoorten die sowieso verwacht worden zijn unit tests en functional tests. De functional tests moeten worden uitgevoerd in Selenium.

Aan de start van het project zal er in samenspraak met de Product Owner besloten worden op welke ISO 9126 kwaliteitsattributen er het meeste gericht moet worden om tot een acceptabel eindproduct te komen.

[1] http://en.wikipedia.org/wiki/Solid_(object-oriented_design)

De producten die bij deze competentie horen zijn het master test plan, detail test plan. Daarnaast zijn ook de module en functionele testen onderdeel van deze competentie.

3.5 Uitvoeren van en rapporteren over het testproces [niveau 3]

Aan de hand van de diverse detail test plannen zullen de verschillende tests gemaakt en uitgevoerd worden. De rapportage van de uitvoer zal gebruikt worden om de kwaliteit te bepalen van de code en de functionaliteiten.

De eindproducten van deze competentie zullen aangetoond worden door de testresultaten.

Projectplan

Ontwikkelen van een dynamische formulieren module binnen Toolbox WMS bij Eleven

Titel	Plan van Aanpak "Toolbox Formulierenmodule"
Student	Paul de Raaij (09081852)
Opdrachtgever	Eleven te Maasdijk
Datum	maandag 2 januari 2012 te Honselersdijk

Inhoudsopgave

1. Inleiding 3
2. Projectomschrijving 3
   Probleemstelling 3
   Doelstelling 3
   Afbakening 4
3. Aanpak 4
   Projectmethodiek 4
   Programmeertechnieken 5
4. Projectorganisatie 6
   Structuur 6
   Resources 7
   Tijdregistratie 7
   Projectnotities 8
5. Risicomanagement 8
   Risico: De scope van het project is onduidelijk 8
   Risico: Het gestelde tijdspad is onrealistisch 8
6. Kwaliteitsborging 9
   Testen 9
   Review meetings 9
7. Planning 10

1. Inleiding

Het Plan van Aanpak beschrijft de aanpak voor het project "Toolbox Formulierenmodule". Het beschrijft de aanleiding tot het project, hoe het project uitgevoerd gaat worden en wat de doelstellingen zijn.

2. Projectomschrijving

Probleemstelling

De websites die Eleven oplevert in de programmeertaal Java zijn gemaakt op het eigen ontwikkelde Toolbox. Toolbox is een web management systeem waar een website in zijn totaliteit beheerd kan worden door een website beheerder.

Toolbox bestaat uit diverse modules die een klant apart kan afnemen. Er is een basis module waarin de basis van het framework is opgenomen. De modules die nu gekozen kunnen worden zijn de volgende:

- CMS
- Catalogus
- Mailing
- CRM

Daarnaast kunnen er voor ieder project specifieke wijzigingen uitgevoerd worden aan Toolbox.

Toolbox is niet alleen een modulair systeem, maar ook een framework. Een geheel van softwarecomponenten en afspraken die verenigd zijn binnen Toolbox. Dit framework is de basis voor iedere module en specifieke toevoeging voor een klant.

Groot gemis is dat nu voor ieder formulier dat gevraagd wordt door de klant, het werk moet worden uitgevoerd door een ontwikkelaar van Eleven. Soms is het een heel nieuw formulier, maar vaker is het een extra veld erbij of een waarde toevoegen aan een drop-down formulier.

Dit kost tijd en zal ingepland moeten worden. Het werk wat door de ontwikkelaar gedaan moet worden is vrij triviaal en repeterend. Voor de klant is dit vervelend omdat de aanpassing vaak langer duurt dan nodig.

Ander probleem is dat binnen Eleven projecten via een niet gestandaardiseerde wijze verlopen. Gaandeweg de tijd zijn er eigen gewoonten en procedures ontwikkeld. Ook is er geleend vanuit andere methodes.
Dit is lastig voor nieuwe medewerkers en sommige klanten omdat zij niet bekend zijn met onze projectmethodiek. Het duurt dan ook even voordat een nieuwe medewerker of klant efficiënt meedraait in een project.

Doelstelling

Om de probleemstelling op te lossen ga ik gedurende de afstudeeropdracht een nieuwe module voor Toolbox maken. Met deze nieuwe module kan een beheerder op dynamische wijze zelf formulieren maken, de inhoud van deze formulieren bepalen en aangeven hoe ieder formulier afgehandeld moet worden.

Het voordeel voor de klanten is dat ze zelf formulieren kunnen aanmaken of wijzigen, zonder dat ze Eleven hoeven in te schakelen. Dit zal het gebruikersgemak voor de beheerder verhogen en extra mogelijkheden bieden.

Voor Eleven biedt deze nieuwe formulierenmodule een extra verkoopargument. Daarnaast zal deze module veel triviaal werk uit handen nemen van de ontwikkelaar en druk van de planning wegnemen. Dit omdat dit soort werkzaamheden vaak op korte termijn uitgevoerd moet worden.

Gezien de mogelijkheden van Toolbox WMS zou deze module goed als basis kunnen dienen voor toekomstige modules of op maat gemaakte uitbreidingen. Denk hierbij aan een reserveringsmodule voor kantoorruimte, hotels of congressen.

Om het probleem omtrent de projectmethodiek aan te pakken wil ik met dit project laten zien dat de methodiek Scrum goed past binnen Eleven. Dat we de houvasten en methodieken waarop Scrum gebaseerd is goed in te passen zijn bij Eleven en dat de methodiek zonder al te veel moeite in te passen is in het bedrijf.

Afbakening

Het project richt zich alleen op de ontwikkeling van de nieuwe formulierenmodule. Met deze module moeten een website beheerder op eenvoudige wijze een eigen gedefinieerd formulier kunnen plaatsen in de website. Tevens kan de website beheerder zelf bepalen wat er met het resultaat van de formulierenmodule moeten worden gedaan.

De exacte afbakening volgt door middel van de requirements sessie die met de Product Owner uitgevoerd zal worden.

3. Aanpak

Projectmethodiek

Voor dit project is gekozen voor de methodiek SCRUM. Deze agile methode kenmerkt zich door het werken in korte iteraties die sprints worden genoemd. Aan het eind van een sprint staat altijd een systeem met werkende functionaliteit.

De cyclus binnen Scrum ziet er als volgt uit:

Alle requirements worden verzameld in het product backlog. Dit is een register met alle geregistreerde user stories die ontwikkeld moeten worden voor het programma.

Bij aanvang van een sprint wordt er een sprint planning meeting opgesteld waarin bepaald word welke user stories uit het product backlog worden ontwikkeld in de komende sprint. Deze keuze word gemaakt door het team in samenspraak met de product owner.

In de sprint word er dan door het team gewerkt aan de user stories. Aan het begin van de dag wordt een " standup meeting" gehouden. In deze bespreking word in een kwartier tijd besproken wat er gisteren is gedaan, wat er vandaag gedaan gaat worden en of er problemen zijn opgetreden.
Aan het eind van de sprint word alle werkende functionaliteit getoond. Dat gebeurt in de Sprint Review Meeting waarin alle opgeleverde user stories worden getoond aan het gehele team.
In dit project duurt een sprint altijd 144 uur, uitgegaan van 18 dagen waarin 8 uur gewerkt wordt. Aangezien het afstuderen in deeltijd wordt uitgevoerd worden deze 144 uur over vijf weken verspreid.

Programmeertechnieken

Het project wordt uitgevoerd in de programmeertaal Java. Dit heeft voornamelijk een historische grondslag aangezien Toolbox WMS in Java is ontwikkeld en deze nieuwe module volledig moet integreren met Toolbox. Eleven maakt gebruik van Java versie 5.

Binnen Toolbox is er gebruik gemaakt van een aantal third-party libraries om het bulkwerk uit handen te nemen.

Belangrijke deelnemer hierin is Spring Framework. Deze library levert Toolbox een aantal functionaliteiten als:

- Dependency Injection Container
- Model-View-Controller architectuur
- Aspect oriënted programming

Vooral van het MVC principe en Dependency Injection wordt veelvuldig gebruik gemaakt binnen Toolbox. Ook in de nieuwe module zullen we gebruik maken van deze functionaliteiten.

Als database library is er binnen Toolbox gekozen voor Hibernate. Deze library acteert als een Object Relation Mapper en draagt zorg voor de verbinding met de database. Binnen Toolbox wordt Hibernate gebruikt met EhCache als een vluchtige cache voor alle Entities. Dit verlicht het aantal aanroepen naar de database en biedt daarmee een flinke winst op het gebied van performance.

Voor de documentatie en de ontwerp documenten maken we binnen dit project gebruik van het paradigma "Just barely good enough". Dat wil zeggen dat we aan deze documenten niet meer tijd steken dan nodig is. Dit moet voorkomen dat we gaan verzanden in details in de documentatie en ontwerpen.

De diverse diagrammen van de ontwerpen worden gemaakt door de modelleertaal UML. Deze taal beidt een grafische weergave van object georiënteerde applicaties. Binnen dit project zullen we, indien nodig, gebruik maken van de modellen:

- Domeinmodel
- Implementatie model
- Sequence diagram
- Deployment diagram

4. Projectorganisatie

Structuur

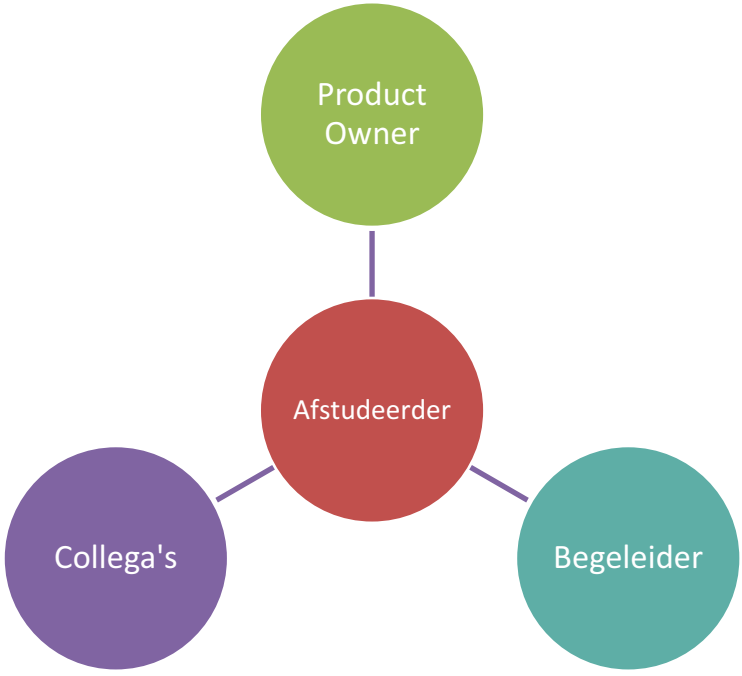
Alhoewel het afstudeerproject een eenmansproject is zijn er diverse mensen betrokken bij de ontwikkeling van de nieuwe formulieren module.

Paul de Raaij (Opdrachtnemer)
Zelf acteer ik als opdrachtnemer aangezien ik de nieuwe modulen zal gaan ontwikkelen. Binnen de scrum methodiek acteer ikzelf als ScrumMaster.

Carlo Vollebregt(Begeleider)
De opdracht is gegeven door Carlo Vollebregt, de technisch directeur van Eleven. Gedurende het project zal hij mij ook begeleiden. Carlo zal iedere scrum meeting aanwezig zijn om de voortgang te bespreken en mee denken met eventuele uitdagingen.

Peter Boon (Product Owner & Opdrachtgever)
Bewaker van Toolbox is Peter Boon. Niet alleen weet Peter wat klanten willen met de uiteindelijke module, maar zal Peter er ook zorg voor dragen dat de nieuwe module binnen het gedachtegoed van Toolbox past. Peter acteert dan ook als Product Owner en zal dan ook de prioriteit van de user stories bepalen.

Collega' s Eleven (Stakeholders)
Diverse collega's zullen in de toekomst moeten werken met de nieuwe module. Zowel functioneel als technisch. Met enige regelmaat zullen er dan ook collega's betrokken worden om naar hun mening te vragen.



Eveneens zal er in dit project gewerkt gaan worden met "hallway usability testing". Dit wil zoveel zeggen dat iemand spontaan, zonder enige voorbereiding achter het product wordt gezet om te zien hoe ze er mee werken. Dit laat de ontwikkelaar zien of de gekozen oplossing in de praktijk handig in het gebruik is voor iemand met geringe kennis van het product.

Resources

Betreffende materiaal zal er voor het project een werkplek beschikbaar gesteld moeten worden met daarbij een pc. Deze pc moet beschikken over de IDE Eclipse en de java development kit versie 5. De applicatie gaat uiteindelijk draaien op de applicatieserver Tomcat,v ersie 5.5. Die zal dus ook op de PC beschikbaar gesteld moeten worden.

De betrokkenen zullen ook hun tijd vrij moeten maken om mee te werken aan het project. Ze zullen minimaal twee weken van te voren een uitnodiging krijgen voor een bespreking zodat zij dit in hun planning kunnen inpassen. Eveneens zullen met de begeleider en product owner op gezette tijden een bespreking ingepland worden.

Tijdregistratie

Om de duur van diverse activiteiten te registreren maken we binnen dit project gebruik van de website toggl.com. Het voordeel van het gebruik van deze website is dat ik overal mijn tijd kan registreren onafhankelijk van mijn locatie. Eveneens levert de applicatie een prima rapportage die ik kan gebruiken in de verantwoording.

Projectnotities

Gedurende het project zal ik regelmatig tegen uitdagingen aanlopen. Om deze uitdagingen en de motieven achter de keuzes te registreren maak ik gebruik van het online notitieprogramma Springpad. De notities die ik hierin maak zullen als projectdocumentatie meegenomen worden, maar vormen ook mijn basismateriaal met het maken van het afstudeerverslag.

5. Risicomanagement

Zoals bij ieder project bevat ook dit project enkele risico's.

Risico: De scope van het project is onduidelijk

Door het dynamische karakter van Scrum en het bedrijf Eleven kan het gebeuren dat de scope van het project onduidelijk wordt en afwijkt van de opdracht. Om dit te voorkomen is het zaak om tijdens de sprint planning meetings de user stories te toetsen aan de projectopdracht in samenspraak met de product owner.

Risico: Het gestelde tijdspad is onrealistisch

In tegenstelling tot een voltijd student die voor een periode geplaatst wordt bij een bedrijf ben ik werkzaam bij Eleven en is het mijn full time baan. Mijn werkzaamheden aan het project zullen dan ook voornamelijk in de avonden en weekenden plaatsvinden. Wanneer de planning van Eleven het toelaat zal ik ook binnen mijn werktijden aan het project kunnen werken. Dit alles brengt als risico dat het een uitdaging kan worden om het gestelde werk binnen de projectperiode af te ronden.

Om dit te ondervangen heb ik bij aanvang van het project al rekening gehouden met een langer traject. De tijd voor de sprints zijn ook drie weken langer zodat ik meer tijd heb om het absolute aantal uur voor een sprint te kunnen maken.

6. Kwaliteitsborging

De diverse producten van dit afstudeerproject moeten wel van dusdanige kwaliteit zijn dat ze te gebruiken zijn in een professionele omgeving. Om dit te realiseren zullen een aantal activiteiten uitgevoerd moeten worden.

Testen

Voor het eindproduct zullen in de requirements sessie de kwaliteitsattributen besproken worden waarop het project moet voldoen.

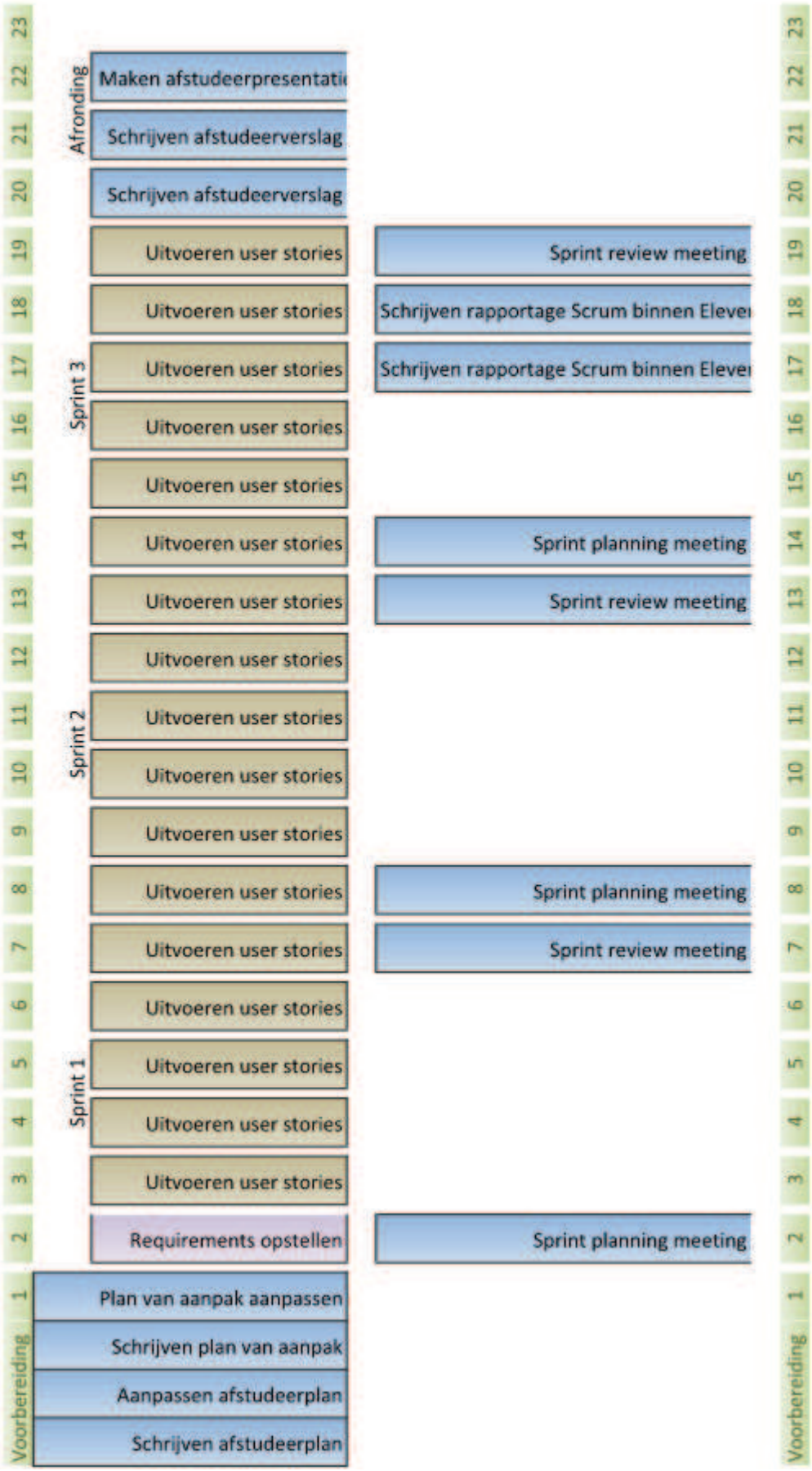
Op deze kwaliteitsattributen zal dan uitgebreid getest worden. Ligt de nadruk bijvoorbeeld op usabillity dan zal er veel nadruk op " hallway testing" en proefpersonen gelegd worden. Gaat het meer om betrouwbaarheid en functionaliteit dan zal het testtraject zich meer richten op functionele unit tests.

Het testen gebeurd via het principe van Agile testing om het makkelijk te integreren in de Agile methodiek van het hele project.

Review meetings

Binnen Scrum zijn diverse regelmatige besprekingen waarin de kwaliteit en voortgang besproken wordt. Het gaat dan om de Daily Standup meeting en de Sprint review meeting. Deze meetings zijn er op gericht om de kwaliteit en voortgang te bewaken.

7. Planning


Voorbereiding
Sprint 1
Sprint 2
Sprint 3
Afronding
Plan van aanpak aanpassen
Schrijven plan van aanpak
Aanpassen afstudeerplan
Schrijven afstudeerplan
Requirements opstellen
Uitvoeren user stories
Sprint planning meeting
Sprint review meeting
Schrijven rapportage Scrum binnen Eleve
Schrijven afstudeerverslag
Maken afstudeerpresentati
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

Vertaald naar daadwerkelijke data kom je tot het volgende schema:

Start datum	Eind datum	Activiteit
1 augustus 2011	13 augustus 2011	Sprint 0
15 augustus 2011	30 september 2011	Sprint 1
3 oktober 2011	11 november 2011	Sprint 2
14 november 2011	23 december 2011	Sprint 3

Master Test Plan

Ontwikkelen van een dynamische formulieren module binnen Toolbox WMS bij Eleven

Titel	Master Test Plan afstudeerproject
Student	Paul de Raaij (09081852)
Opdrachtgever	Eleven te Maasdijk
Datum	maandag 2 januari 2012 te Honselersdijk

Inhoudsopgave

1. Introductie 3
2. Testaanpak 4
   Kwaliteitsattributen 4
   Agile Testing 4
3. Testtechnieken 5
   Mock-ups 5
   Hallway testing 5
   Exploratory testing 5
   Unit testing 6
   GUI Testing 6
   User acceptance testing 6
4. Testomgeving 7
   Infrastructuur 7
   Continious Integration Server 7
5. Testproducten 9
   Test cases 9
   CI Metrieken 9
6. Risicoanalyse 10
   Nieuwe toepassing van testen binnen Eleven 10
   Omgeving onbekend met het onderwerp 10

1. Introductie

Het mastertestplan is de leidraad voor alle testprocessen gedurende het project. In dit mastertestplan staat namelijk beschreven op welke kwaliteitsattributen, en daarmee de waardevolste aspecten van de applicatie, de focus ligt met het testen.

Tevens beschrijft dit plan welke testtechnieken er gebruikt worden om de kwaliteit van de software te testen en welke rapportages en producten over gedragen worden aan de opdrachtgever bij oplevering van het product.

2. Testaanpak

Een applicatie volledig op alle facetten testen is een vrijwel onhaalbare en kostbare zaak. We richten ons dan ook op een aantal facetten van de applicatie op basis van kwaliteitsattributen.

Deze kwaliteitsattributen zijn beschreven in de ISO 9126 standaard. Deze standaard heeft als doel om enkele menselijke vooroordelen te beschrijven die invloed hebben op de kwaliteit en beleving van een software applicatie.

De kwaliteitsattributen zijn in overleg met de Product Owner uitgekozen. Door een aantal attributen te selecteren, zorgen we ervoor dat de testers de juiste onderdelen testen die voor de product owner van hoge waarden zijn. Juist deze attributen zorgen er voor dat de nieuwe formulieren module naar tevredenheid van de opdrachtgever zal worden opgeleverd.

Kwaliteitsattributen

De kwaliteitsattributen zijn bepaald aan de hand van de kwaliteitsattributen zoals beschreven in de ISO 9126 standaard. In overleg met de product owner hebben we besloten om er drie kwaliteitsattributen uit te pikken en daar de voornaamste aandacht tijdens het testtraject op te leggen. De gekozen attributen zijn:

- Understandability
- Learnability
- Maturity

Understandability en learnability zijn twee attributen die door de Product Owner zijn gekenmerkt als zeer belangrijk. Het idee achter Toolbox en haar modules is dat een gebruiker met minimale training aan de slag moet kunnen met het product zonder hulp vanuit een senior gebruiker of Eleven.

Een andere testfocus is het kwaliteitsattribuut maturity. Dit kwaliteitsattribuut staat voor de mate waarin storingen kunnen optreden in de applicatie door fouten in de software. Dit is onwenselijk gedrag en moet zo veel mogelijk voorkomen worden.

Het doel van onze testprocessen is dan ook niet om te laten zien dat de applicatie werk, maar om alle mogelijke fout paden te bedenken en die te testen op eventuele problemen.

Agile Testing

Dit project zal volgens de SCRUM methode worden opgepakt. Dit heeft niet alleen gevolgen voor het ontwikkelproces, maar ook voor het testproces. Dit proces zal nu ook op een agile wijze opgepakt worden. Zodra een user story nader wordt beschreven kunnen ook de testpaden beschreven worden. Bij het beschrijven van ieder testpad zal ook vrij snel duidelijk zijn welke testtechniek nodig is. De testtechnieken die toegepast worden binnen dit project staan beschreven in dit document.

Tijdens en na de ontwikkeling wordt gelijk de user story volgens het testontwerp getest. Als de ontwikkelaar de user story heeft afgerond kan de tester er mee aan de slag om de story te testen.
Voordeel van deze manier van werken is dat het testen veel korter plaats vindt op de ontwikkeling en dat daardoor resultaten efficiënter en beter verwerkt zullen worden omdat de probleemstelling en de requirements ook voor de ontwikkelaar nog vers zijn.

Daarnaast zijn er ook nog geen andere user stories die invloed hebben gehad op dat specifieke stukje functionaliteit. Eventuele wijzigingen hebben dan ook geen impact op andere user stories waardoor meerdere testen niet opnieuw uitgevoerd hoeven te worden.

3. Testtechnieken

Om de nieuwe formulierenmodule te testen maken we gebruik van diverse testtechnieken. Deze testtechnieken moeten ons de juiste handvaten bieden om de applicatie op een kwalitatief niveau op te leveren met zo min mogelijk fouten.

Mock-ups

Door mock-ups te gebruiken kunnen we de user interface testen aan (toekomstige) gebruikers van het systeem of de product owner zonder eerst de volledige user story uit te werken. Zo krijgen we snel helder of de user interface gebruiksvriendelijk is en of alle elementen duidelijk zijn.

Deze schermen helpen ons ook om te testen of de requirements op de juiste manier hebben begrepen en of er geen perceptieverschillen zijn ontstaan.

Mock-ups kunnen op diverse manieren gemaakt worden. Zo kan er gebruik worden gemaakt van paper prototyping. Een vorm waarbij gewoon op papier de mockup getekend word. Daarnaast kan er ook gebruik gemaakt worden van een whiteboard of door middel van het programma Balsamiq.
Alle mock-ups die worden gemaakt, in welke vorm dan ook, worden bewaard als project documentatie.

Hallway testing

Een ander manier om de vriendelijkheid en duidelijkheid van de user interface te testen is door de methode Hallway testing.

Hallway testing werkt door een of meer willekeurige personen uit te nodigen om aan de slag te gaan met een functionaliteit van het systeem. Deze personen moeten onbevangen zijn en bij voorkeur geen achtergrond met het systeem hebben.

Het doel is niet om de persoon te helpen om foutloos van A naar B te gaan, maar om te zien of de persoon überhaupt zelfstandig van A naar B kan gaan.

Interessante waarneming hierin is voornamelijk de paden waardoor de persoon van A naar B gaat. Is de user interface duidelijk genoeg om zelfstandig bij B te belanden. Heeft de gebruiker extra hulp nodig? Zijn de elementen duidelijk? Allemaal zinvolle constateringen die ons kunnen helpen in de ontwikkeling van de module.

Exploratory testing

Tijdens het testen van de applicatie zullen we ook gebruik maken van de techniek Exploratory Testing. Dit houdt in dat we zonder vast plan gaan testen en op basis van ervaring en gevoel specifieke onderdelen van de applicatie gaan testen.

Veel verantwoordelijkheid ligt er bij de tester die op basis van zijn ervaring en assertiviteit het systeem wel of niet intensief test op de juiste punten.

Omdat deze techniek zoveel vrijheid geeft is de kans groot dat we op nieuwe, onbedachte testpaden terecht komen. Testpaden die we in de meer formelere technieken wellicht over het hoofd gezien hebben en daardoor toch ondervangen.

Unit testing

Vrijwel alle code die geschreven gaat worden in de servicelaag zal getest worden door middel van unit tests. Unit tests zijn testen die een zo klein mogelijk stukje van de applicatie testen op een foutloze werking.

Ook hier gaat het er om dat we niet testen of de code werkt, maar dat het geen fouten bevat.

In onze applicatie maken we voor de unit tests gebruik van het test framework Junit. Deze unit tests vormen ook een onderdeel van het bouwproces in de Continious Integration server.

Het voordeel van het gebruik van deze testen in de CI server is dat de applicatie continu getest word bij iedere nieuwe build van de applicatie. Op deze manier is er zekerheid te geven over de betrouwbaarheid van de applicatie. Ook kunnen we voorkomen dat software die niet functioneert verder in de release cycle terecht komt. Dit is een automatisch proces waar geen manuele handeling bij komt kijken.

GUI Testing

Voor vele user stories kunnen we ook geautomatiseerde GUI tests schrijven. Deze tests controleren niet alleen of alle elementen aanwezig zijn in de GUI, maar ook of ze op de juiste manier werken.

Omdat ook deze testen geautomatiseerd uitgevoerd worden draaien ze ook mee in het bouwproces van de CI server. Zo word ook de user interface continu getest en hebben we controle over wat er uitgeleverd wordt bij iedere release. Op deze manier is het een stuk lastiger om een gebroken user interface te hebben in een applicatie.

Voor deze testen maken we gebruik van het framework Selenium. Dit framework biedt diverse mogelijkheden om de user interface op diverse manier te testen. Wij zullen gebruik maken van Selenium IDE en Selenium 2.

Selenium 2 biedt ons een geavanceerde API waar we op dezelfde wijze als bij Junit tests kunnen schrijven. De tool Selenium IDE is een add on op firefox waarin we eenvoudige acties kunnen opnemen. Deze acties kunnen we weer exporteren naar code voor Selenium 2.

User acceptance testing

Aan het eind van een sprint zal er een gebruikersacceptatietest plaatsvinden door de Product Owner. Hierin zal bepaald worden of alle volledig ontwikkelde user stories binnen de sprint voldoen aan de gestelde eisen.

In deze test wordt ook bekeken of de user stories gereed zijn voor productie en dus in een release opgenomen kunnen worden.

4. Testomgeving

Om goed en in een gecontroleerde omgeving te kunnen testen is het niet gewenst dat er op de ontwikkelomgeving getest gaat worden door de Product Owner en andere betrokkenen. Hiervoor zal een aparte testomgeving opgesteld worden die geïsoleerd is van de ontwikkel- en productieomgeving.

Daarnaast moeten ook de ontwikkelaars de voorziening krijgen over de verschillende test frameworks en relevante applicaties die in gebruik zijn.

Infrastructuur

Iedere ontwikkelaar moet de beschikking krijgen over de test frameworks JUnit en Selenium zodat de unit- en GUI tests ontwikkeld kunnen worden. Beide frameworks zijn open source en gratis te verkrijgen.

Een ontwikkelaar moet daarnaast ook verschillende browsers geïnstalleerd hebben zodat hij zijn gemaakte werk in ieder veel gebruikte browser kan testen. Het gaat dan om de volgende browsers:

- Internet Explorer(7, 8 & 9)
- Firefox (3.5 & 4)
- Google Chrome (12)
- Safari (5)
- Opera (11 & 12)

De tests worden door ontwikkelaas uitgevoerd op de ontwikkelomgeving. De testers werken op een separate testomgeving. Daarvoor zal er dus een testserver gefaciliteerd moeten worden die gelijk is aan de productieservers.

Aangezien de formulierenmodule een onderdeel van Toolbox zal er dus een ook test toolbox omgeving beschikbaar gesteld moeten worden. Uit de verschillende projecten die Eleven al heeft ontwikkeld is er gekozen om gebruik te maken van de omgeving van VDH foliekassen.

Dit is een schoon project waar geen interferentie met andere modules optreed. Dit zorgt er voor dat de test zich richt puur en alleen op de nieuwe module. Daarnaast is het ontwerp van VDH foliekassen een ontwerp zoals we ze regelmatig toepassen. We zien dus ook gelijk het effect van de nieuwe module op het ontwerp van de front-end websites.

Continious Integration Server

Belangrijk onderdeel in het waarborgen en beoordelen van de kwaliteit is de Continious Integration Server (CI Server). Deze server bouwt het project na iedere check-in in het versiebeheersysteem, in ons geval is SVN het versiebeheersysteem.

Binnen Eleven maken we gebruik van CruiseControl als CI server. Aangezien alle projecten en modules waar de nieuwe formulierenmodule verband heeft al in deze CI server staan, maken wij met deze nieuwe module ook gebruik van deze CruiseControl instantie. Dit biedt als voordeel dat eventuele verbanden met andere modules ook gelijk getest worden en eventuele integratie problemen ondervangen worden.

De CI server bij Eleven heeft een aantal taken die we voor dit project met alle liefde hergebruiken. Zo bouwt de CI server alle 'parent' projecten van de module. In ons geval is dat Toolbox, maar Toolbox an sich heeft ook weer parent projecten die gelijk gebouwd worden. De complete project hiërarchie wordt dus opnieuw gebouwd en getest.

Tijdens iedere build worden de geautomatiseerde unit- en GUI tests uitgevoerd. Als een van deze testen faalt wordt de totale build als mislukt beschouwd.

Daarnaast worden er tijdens een build diverse metrieken berekend. Deze metrieken gebruiken we om de kwaliteit van de software te bepalen. Hiervoor maken we gebruik van een aantal tools.

Findbugs
Findbugs is een statische code analyzer die honderden verschillende Java fouten kan onderkennen door te kijken naar de bytecode van het programma. De fouten zijn gecategoriseerd in diverse groepen. Voor de formulierenmodule moet de fouten in de groep 'correctness bug' lager dan 5% zijn.
Fouten in de groep 'correctness bug' zijn fouten die met grote waarschijnlijkheid een fout zijn met een resultaat die de ontwikkelaar niet bedoelde tijdens het ontwikkelen. Bijvoorbeeld een onmogelijke cast of onvolledige format strings.

Checkstyle
Ook Checkstyle is een statische code analyzer, maar deze richt zich op de uitlijning en formattering van de code. Het controleert of de code zoals die geschreven is voldoet aan de coding standaard die we binnen Eleven hebben opgesteld.
Alhoewel dit niet direct gevolg heeft op de wijze van uitvoering van de nieuwe module, zegt het wel veel over de kwaliteit van de code en de onderhoudbaarheid van de applicatie.

PMD
PMD is eveneens een statische code analyzer die veel veel gelijkenis met findbugs toont. Voornaamste reden om deze ook naast Findbugs te gebruiken is de uitstekende Copy/Paste detector.
Deze analyzer zoekt naar code die een of meerdere duplicaten heeft. Het geeft een melding als hij duplicaten vind en voor de ontwikkelaar is dat dan een signaal om te kijken of de code niet beter kan.

5. Testproducten

Alle testen die uitgevoerd worden zullen een resultaat opleveren. Het resultaat van de laatste testen voor een release zal onderdeel worden van de projectdocumentatie. Naast deze resultaten zijn er nog meer producten die het test proces oplevert.

Test cases

Tijdens het ontwikkelen en testen van een user story zullen er regelmatig test cases en testpaden beschreven worden. Deze test cases vormen een onderdeel van de projectdocumentatie en zullen altijd gearchiveerd worden zodat de testen eenvoudig herhaald kunnen worden, ook in een veel later stadium door een ander projectteam.

CI Metrieken

De metrieken die uit CruiseControl komen worden opgeslagen als projectdocumentatie. Het gaat hier dan voornamelijk om de laatste build die gedaan is om het product vrij te geven als release.

De metrieken van deze builds worden gebruikt om de kwaliteit van de applicatie bij een release te bepalen.

6. Risicoanalyse

Ook het testproces herbergt enkele risico's in zich. De grootste risico's hebben we onderkend en maatregelen voor getroffen.

Nieuwe toepassing van testen binnen Eleven

Binnen Eleven is er nog niet eerder op een heel formele manier gewerkt met testen. Er wordt wel gebruik gemaakt van unit testing en end-to-end testen, maar dit wordt niet geformaliseerd en op papier beschreven.

Het test proces zoals hier beschreven is dan ook formeler dan tot nu toe eerder gedaan bij Eleven. Het zal dan dus ook even wennen zijn en een kleine cultuurwijziging met zich mee brengen.

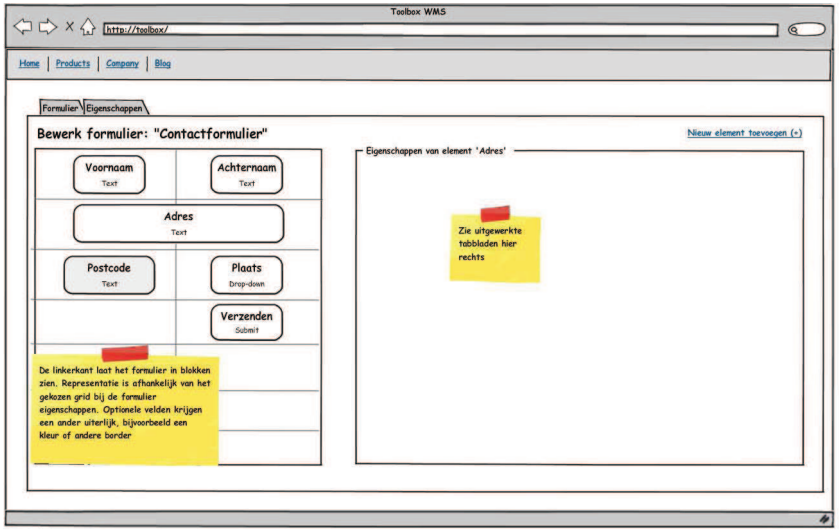
Er bestaat een kans dat er snel buiten de formaliteit gewerkt zal worden. Dit moet voorkomen worden door veel te communiceren over het belang van de formaliteiten en het verplicht toevoegen aan de projectdocumentatie.

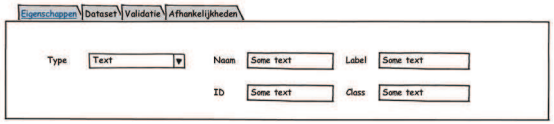
Omgeving onbekend met het onderwerp

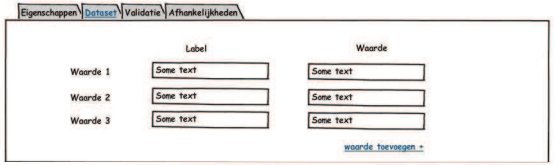
Aangezien Agile Testing voor ons allemaal nieuw is, is de kennis binnen Eleven over dit onderwerp minimaal. Alle kennis en eventuele hulp bij problemen komen dus van buitenaf.

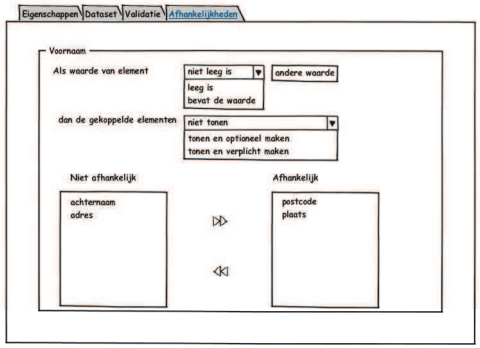
Er bestaat dan ook een risico dat de toepassing van Agile Testing niet volledig tot zijn recht komt door gebrek aan kennis van het onderwerp.

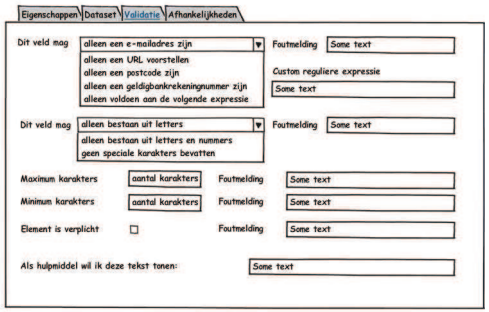
Om dit te voorkomen hebben we een boek aangeschaft over Agile Testing enlezen we op internet veel bij over Agile Testing. Niet alleen de afstudeerder zal deze informatie gaan lezen, maar ook de bedrijfsmentor zodat we het ideale Agile Testing pad voor Eleven kunnen bepalen en volgen en elkaar scherp kunnen houden.

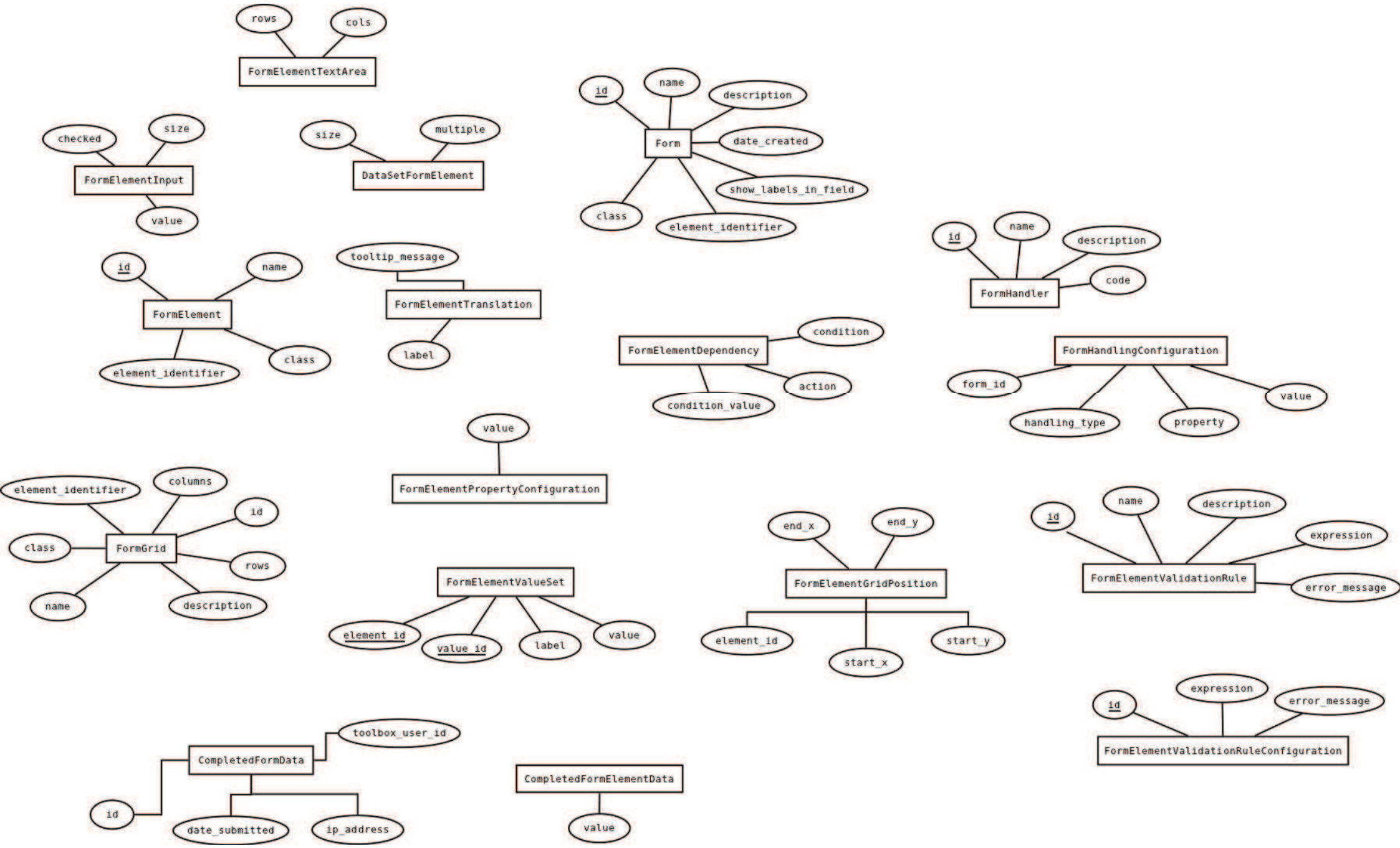

Toolbox WMS
http://toolbox/
Home | Products | Company | Blog
Formulier | Eigenschappen
Bewerk formulier: "Contactformulier"
Nieuw element toevoegen (+)
Voornaam
Text
Achternaam
Text
Adres
Text
Postcode
Text
Plaats
Drop-down
Verzenden
Submit
De linkerkant laat het formulier in blokken zien. Representatie is afhankelijk van het gekozen grid bij de formulier eigenschappen. Optionele velden krijgen een ander uiterlijk, bijvoorbeeld een kleur of andere border
Eigenschappen van element 'Adres'
Zie uitgewerkte tabbladen hier rechts

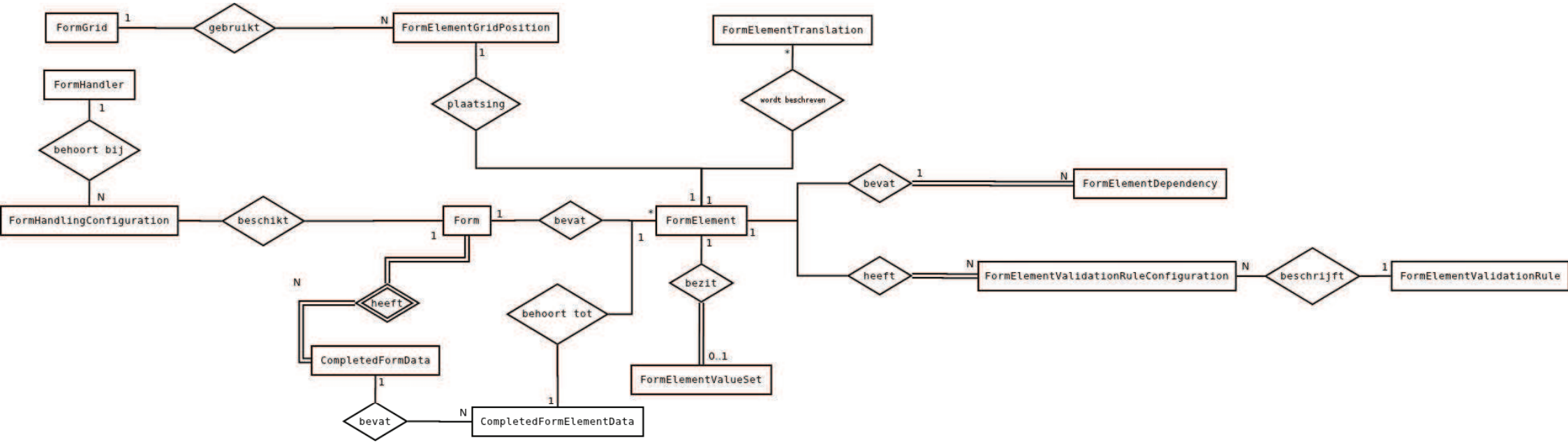

Eigenschappen | Dataset | Validatie | Afhankelijkheden
Type
Text
Naam
Some text
Label
Some text
ID
Some text
Class
Some text


Eigenschappen | Dataset | Validatie | Afhankelijkheden
Label
Waarde
Waarde 1
Some text
Some text
Waarde 2
Some text
Some text
Waarde 3
Some text
Some text
waarde toevoegen +


Eigenschappen | Dataset | Validatie | Afhankelijkheden
Voornaam
Als waarde van element
niet leeg is
leeg is
bevat de waarde
andere waarde
dan de gekoppelde elementen
niet tonen
tonen en optioneel maken
tonen en verplicht maken
Niet afhankelijk
achternaam
adres
Afhankelijk
postcode
plaats

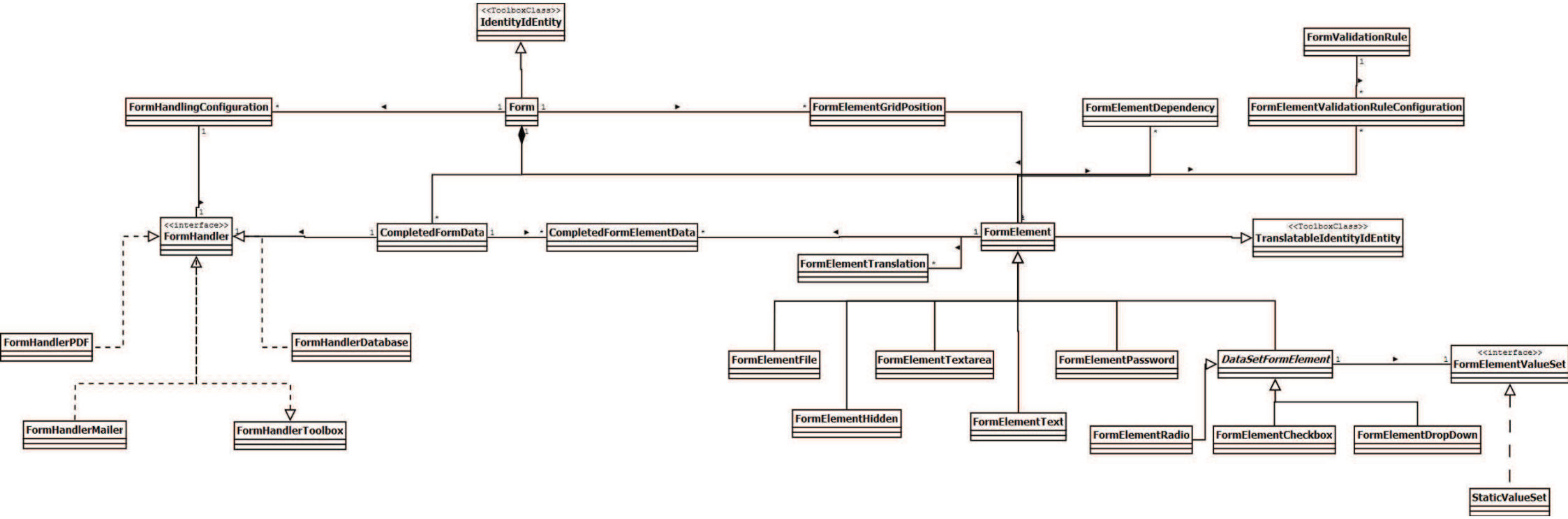

Eigenschappen | Dataset | Validatie | Afhankelijkheden
Dit veld mag
alleen een e-mailadres zijn
alleen een URL voorstellen
alleen een postcode zijn
alleen een geldigbankrekeningnummer zijn
alleen voldoen aan de volgende expressie
Foutmelding
Some text
Custom reguliere expressie
Some text
Dit veld mag
alleen bestaan uit letters
alleen bestaan uit letters en nummers
geen speciale karakters bevatten
Foutmelding
Some text
Maximum karakters
aantal karakters
Foutmelding
Some text
Minimum karakters
aantal karakters
Foutmelding
Some text
Element is verplicht
Foutmelding
Some text
Als hulpmiddel wil ik deze tekst tonen:
Some text


rows
cols
FormElementTextArea
checked
size
FormElementInput
value
size
multiple
DataSetFormElement
id
name
description
Form
date_created
show_labels_in_field
class
element_identifier
id
name
FormElement
element_identifier
class
tooltip_message
FormElementTranslation
label
id
name
description
FormHandler
code
FormElementDependency
condition
action
condition_value
FormHandlingConfiguration
form_id
handling_type
property
value
value
FormElementPropertyConfiguration
element_identifier
columns
id
class
FormGrid
rows
name
description
FormElementValueSet
element_id
value_id
label
value
end_x
end_y
FormElementGridPosition
element_id
start_x
start_y
id
name
description
expression
FormElementValidationRule
error_message
id
expression
error_message
FormElementValidationRuleConfiguration
toolbox_user_id
CompletedFormData
id
date_submitted
ip_address
CompletedFormElementData
value


FormGrid
1
gebruikt
N
FormElementGridPosition
FormElementTranslation
*
wordt beschreven
1
plaatsing
FormHandler
1
behoort bij
N
FormHandlingConfiguration
beschikt
Form
1
1
bevat
*
FormElement
1
1
1
1
1
bevat
1
N
FormElementDependency
heeft
N
FormElementValidationRuleConfiguration
N
beschrijft
1
FormElementValidationRule
N
heeft
behoort tot
1
bezit
0..1
FormElementValueSet
CompletedFormData
1
bevat
N
CompletedFormElementData

1

1


<<ToolboxClass>>
IdentityIdEntity
FormValidationRule
FormHandlingConfiguration
Form
FormElementGridPosition
FormElementDependency
FormElementValidationRuleConfiguration
<<interface>>
FormHandler
CompletedFormData
CompletedFormElementData
FormElement
<<ToolboxClass>>
TranslatableIdentityIdEntity
FormElementTranslation
FormHandlerPDF
FormHandlerDatabase
FormElementFile
FormElementTextarea
FormElementPassword
DataSetFormElement
<<interface>>
FormElementValueSet
FormHandlerMailer
FormHandlerToolbox
FormElementHidden
FormElementText
FormElementRadio
FormElementCheckbox
FormElementDropDown
StaticValueSet

Implementatieadvies

Scrum bij Eleven

Titel	Scrum bij Eleven
Student	Paul de Raaij (09081852)
Opdrachtgever	Eleven te Maasdijk
Datum	maandag 2 januari 2012 te Honselersdijk

Inhoudsopgave

1. Inleiding 4
2. De methode scrum 5
   - Agile methode 5
   - Oorsprong 5
   - Doelstellingen 5
   - Empirisch proces 5
3. Onderdelen van scrum 6
   - Vergaderingen 6
     - Daily Scrum 6
     - Sprint planning meeting 6
     - Sprint review meeting 7
     - Sprint Retrospective meeting 7
   - Artefacten 7
     - Product backlog 7
     - Sprint backlog 7
     - Burndown chart 8
   - Technieken/Hulpmiddelen 8
     - Planning poker 8
     - Definition of Done 9
     - Taken 9
     - Sprint 9
     - User story 10
     - Velocity 10
   - Rollen 11
     - Scrum master 11
     - Scrumteam 11
     - Product Owner 11
4. Huidige inzet scrum bij Eleven 12
   - Project kick-off 12
   - Daily Scrum 12
   - Weekly scrum 12
   - Planning poker & taken 12
5. Aanbevolen onderdelen 13
   - Rollen 13
   - Vergaderingen 13

Sprint planning meeting 13
Daily scrum 13
Sprint retrospective meeting 14
Artefacten 14
Burndown chart 14
Hulpmiddelen 14
Velocity 14
Puntenschaal/Story points 15
Definition of Done 15
User story 16
6. Advies implementatie 17
7. Literatuurlijst 18

1. Inleiding

Voor u ligt het rapport Scrum bij Eleven. In dit rapport word beschreven hoe de agile software ontwikkelmethode Scrum op dit moment bij Eleven wordt toegepast.

Dit rapport beschrijft diverse onderdelen van de methode Scrum die nu nog niet toegepast worden bij Eleven of waar nog verbeteringen mogelijk zijn. Daarnaast geeft het adviezen over toepassingen en implementatie van onderdelen.

De totstandkoming van dit rapport vind zijn oorsprong in de afstudeeropdracht van Paul de Raaij uitgevoerd in 2011. Als sub opdracht van het afstuderen was opgenomen dat er een advies gegeven zou worden over de inzet van Scrum bij Eleven.

2. De methode scrum

Agile methode

Scrum is een software ontwikkelingsmethode die afkomstig is uit de agile methodiek. Agile is een Engels woord voor 'behendig' en 'lenig'. Een agile methode is er dan ook op gespitst om snel op wijzigingen in het project te reageren.

Scrum richt zich specifiek op samenwerking en communicatie. Het draait in essentie om het team en de filosofie is er dan ook op gericht dat het team goed met elkaar kan communiceren en samenwerken om het project soepel tot een eind te brengen.

Oorsprong

De scrum methode is voor het eerst publiekelijk genoemd in 1986 in de Harvard Business Review door de onderzoekers Ikujiro Nonaka en Hirotaka Takeuchi. In de publicatie is beschreven dat projecten met kleine, kruislings functionele, team historisch gezien het beste resultaat opleveren.

Deze publicatie vormde in 1993 de basis voor het scrum-proces zoals ontwikkeld door Jeff Sutherland. Gelijktijdig ontwikkelde Ken Schwaber een eigen benadering voor zijn bedrijf. De mannen ontwikkelde samen de scrum methode en formaliseerde de methode in 1995 tot hoe we het nu kennen.

Scrum is een term die afkomstig is uit de rugbysport. In een scrum staan de spelers in een groep en proberen ze al duwend de bal naar de andere kant van het veld te brengen om punten te scoren.

Doelstellingen

De methode scrum heeft een aantal doelstellingen die de methode binnen een organisatie kan behalen.

- Verhogen van de effectiviteit van het team
- Het bewaken van de vooruitgang van het team
- Het oplossen van blokkades
- Het bewaken van de projectvoortgang
- In kaart brengen en minimaliseren van de risico's

Empirisch proces

Scrum is een empirische[1] methode. De deelnemers en gebruikers van Scrum worden beter in het proces doordat zij leren van opgedane ervaringen en kennis. In de methode zitten dan ook een aantal reflectiemomenten waarin de organisatie kritisch naar zichzelf dient te kijken om daar weer van te leren en te verbeteren.

[1] Empirisme is een filosofische stroming waarin gesteld wordt dat kennis uit ervaring voorkomt. Aristoteles is een van de personen die aan deze stroming gekoppeld wordt.

3. Onderdelen van scrum

De methode scrum bestaat uit een grote set van tools, artefacten, technieken en rollen. In dit hoofdstuk zullen alle onderdelen besproken en toegelicht worden.

Vergaderingen

Eén van de belangrijkste speerpunten van scrum is het verhogen en verbeteren van de communicatie binnen het team. Daartoe kent de methoden een aantal vergadersessies die ieder een specifiek doel dient.

Daily Scrum

Iedere dag tijdens het project vindt er een status overleg plaats. Dit overleg word de daily scrum of daily standup genoemd. Aan deze vergadering zijn een aantal vereisten van toepassing.

- De vergadering start precies op tijd
- Iedereen is welkom, maar alleen het team spreekt
- De vergadering heeft een vaste duur van vijftien minuten
- De vergadering start altijd op hetzelfde tijdstip en vindt plaats op dezelfde locatie

Belangrijke vereiste is dat het team blijft staan tijdens de vergadering zodat men actief deelneemt aan de vergaderingen en het niet onnodig uitloopt. Door vast te houden aan deze voorwaarden komt er duidelijkheid in het team en zal het snel in de routine van het team slijten.

Doel van deze vergadering is om de voortgang van het project te monitoren. Andere belangrijke reden is het vroegtijdig identificeren van risico's en problemen die het team ziet. Hiertoe moet ieder lid van het team de volgende vragen beantwoorden:

1. Wat heb je gedaan sinds gisteren?
2. Wat ga je vandaag doen?
3. Zie je belemmeringen of enige risico's?

De vergadering wordt voorgezeten door de ScrumMaster. Het is zijn rol om de vergadering te leiden en de blokkeringen die door het team aangegeven worden te beoordelen en een oplossing voor te verzinnen.

Om de vergadering binnen de timebox van vijftien minuten te houden zal de oplossing buiten de daily scrum besproken moeten worden. Het verzinnen van de juiste oplossing moet namelijk niet de rest van het team en de daily scrum ophouden.

Sprint planning meeting

De start van iedere sprint wordt vooraf gegaan door de sprint planning meeting. Deze vergadering bestaat uit twee delen en is ook gebonden aan een tijdslimiet, namelijk acht uur.

Het eerste deel van de sprint planning meeting duurt vier uur en is bedoeld om de product backlog aan te vullen en de prioriteit van iedere item in het backlog vast te stellen. Hiervoor is zowel het team als de product owner aanwezig.

Na de prioritering van de diverse items zal het team, zonder de product owner, een plan voor de komende sprint opstellen. Dit plan wordt de sprint backlog en geeft de user stories aan die uitgevoerd gaan worden.

Sprint review meeting

Aan het eind van ieder sprint worden er twee vergadering gehouden. Eén vergadering daarvan is de sprint review meeting en duurt maximaal vier uur. In deze vergadering wordt het werk wat is gedaan besproken. Hierbij maakt het niet uit of het werk compleet of incompleet is.

Het werk wat wel af is wordt getoond aan de stakeholders van het project en staat ook wel bekend als de demo. Het werk wat niet afgerond is, mag niet getoond worden.

Sprint Retrospective meeting

Waar in de sprint review meeting het gemaakte werk centraal staat, gaat het in de sprint retrospective meeting over de samenwerking en het verloop van de sprint. De input die in deze vergadering word gegeven door het team vormt de leidraad om de komende sprints beter te laten verlopen.

In deze vergadering staan er twee vragen centraal:

- Wat ging er goed in deze sprint?
- Wat kan er beter in de volgende sprint?

Op deze manier is het team consequent met het eigen werken bezig en maakt het team continu procesverbeteringen. Het leert van zijn eigen fouten en tekortkomingen en zal het project in het vervolg beter doorlopen worden.

Artefacten

Binnen iedere methodiek ontstaan er documenten en producten als gevolg van de processen in de methodiek. Scrum kent ook een aantal artefacten die er op gericht zijn om de communicatie en voortgang van het project te beschrijven. Artefacten kunnen documenten, ontwerpen, modellen of grafieken zijn. Kortom alle materialen die gemaakt worden ten bate van de uitvoer van het project,

Product backlog

Een belangrijk onderdeel binnen Scrum is het product backlog. Dit backlog is een lijst van backlog items wat gedurende het hele project onderhouden wordt. Ieder backlog item geeft een algemene beschrijving van mogelijke functionaliteiten. In deze lijst worden deze items gesorteerd op hun business waarde.
Ieder item beschikt over een ruwe inschatting van wat het oplevert voor de business en de ontwikkelingsinzet om de feature te implementeren. Deze inschattingen zijn van belang zodat de Product Owner over een handvat beschikt om de tijdlijn in de peiling te houden.

De product owner kan door deze getallen de juiste keuzes maken als er bepaald moet worden welke user stories in de komende sprint uitgevoerd moeten worden. Als story A & B beide een business waarde van 8 hebben, maar B 2 uur korter ontwikkeld kan worden. Dan zal de product owner voor story B kiezen omdat de Return On Investment dan hoger is. Accurate inschattingen zijn dan ook van belang.

De items en de business waarde van iedere item zijn een verantwoordelijkheid van de Product Owner. De ontwikkelingsinzet is een verantwoordelijkheid van team, voornamelijk van de ontwikkelaars.

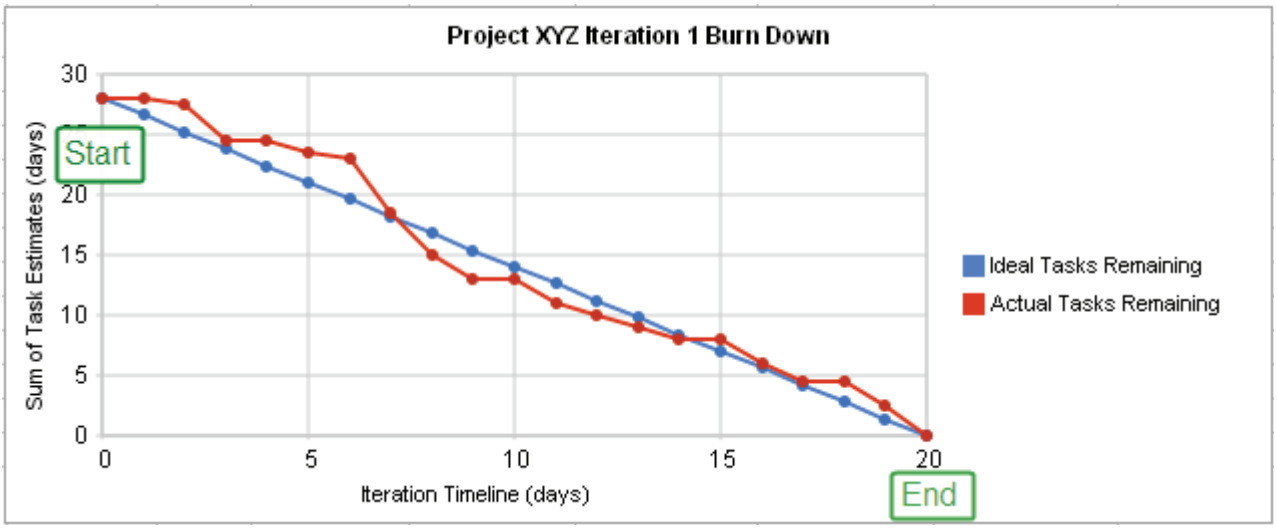
Sprint backlog

Ieder sprint is gebaseerd op een sprint backlog. In dit backlog staat beschreven welke user stories er in deze sprint worden opgepakt. Het verdeelt de user stories in taken, die bij voorkeur tussen de vier en zestien uur in beslag nemen.

Door deze verdeling van de user stories worden de taken dusdanig eenvoudig en duidelijk dat in feite ieder teamlid de taak zou kunnen oppaken. Taken in de sprint backlog worden nooit toegekend aan een persoon, maar worden aangemeld voor teamleden. Dit is afhankelijk van de prioriteit en de skills van de teamleden. Het team is eigenaar van de sprint backlog, evenals alle inschattingen die gemaakt zijn in dit backlog.

Burndown chart

De burndown chart is een grafische weergave van de voortgang van de sprint. Het laat zien hoeveel werk er nog uitgevoerd moet worden en hoeveel werk er al verzet is.



De grafiek bestaat doorgaans uit twee assen en twee lijnen waarvan er één linear is. De x-as is de tijdlijn van het project. Aangezien de grafiek dagelijks wordt bijgewerkt is het verstandig om deze x-as ook in de eenheid van dag op te zetten.

De y-as toont het werk wat gedaan moet worden. Hier kan het team kiezen voor een eenheid. Dit kan het werk in dagen, functiepunten of uren zijn. De methode laat het project team vrij in deze keuze.

Lijn 1, getiteld ideal tasks remaining, is linear en laat zien wat de ideale burndown van het werk is. De andere lijn, getiteld: actual tasks remaining, is de representatie van het werkelijk uitgevoerde werk. Zodra een user story is afgerond neemt het werk te doen af en zal de lijn dichter bij de lijn met het ideale taak verloop komen.

Als de actuele lijn zich boven de ideale lijn bevindt zal er geanalyseerd moeten worden waarom er minder werk is uitgevoerd dan er idealiter gedaan had moeten zijn. Het kan zijn omdat user stories zich in de afrondende fase zitten en op korte termijn afgerond worden. Het kan zijn dat er problemen zijn die opgelost moet worden.

In ieder geval laat de burndown chart zien wat de status van het project is en of er ingegrepen moet worden. Dit kan door het user stories door te schuiven naar de volgende sprint of door meer resources beschikbaar te maken.

Technieken/Hulpmiddelen

Voor een scrum team zijn vele technieken en hulpmiddelen beschikbaar om het project optimaal uit te voeren. Dit hoofdstuk beschrijft een groot deel van deze aanwezige technieken.

Planning poker

In de diverse artefacten beschikken de items vaak over inschattingen die nodig zijn om een correct beeld te krijgen over de waarde en ontwikkelingsmoeite van een item. Een inschatting moet realistisch en gedragen worden door het team. Voor het team is het dan ook zaak om overeenstemming over de inschatting te hebben.

Om te zorgen dat het team overeenstemming heeft over de inschatting kan het team Planning Poker spelen. Planning poker is een techniek om een team een inschatting te laten maken.

Ieder teamlid die deelneemt aan de sessie beschikt over een pak kaarten. Deze kaarten bestaan doorgaans uit de Fibonacci reeks, oftewel 0, 1, 2, 3, 5, 8, 13, 21, 34, 55. Daarnaast bestaat de set vrijwel altijd ook uit een vraagteken en koffie kaart.

Met de vraagteken kan de deelnemer aangeven dat de user story niet duidelijk genoeg is om een inschatting te maken en dat de user story eerst verder uitgewerkt zal moeten worden. De koffie kaart wordt opgegooid als de deelnemer behoefte heeft aan een pauze.

De sessie zal worden voorgezeten door een moderator die geen deelname heeft aan de stemmingen. De product owner is eveneens aanwezig om toelichting te geven op de user story. Het team kan vragen stellen om de risico's en aannames inzichtelijk te maken. Overigens is het de taak van de ScrumMaster om ervoor zorg te dragen dat deze gesprekken genotuleerd worden voor het verdere vervolg.

Na de toelichting volgt de stemming. Iedere deelnemer legt gelijktijdig zijn kaart op tafel. De deelnemers met de hoogste en de laagste inschatting krijgen de kans om hun inschatting toe te lichten. Deze toelichting laat zien hoe ze tot deze inschatting zijn gekomen. Hiermee kunnen interpretatie fouten of verkeerde verwachtingen tevoorschijn komen.

Dit proces herhaalt zich totdat er consensus van het team is over de inschatting. Om eindeloze discussies te voorkomen zal er een tijdslimiet gehangen worden aan deze stemmingsronde. Indien het team geen consensus bereikt over de inschatting stapt men over op de volgende inschatting en komt men later terug bij deze inschatting.

Definition of Done

De definition of done is een eenduidige richtlijn die bepaalt wanneer een user story als af kan worden bestempelt. De definition of done zorgt er voor dat iedere user story aan minimale vereisten voldoen en over een zekere kwaliteit beschikken als ze opgeleverd worden.

De definition of done wordt in de beginfase van het project opgesteld. Ze worden door het gehele team opgezet zodat er draagvlak is binnen het team en de leden gemotiveerder zullen zijn om te voldoen aan deze richtlijnen.

Taken

Iedere user story word aan het begin van de sprint opgedeeld in taken die tussen de vier a zestien uur werk zijn. Het werk wat gedaan moet worden is hierdoor opgedeeld in werkbare stukken die eenvoudig door meerdere ontwikkelaars uitgevoerd kunnen worden.

Dit zorgt er ook voor dat risico's meer gespreid kunnen worden over het team. Mocht een teamlid ziek uitvallen kunnen de resterende taken door zijn collega's eenvoudig opgepakt worden.

Sprint

Een sprint is een periode in het project waarin actief aan het project gewerkt wordt. Een sprint duurt doorgaans minimaal een week tot maximaal vier weken. Het levert altijd een werkend product op. De werkzaamheden die in een sprint uitgevoerd worden staan beschreven in het sprint backlog.

User story

Een user story beschrijft in één of meer zinnen de gewenste functionaliteit wat een gebruiker wilt bereiken. Dit word beschreven in de gebruikelijke of business taal.

Iedere user story is gelimiteerd om slechts één functionaliteit te beschrijven. De story is doorgaans geschreven door of voor een gebruiker van het project. Een user story kan ook geschreven zijn door een persoon met een technische achtergrond. Deze user stories beschrijven dan niet functionele eisen omtrent kwaliteit, performance en beveiliging.

Een user story word doorgaans in de volgende vorm geschreven:

```
Als een <ROL>, wil ik dat <wens> zodat <voordeel>
```

Velocity

Een handig kenmerk om productiviteit te meten is velocity. Het laat het tempo zien waarmee het team werkt aan een sprint. Dit kenmerk is een handig hulpmiddel bij de sprint planning meeting om te bepalen hoeveel 'eenheden' aan werk uitgevoerd kan worden.

Als een sprint vier weken duurt en er zijn drie ontwikkelaars fulltime beschikbaar dan kan er 480 uur aan werk uitgevoerd worden. Nu is het niet realistisch om aan te nemen dat de ontwikkelaar 100 procent van zijn tijd volledig aan ontwikkeling van user stories kan besteden. Een ontwikkelaar kan ziek worden of moet zijn tijd ineens aan andere projecten besteden.

Aan het eind van de sprint blijkt dat het team voor 288 uur aan werk heeft kunnen uitvoeren. Een ontwikkelaar was een aantal dagen ziek, één had een studiedag en de andere moesten onderhoud plegen aan andere projecten.

De velocity van het projectteam is dan 288 (gewerkte uren) / 480 (beschikbare uren) = 0.6. Deze factor geeft de productiviteit van het team aan en zorgt er voor dat er voor de komende sprint een realistische hoeveelheid aan werk ingepland kan worden. Dit voorkomt teleurstellingen voor de product owner en het team en geeft een hoeveelheid werk die ook daadwerkelijk uitgevoerd kan worden.

De velocity kan ook voor personen apart berekend worden door een bepaalde periode te nemen. Van die periode wordt berekend hoeveel werk er gedaan had moeten worden en hoeveel er ook daadwerkelijk is gedaan. Bijvoorbeeld, in de weken 22 t/m 34 was er 480 uur beschikbaar, maar is er voor 300 uur werk aan projecten uitgevoerd, de overige uren waren allemaal support of vergaderuren. De velocity van die persoon is dan (300 / 480 =) 0,625. Van een week kan er dan maar 40 * 0.625 = 25 uur ingepland worden voor projecten. De overige uren zouden beter vrijgehouden kunnen worden voor onderhoudsuren en contactmomenten.

Rollen

Binnen een scrum project zijn er diverse rollen die aangenomen moeten worden om zorg te dragen voor een vlot verloop van het project.

Scrum master

De scrummaster is de persoon die verantwoordelijk wordt gesteld voor het scrum team. Het is zijn of haar taak om iedere blokkade die het team ondervindt om deliverables op te leveren weg te nemen. De scrummaster is niet per definitie de leider van het team. Hij of zij acteert als buffer tussen het team en derden. De ScrumMaster moedigt het team aan om continu te verbeteren zodat het team productiever en met meer plezier aan het werk gaat in de komende sprint.

Scrumteam

Het team heeft de verantwoordelijkheid om het product te ontwikkelen en op te leveren. Doorgaans bestaande uit vijf tot negen personen, beschikt het over een variëteit aan leden met diverse functionele vaardigheden.

Over het algemeen is het team zelf organiserend en is er geen formele project of teamleider aanwezig.

Product Owner

De Product Owner is de afgevaardigde van de klant en moet ervoor zorg dragen dat het opgeleverde product daadwerkelijk business waarde oplevert en voldoet aan de wensen van de klant. Het kan zijn dat de Product Owner spreekt voor een comité, maar de rol word altijd door één persoon gespeeld.

Hij of zij is verantwoordelijk voor de user stories aanwezig in het Product Backlog en de prioritering hiervan. De Product Owner moet er zorg voordragen dat de items die in het Product Backlog staan, begrepen worden door het ontwikkelteam tot een niveau dat nodig is voor ontwikkeling.

Het is toegestaan dat de rol van product owner word gedragen door een lid van het team. Bij voorkeur word deze rol niet door de scrum master op zich genomen omdat deze rollen conflicterende belangen hebben.

4. Huidige inzet scrum bij Eleven

Eleven maakt op dit moment al gebruik van de scrum methode. Echter gebruiken zij niet de volledige methode en hebben ze een aantal onderdelen uit de methode gepakt. Voordat een gedegen advies mogelijk is, een korte inventarisatie van de huidige inzet van deze onderdelen.

Project kick-off

Voor de start van ieder project wordt het team bij elkaar geroepen om het project te introduceren. Voornaamste reden is de achtergrond van de opdracht toe te lichten en het design van de website te bespreken.

Aan de hand van het design worden de functionaliteiten die men dan tegen komt besproken.

Daily Scrum

Een veelgebruikt onderdeel is de daily scrum. In vrijwel ieder project waar meer dan twee personen werkzaam zijn wordt iedere ochtend een daily scrum gehouden.

De daily scrum wordt voorgezeten door de projectleider. Hij of zij stelt de drie vragen conform de methode gewenst aan ieder lid van het team.

Wat opvalt is dat als een lid aangeeft een probleem of risico te voorzien word geprobeerd in die vergadering al de oplossing te vinden. Hierdoor lopen deze scrums ook vaak uit en duren ze langer dan de gestelde timebox.

De projectleider heeft regelmatig een dubbelrol in deze besprekingen. Hij of zij speelt zowel de scrummaster als de product owner. Soms is de accountmanager aanwezig die de rol van product owner dan op zich neemt.

Ook de starttijd van de vergadering wisselt regelmatig van tijdstip.

Weekly scrum

Een instrument wat scrum niet formeel definieert, maar binnen Eleven wel toegepast wordt is de Weekly Scrum. Deze scrum is een combinatie van een daily scrum en een sprint review meeting.

Bij deze scrum is de klant aanwezig die vaak ook de rol van product owner toegewezen krijgt. Voor het team is het mogelijk om openstaande vragen beantwoord te krijgen van de product owner, in dit geval de klant.

Een ander belangrijk facet in deze bespreking is het tonen van het gemaakte werk aan de klant. Vaak wordt er een korte demonstratie van het nieuwe opgeleverde werk gegeven.

Planning poker & taken

Aan de start van ieder project word er door de technisch manager van het project een work breakdown gemaakt. In deze breakdown worden de user stories verdeeld in taken en ingevoerd in het project management systeem.

De technisch manager kan ervoor kiezen om het project team bij elkaar te roepen om voor de inschattingen van deze taken planning poker te spelen.

5. Aanbevolen onderdelen

Eleven maakt al gebruik van een aantal scrum instrumenten, maar er zijn nog meer instrumenten die Eleven kan toepassen om projecten soepeler en gestroomlijnder te laten lopen. In dit hoofdstuk word de ideale situatie besproken. In het volgende hoofdstuk zal er over de implementatie geadviseerd worden.

Rollen

Zoals geschetst in het vorige hoofdstuk is de projectleider op dit moment degene in de rol van ScrumMaster. Omdat hij of zij over het algemeen ook als product owner acteert zorgt dat voor verwarring in het proces.

Het advies is dan ook om de technisch manager van ieder project de rol van ScrumMaster toe te kennen. Hierdoor is er duidelijkheid in de structuur van het project, maar ligt de verantwoordelijkheid bij degene die er vanuit zijn functie toch al zorg voor moet dragen dat het team zich kan focussen op het project.

De project leider kan zich dan richten op haar rol als Product Owner. Dit verlicht de druk op de accountmanager om actief in het project betrokken te zijn. Het geeft de projectleider meer de ruimte om de wensen van de klant te vertegenwoordigen.

Vergaderingen

Sprint planning meeting

Op dit moment kent Eleven geen fatsoenlijke planning meeting. Wat men nu heeft is de project kickoff, maar dan zijn alle keuzes vaak al gemaakt. Deze keuzes worden vaak gemaakt door de technisch manager en die legt daarmee zijn keuzes op aan het ontwikkelteam.

Door gebruik te maken van de sprint planning meeting kunnen keuzes in overleg met het team gemaakt worden. Dit zorgt ervoor dat het draagvlak van beslissingen binnen het team hoger is, maar dit is ook een stimulans in het leerproces van de teamleden. Ze leren de achtergrond van keuzes kennen en kunnen dat in de toekomst zelf ook toepassen.

Wanneer het project groter word en er gekozen is om gebruik te maken van meerdere sprints zal in deze meeting ook bepaald worden wat er uitgevoerd gaat worden. Het is dan de taak van de Product Owner en het team om tot de juiste keuze van user stories te komen die opgepakt gaan worden.

Daily scrum

Belangrijkste verbetering bij de uitvoer van de daily scrum die behaald kan worden is het nauwer nemen van de regels. Zo mag een daily scrum maximaal 15 minuten duren. De doelstelling van het gesprek is om de projectvoortgang met het team te bespreken en eventuele problemen aan te kaarten.

Als er problemen zijn dan moeten ze wel in deze scrum besproken worden, maar het zoeken en bespreken van de oplossing is een taak die buiten de vergadering om uitgevoerd moet worden. Het is de verantwoordelijk van de ScrumMaster om zorg te dragen dat er een oplossing voor het aangedragen probleem gevonden gaat worden.

Sprint retrospective meeting

Op dit moment worden de projecten uitgevoerd en opgeleverd. Na afloop van het project gaat een ieder weer verder met zijn of haar nieuwe project en is het klaar. Er is geen moment ingeruimd om te leren van de wijze waarop het project is uitgevoerd.

De organisatie zou veel van uitgevoerde projecten kunnen leren door aan het einde van een sprint een sprint retrospective meeting te houden. Het initiatief van deze meeting komt van de ScrumMaster. Het is de taak van de ScrumMaster om het team continu te verbeteren in het scrum proces en dit past dan ook prima in dat straatje.

In de bijeenkomst kunnen de teamleden hun feedback geven hoe het in de laatste sprint gegaan is met betrekking tot mensen, relaties, processen en tools. De ScrumMaster kan identificeren wat er goed is gegaan en wat er volgens het team verbeterd kan worden.

De ScrumMaster creëert dan ook een plan waarin een plan van aanpak staat om de verbeteringen door te voeren. Het is belangrijk om te realiseren dat het team zelf in de hand heeft hoe het gebruik van Scrum binnen Eleven evolueert. Het team heeft hierin de grootste rol met als aanvoerder de ScrumMaster.

Artefacten

Burndown chart

Door gebruik te maken van de burndown chart krijgen zowel de project manager als de scrum master een beter inzage in de voortgang van het project. De grafiek laat in één oogopslag zien hoeveel werk er in de huidige sprint nog gedaan moet worden en of dat volgens planning is.

Het laat ook gelijk zien hoe goed het team is in het maken van inschattingen. Als het team altijd te veel of te weinig werk op zich neemt dan kan de ScrumMaster deze informatie gebruiken in de komende sprint planning meeting om een betere sprint planning op te stellen met het team.

Overigens kan het ook betekenen dat de velocity van het team in de sprint afgeweken heeft van andere maanden, bijvoorbeeld door dat veel teamleden met vakantie zijn geweest. Een eventuele conclusie kan dus niet gemaakt worden zonder andere factoren mee te nemen in de overweging.

Hulpmiddelen

Velocity

Bepalen hoeveel werk er in een sprint uitgevoerd kan worden is een lastig karwei. Naast het werken aan de projecten word er van iedere ontwikkelaar ook nog eens verwacht dat hij een aantal uren van de dag beschikbaar is voor support werkzaamheden.

De planning van een medewerker bij Eleven word iedere week opnieuw bepaald. Om de week van een ontwikkelaar goed te verdelen zou men bij het plannen gebruik kunnen maken van de metriek velocity. Door de velocity van de afgelopen vijf weken te nemen kan de planner in acht nemen welk percentage van de werkzaamheden besteed kunnen worden aan de uitvoer van de sprint.

Als een ontwikkelaar in de afgelopen vijf weken veel tijd aan support uren kwijt is geweest door welke omstandigheid dan ook, is de kans reëel dat in week 6 ook bovengemiddeld aan support besteed moet worden.

De velocity zal daarom laag zijn en het uren wat besteed kan worden aan de sprint daarmee ook. Het geeft dan een eerlijke verwachting van wat er in die week aan de sprint gedaan kan worden. De ScrumMaster kan daarmee tijdig inspringen op eventuele problemen. Er kan bepaald worden door de organisatie in overleg met de ScrumMaster dat het support aan een andere medewerker wordt gegeven of dat het team uitgebreid word met extra resources.

Door velocity te gebruiken kan er in ieder geval met voortschrijdend inzicht gereageerd worden en zullen de planningen die gemaakt worden dichterbij de waarheid liggen.

Puntenschaal/Story points

Huidige inschattingen die nu gemaakt worden zijn op basis van de eenheid tijd. Scrum verlangt dat de inschattingen gemaakt worden op basis van een abstracte puntenschaal.

Het voordeel met het werken van een relatieve puntenschaal in tegenstelling tot een ureninschatting is dat het inschatten van uren zich richt op het beantwoorden van de vraag: “Hoe lang gaat het duren?”. Over het algemeen een wilde gok op basis van eerdere ervaringen.

Het inschatten van punten richt zich meer op de grootte en complexiteit van een user story. Hierdoor kunnen user stories op gelijkwaardige basis met elkaar vergeleken worden. Een goede schaal om te gebruiken is de Fibonacci reeks.

Belangrijk is nog wel om in samenspraak vast te stellen hoe de relatieve schaal zich relateert aan uren. Het is ook zaak dat er altijd met dezelfde schaal wordt gewerkt zodat vergelijkingen op een gelijkwaardige basis gemaakt worden.

Definition of Done

Door met het team af te spreken wanneer taak nou echt af is, is het mogelijk om de kwaliteit van de opleveringen te verhogen. Het opstellen van een definition of done, wat in feite een korte checklist is, geeft de ontwikkelaar een gestandaardiseerd handvat om zijn gemaakte werk op te leveren volgens de gestelde kwaliteitseisen.

Het hanteren van een definition of done geeft de organisatie een baseline van wat men minimaal kan verwachten bij een opgeleverde applicatie. Daarnaast zorgt het ervoor dat iedereen in de organisatie dezelfde definitie van ‘klaar’ hanteert.

Aangezien Eleven in meerdere disciplines actief is adviseer ik wel om voor iedere discipline een aparte Definition of Done op te stellen. Te weten de disciplines:

- Java projecten
- Webshops
- PHP projecten

Omdat webshops een aparte tak van sport zijn en andere verwachtingen bevatten dan een normaal project adviseer ik om voor deze projecten een aparte definition of done op te stellen. Van webshops word verwacht dat ze 24 uur per dag, 7 dagen per week beschikbaar zijn en dat een klant foutloos zijn order kan doorgeven.

Voor de disciplines Java en PHP is het verstandig om een seperate definition of done op te stellen aangezien de tooling van deze disciplines anders in elkaar steekt. De essentie van de twee controlelijsten zullen overigens wel dicht bij elkaar liggen.

Het is belangrijk dat de definition of done draagkracht heeft binnen het team. Het draagvlak zorgt ervoor het naar behoren gebruikt wordt. Het is dan ook belangrijk dat de definition of done niet opgelegd word, maar in samenspraak met het team tot stand komt. Als het team invloed heeft op de checklist zullen ze zich veel meer achter het gebruik scharen en er betrokken bij zijn om de het werk ook volgens de checklist op te leveren.

User story

Op dit moment is er geen eenduidige stijl in het beschrijven van functionaliteit en eisen. Het is wenselijk als alle functionaliteiten vanuit het zelfde perspectief en in dezelfde stijl zijn beschreven. Daardoor is het eenvoudiger om een user story te lezen, te interpreteren en in te schatten.

Eventuele inschattingen die gedaan worden zullen minder discutabel zijn omdat het duidelijk is vanuit welk licht ze zijn opgesteld. Door te werken in een vaste vorm zal het team snel gewennen aan deze wijze van functionaliteiten beschrijven.

6. Advies implementatie

Het implementeren van deze onderdelen zal geen eenvoudige taak zijn aangezien een aantal onderdelen een kleine cultuuromslag verlangen.

Om te beginnen is het belangrijk dat er in overleg tussen directie en het team besloten wordt welke onderdelen ook daadwerkelijk overgenomen worden. Het is van groot belang dat eventuele koerswijzigingen die gemaakt worden, ook daadwerkelijk gedragen worden door het complete team om de implementatie succesvol te laten verlopen.

Een aantal onderdelen zijn redelijk eenvoudig en op korte termijn te implementeren. Deze onderdelen kunnen ook in de huidige projectmethodiek een prettige ondersteuning zijn. De volgende onderdelen zouden op korte termijn al ingezet kunnen worden.

- Definition of done
    - De DOD kan nu ook al gebruikt worden om te controleren of een taak of mijlpaal naar wens is afgerond.
- Sprint retrospective meeting
    - Door nu al terug te blikken op de uitvoer van projecten kunnen er lessen getrokken worden die bij de implementatie van andere onderdelen kunnen helpen. Welke rol hiertoe in het begin het initiatief neemt is nog niet van belang. Wel moet er een persoon verantwoordelijk worden gesteld voor het oppakken van eventuele verbeterpunten.
- Daily Scrum (stricter)
    - Door de daily scrum stricter uit te voeren zal de scrum met meer plezier en effectiever doorlopen worden.

Een vraagstuk waar ondertussen over nagedacht kan worden is de rolverdeling van Product Owner en ScrumMaster. Van welke functies binnen de organisatie wordt geacht dat zij deze rollen kunnen vervullen tijdens een project.

De ScrumMaster(s) in spe zouden naar een Certified ScrumMaster cursus gestuurd worden of zichzelf autodidactisch opleiden met behulp van de boeken “Do Better Scrum” van Peter Hundermark en “The Scrum Primer” van Jeff Sutherland. Sutherland is één van de coauteurs van de methode Scrum. Voor welke methode er gekozen worden zou bepaald moeten kunnen worden door de persoon zelf. Op deze manier kan hij of zij kiezen voor de leermethode die bij hem of haar aansluit.

Voor de onderdelen velocity en burn-down chart zal het projectvolgsysteem wat gebruikt wordt binnen Eleven aangepast moeten worden. Het huidige volgsysteem, Workmate, kent het begrip van een sprint en een user story niet. Het is gebaseerd op mijlpalen en taken en richt zich daarmee wat meer op de traditionele projectmethodieken.

Aangezien Workmate ook gebruikt wordt voor administratieve doeleinden zou het eventueel vervangen van het systeem betekenen dat er meerdere systeem en processen aangepast moeten worden.

7. Literatuurlijst

http://nl.wikipedia.org/wiki/Scrum_(softwareontwikkelmethode)
http://scrummethodology.com/scrum-effort-estimation-and-story-points/
http://pm.stackexchange.com/questions/2765/agile-why-use-points-instead-of-hours
http://en.wikipedia.org/wiki/Scrum_(development)
http://www.scrum.org/storage/scrumguides/Scrum%20Guide%20-%20NL.pdf#view=fit
http://en.wikipedia.org/wiki/Burn_down_chart
http://www.scrumalliance.org/articles/106-definition-of-done-a-reference
http://www.scrumalliance.org/articles/107-how-do-we-know-when-we-are-done
http://scrummethodology.com/what%E2%80%99s-the-point-of-velocity/
http://www.infoq.com/news/2009/07/velocity-value

[BOEK] Software Development Using Scrum – Mike Cohn
http://www.amazon.com/gp/product/B002TIOYWQ/ref=kinw_myk_ro_title