BACHELORSCRIPTIE

JASPER JANSEN

CONFIGUREERBAAR MAKEN VAN ACCORDERINGS PROCESSEN

J. Jansen, 13092197 i.o.v. HHS Informatica Den Haag, 2 juni 2017

REFERAAT

Ontwerpen en ontwikkelen van een webapplicatie in Salesforce waarmee accorderings-processen configureerbaar worden gemaakt.

Rijswijk, Nétive, 2017

Afstudeerverslag geschreven door Jasper Jansen, voor het afstuderen bij de opleiding Informatica aan de faculteit IT & Design aan de Haagsche Hogeschool.

Samenvatting

In dit verslag wordt het ontwikkelproces beschreven uitgevoerd door Jasper Jansen bij Nétive VMS BV in de periode februari – juni 2017. Het doel van dit project was om een workflow-configuratietool op te zetten en te implementeren in een Salesforce-omgeving. Om dit te bereiken zijn requirements opgesteld, is software-selectie uitgevoerd, en heeft een ontwikkeltraject gefaseerd met het Scrum-framework plaatsgevonden.

Descriptoren

Afstudeeropdracht
Requirements
Softwareselectie
Webapplicatie
Scrum
Salesforce
Force.com
HTML
CSS
Javascript
JQuery
Backbone
Lodash
JointJS
Visualforce
Apex
SVG

COLOFON

ONDERWIJSINSTELLING

Onderwijsinstelling	De Haagsche Hogeschool Faculteit IT & Design Opleiding Informatica
Adres	Johanna Westerdijkplein 75 2521 EN Den Haag
Telefoon	070 - 445 8888
Begeleidend Examinator	Dhr. O. Zor
Tweede Examinator	Mw. A.M.J.J. Lousberg – Orbons

BEDRIJF

Bedrijf	Nétive VMS B.V. Afdeling Software Engineering
Adres	Patrijsweg 20 2289 EX Rijswijk
Telefoon	015 - 251 1840
Opdrachtgever	Dhr. M. van der Wal mike.van.der.wal@netive.nl
Bedrijfsmentor	Dhr. S. Kuiper sven.kuiper@netive.nl

STUDENT

Naam	Dhr. J. Jansen j.jansen-4@student.hhs.nl jasper.jansen@netive.nl
Studentnummer	13092197
Opleiding	IT & Design: Informatica

VOORWOORD

De afgelopen vier maanden heb ik in Rijswijk een stage gevolgd bij het bedrijf Nétive. Ik heb hier een opdracht gevolgd die goed aansloot bij mijn interesses als programmeur: het helpen van andere mensen door hun leven makkelijker te maken. Dit is natuurlijk niet alleen een kwestie van een mooi opgemaakte webpagina met een paar leuke JQuery-functies; het is ook leren begrijpen wat de gebruiker wil bereiken met de software en inzicht krijgen in de denkwijze van mensen die wellicht niet technisch geleerd zijn.

Ik wil daarom ten eerste Mike van der Wal, Product Development Manager van Nétive, bedanken dat hij mij de kans heeft gegeven deze opdracht uit te voeren. Afstuderen mag dan erg spannend zijn, maar het helpt heel erg als je een project hebt waar je gemotiveerd bent je best voor te doen.

Uiteraard wil ik ook Sven Kuiper, mijn bedrijfsmentor en Senior Developer van Nétive, bedanken voor alle hulp die hij mij geboden heeft. Hij was altijd bereid mij te helpen en het was leuk te zien dat hij enthousiast was over mijn werk. Dat motiveerde mij ook om mijn best te doen.

Tot slot gaat er ook een bedankje uit naar de begeleiders van de Haagsche Hogeschool, die mij met hun kritische vragen geholpen hebben goed mijn werk te evalueren.

Jasper Jansen, Juni 2017

Inhoudsopgave

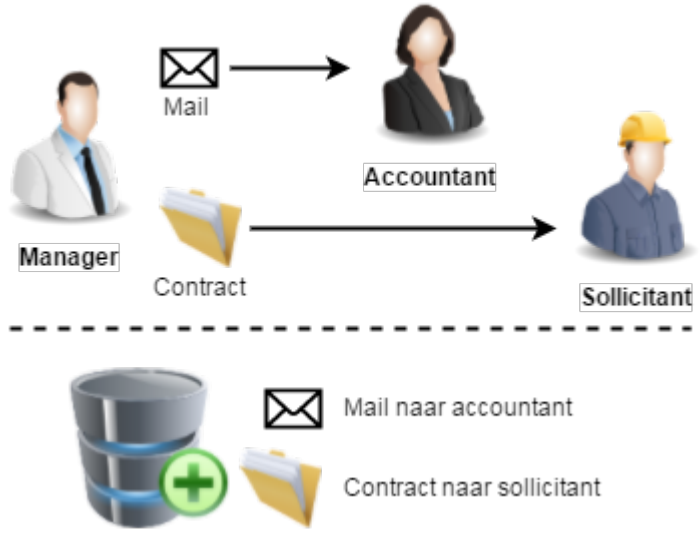
Lijst van termen en afkortingen

Cloud	Het woord Cloud duidt op informatie, software of een service dat deels of volledig via het internet toegankelijk is.
Configuratietool	Software waarin informatie kan worden geconfigureerd, beheerd en gedistribueerd.
Force.com	Een van de producten van Salesforce. Een Platform as a Service (PAAS) applicatie waarop bedrijven een gepersonaliseerde omgeving kunnen inrichten en een database kunnen opzetten. Met name ontworpen voor bedrijfsadministratie (https://www.force.com).
HRM	Human Resource Management, het algemene proces van het beheren en administreren van het personeelsbeleid binnen een organisatie.
PaaS	Platform as a Service, een software distributie model waarin een platform in de Cloud wordt opgeleverd, waar gebruikers controle hebben over de software die erop wordt gedraaid, zonder zich bezig te hoeven houden met de infrastructuur daarachter.
SaaS	Software as a Service, een software distributie model waarin een of meer applicaties via een Cloud-platform wordt opgeleverd, waar de gebruikers te allen tijde toegang toe hebben, zonder zich bezig te hoeven houden met de ontwikkeling daarvan.
Salesforce	Een software-aanbieder met als primaire doel bedrijven te ondersteunen bij het beheren van hun bedrijfsgegevens (https://www.salesforce.com).
Visualforce	Een uitbreiding op de bekende Hypertext Markup Language (HTML). Deze maakt het mogelijk om informatie uit de Force.com database in te vullen in de webpagina.
VMS	Vendor Management Systeem, software waarin men de administratie rondom de inhuur en beheer van werknemers kan beheren.
Workflow	Een systematische uitwerking van een bedrijfsproces, gebruikt binnen het Nétive VMS om een bepaalde applicatiestructuur af te dwingen.

1. Inleiding

De laatste jaren zijn er op het gebied van bedrijfsautomatisering veel ontwikkelingen geweest. Meer en meer werk wordt uit de handen van mensen genomen, maar niet altijd ten koste van de mens. Denk bijvoorbeeld aan processen waarbij keuzes en evaluaties moeten worden gemaakt op basis van meer dan alleen cijfers en rekensommen, zoals het werven en aannemen van medewerkers. Dit zijn zeer op de mens gerichte processen die niet alleen door een machine kan worden gedaan.

Voor dit soort situaties is ook software beschikbaar, die bedrijven hierin kan begeleiden. In Figuur 1 zien wij een situatie waarin een manager een sollicitant een contract op wilt sturen. Hij wil ook dat de accountant hiervan op de hoogte wordt gesteld. In deze voorbeeldsituatie hoeft de manager enkel naar het dossier van de sollicitant in de software te gaan en op een knop te drukken. Dan stuurt het systeem automatisch het reeds opgestelde contract naar de sollicitant en krijgt de accountant hier een mailtje van.



Figuur 1: Bedrijfsproces ondersteund door software

Maar hoe weet het systeem wie welke gegevens moet ontvangen? Kan een medewerker van een andere afdeling ook deze actie ondernemen? Wie moet nu de volgende stap ondernemen in dit proces? Deze informatie moet worden vastgelegd in de software, maar het probleem is dat de persoon die deze kennis heeft, meestal geen technische kennis heeft van deze software.

Dit is het probleem waar ik mij dit afstudeerproject mee bezig heb gehouden: hoe kan ik zorgen dat een persoon die geen kennis heeft van programmeren software-processen kan inrichten die volledig in overeenstemming zijn met bestaande bedrijfsprocessen, zonder dat het inrichten te complex en ingewikkeld wordt?

Om dit probleem op te lossen heb ik een applicatie ontwikkelt, waarin gebruikers bedrijfsprocessen flowcharts kunnen tekenen in de vorm van. Achter de verschillende onderdelen van de flowchart kunnen zij informatie hangen die relevant is voor het bedrijfsproces, zoals e-mails die moeten worden verstuurd of beperkingen in wie daarbij betrokken mogen zijn.

In dit verslag zal worden beschreven hoe deze applicatie, genaamd de workflow-configuratietool, tot stand is gekomen.
Het verslag is in de volgende vier fasen verdeeld:

1. Analyse van het bedrijf, het probleem en de gewenste oplossing.
2. Voorbereiding op het ontwikkeltraject met behulp van planning, requirements en software-selectie.
3. Ontwikkelen van de workflow-configuratietool met behulp van Scrum.
4. Evaluatie van de uitgevoerde werkzaamheden.

2. Achtergrond

Inleiding
In dit hoofdstuk zal de context van de afstudeeropdracht worden beschreven. Hierbij zal het bedrijf aan bod komen en zal de opdracht zelf worden uitgewerkt. In de opdrachtbeschrijving wordt uitgelegd wat de aanleiding was voor de opdracht, wat de gewenste resultaten waren, en hoe de opdracht zich verhoudt tot andere projecten binnen Nétive. Er zal ook een beschrijving worden gegeven van de werking en ontwikkeling van workflows in het huidige systeem.

2.1 Bedrijf

2.1.1 Nétive VMS BV

Nétive is een bedrijf, gevestigd in Rijswijk en opgericht in 2003, dat software ontwikkelt voor de arbeidsmarkt. Zij telt ongeveer 25 medewerkers, met afdelingen voor development, support, consultancy en management.
Deze software ondersteunt bedrijven bij interne processen omtrent het werven, selecteren, contracteren en beheren van vaste en tijdelijke medewerkers en wordt al door meer dan 140 bedrijven gebruikt. De software zelf wordt voor elk bedrijf op maat gemaakt en gedistribueerd als webapplicatie via het Salesforce Cloud-platform.
Nétive is uniek in hoe zij het Salesforce platform gebruikt. Salesforce levert software volgens het Platform-as-a-Service (PaaS) model, waarbij bedrijven zelf de applicatie moeten inrichten en configureren. Nétive is echter een tussenpersoon tussen de klant en Salesforce, en regelt zelf de licenties, configuratie en programmatuur. De klant krijgt dan een op maat gemaakte Salesforce omgeving opgeleverd volgens het Software-as-a-Service (SaaS) model, die hij direct kan gebruiken.

2.1.2 Klanten

De doelgroep voor de software van Nétive zijn bedrijven die zich bezighouden met het contracteren en beheren van medewerkers binnen een organisatie. Deze bedrijven kunnen zowel organisaties zijn die zelfstandig hun werknemers beheren, of bedrijven die deze service aan andere organisaties bieden, zoals uitzendbureaus of detacheringsbedrijven. Een paar voorbeelden van bestaande klanten van Nétive zijn de Rijksoverheid, NS, KLM en Randstad.

2.1.3 Vendor Management Systeem

De software die Nétive ontwikkelt heet het Vendor Management Systeem (VMS). Met deze software kunnen bedrijven hun processen omtrent flexibele arbeid beheren. Elke klant van Nétive heeft een op maat gemaakte distributie van het VMS, aangepast op de bedrijfsprocessen, informatie en afdelingen die in dat bedrijf aanwezig zijn. De software zelf is een SaaS-pakket, gehost op het Salesforce Cloud-platform.

nétive
"We create secure workforce agility."

Figuur 2: Logo Nétive VMS BV

Force 2 Cloud Edition
nétive VMS

Figuur 3: Logo VMS product

2.1.4 Werkomgeving

Ik heb de afstudeeropdracht uitgevoerd als software engineer in het development team van Nétive. Op de bedrijfsvloer zat ik in dezelfde ruimte als mijn bedrijfsmentor, samen met het Scrum team waar hij Scrum Master van was.
Van Nétive heb ik een Macbook in bruikleen gekregen met bijbehorende randapparatuur (zie figuur 3) waarop ik mijn programmeerwerkzaamheden kon uitvoeren. Deze mocht ik mee naar huis nemen als ik de behoefte had om thuis verder te werken.
Nétive maakt voor hun interne informatievoorziening veel gebruik van de software van Google. Ik heb dan ook van Nétive een Google-account gekregen, waarmee ik toegang had tot de centrale opslag, kalender en e-mail van Nétive.
Voor de uitvoering van de opdracht heb ik een test-omgeving opgezet in Salesforce en heb ik zogeheten Salesforce Trailheads gevolgd om vaardig te worden in het gebruik van Force.com en de programmeertalen die daarin gebruikt worden. In deze omgeving heb ik de workflow-configuratietool opgezet en beschikbaar gemaakt voor de andere werknemers.

Figuur 4: Foto van de werktafel

2.2 Workflows

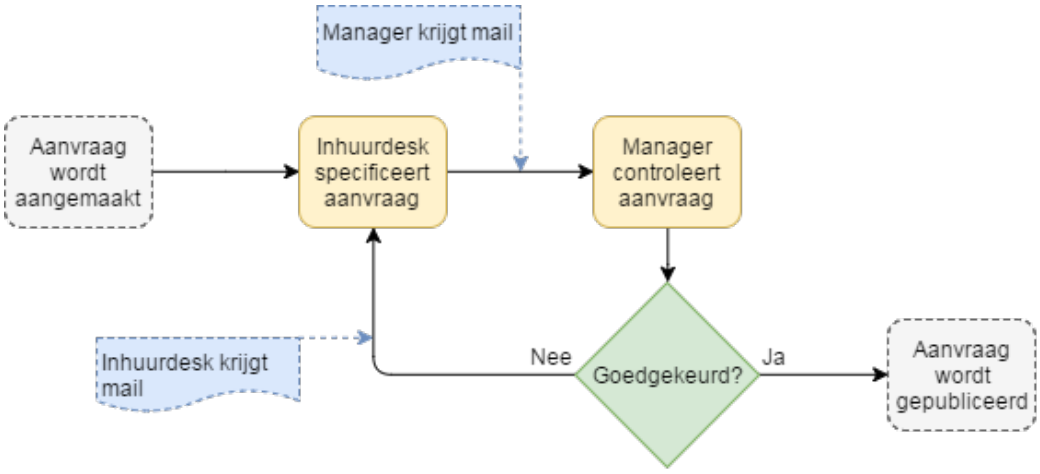
Om de context van de opdracht goed te begrijpen, is het verstandig kennis te nemen van de werking van workflows en de manier waarop deze in de oude situatie werden ontwikkeld. Deze informatie is bekend geworden in de inwerkperiode van het project, door in overleg te gaan met de bedrijfsmentor. Ik heb daarvoor verschillende voorbeelden gezien van workflow-bestanden en uitleg gekregen over de toepassing en ontwikkeling daarvan.

2.2.1 Functie

De functie van workflows is om vast te leggen hoe bedrijfsprocessen moeten worden doorlopen, wie dit kan doen, en wat er daarvoor moet gebeuren. Voor elke eindklant van Nétive zijn verschillende workflows ontwikkeld, die elk een ander bedrijfsproces vastleggen dat relevant is voor het VMS.
Wat een workflow doet in het VMS is bepalen welke informatie beschikbaar is voor de gebruiker. De workflows zelf zijn simpelweg een computerbestand, verborgen in de backend van het systeem. Het enige wat de gebruiker in het VMS te zien krijgt, zijn de knoppen en invoervelden waarvan de workflow aangeeft dat deze getoond mogen worden.
De gebruiker gebruikt deze knoppen en invoervelden en stuurt daarmee het bedrijfsproces volgens de banen die in de workflow zijn vastgelegd. Het is hierbij mogelijk dat in de workflow bepaalde processtappen zijn afgezonderd, op basis van bepaalde rechten en condities die in de workflow zijn aangegeven. Op deze manier wordt gezorgd dat de juiste mensen de juiste stappen kunnen zetten in het bedrijfsproces en dat het proces voldoet aan de regels van het bedrijf.

Voorbeeld

Om te demonstreren hoe een workflow achter de schermen wordt gebruikt in het VMS, zal een versimpeld voorbeeld worden gegeven van een bedrijfsproces voor het aanmaken en publiceren van een nieuwe werknemersaanvraag. Dit proces is schematisch afgebeeld in figuur 4.



Figuur 5: Bedrijfsproces aanmaken van “Aanvraag”

Het proces begint wanneer een medewerker, hier van de inhuurdesk, in de VMS-omgeving van zijn bedrijf op de knop klikt om een nieuwe aanvraag aan te maken. Het systeem maakt dan een nieuw object van het type “aanvraag” aan in de database, waarin informatie kan worden opgeslagen relevant voor het aanvraagproces.
Daarnaast wordt ook de workflow voor het aanvraagproces uit het systeem ingeladen en slaat het systeem op dat de workflow in de eerste stap van het proces zit. De workflow bevindt zich dan in de status ‘concept,’ wat in dit geval betekent dat de aanvraag bij de inhuurdesk ligt.

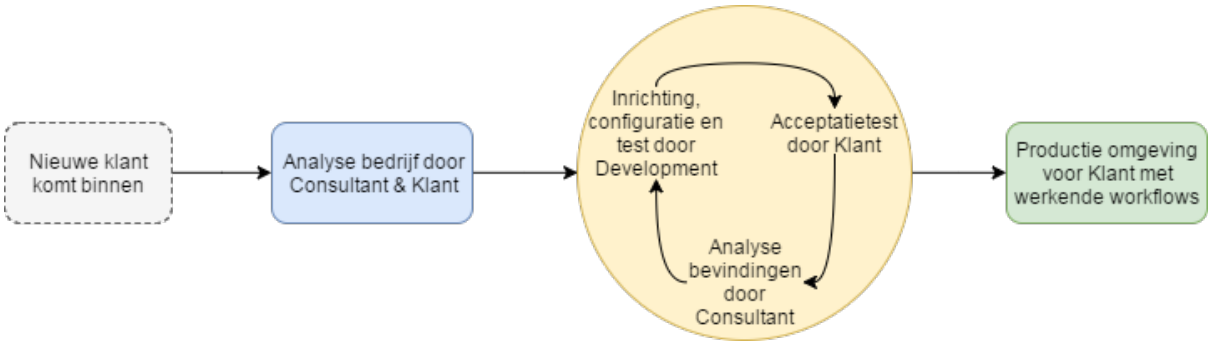
Op het scherm in de VMS-applicatie krijgt de inhuurdeskmedewerker een aantal invoervelden te zien waarin informatie kan worden ingevuld over de aanvraag, zoals een naam, functietitel en kosten. Deze worden ingevuld en opgeslagen in het eerder aangemaakte aanvraag-object.
Vervolgens heeft de medewerker de optie om op een aantal knoppen te klikken. De naamgeving en functie van deze knoppen wordt afgeleid uit het workflow-bestand. De medewerker kan hiermee de aanvraag tussentijds opslaan of doorsturen naar de volgende stap in het proces.
Als de medewerker de aanvraag wilt doorsturen, klikt hij in dit geval op de knop "Verstuur naar manager". Hierdoor vindt een overgang in de workflow plaats (aangegeven in figuur 4 met "overgang A"), specifiek van de status "concept" naar de status "verstuurd naar manager."
Bij deze overgang kunnen ook bepaalde geautomatiseerde acties plaatsvinden, zoals gedefinieerd in de workflow. In dit voorbeeld krijgt de manager een mailtje binnen, waarin staat dat er een aanvraag is gemaakt die door hem moet worden gecontroleerd.
Als de overgang van de status "concept" naar "verstuurd naar manager" eenmaal heeft plaatsgevonden, mag de inhuurdeskmedewerker de aanvraag niet meer wijzigen. Dit komt doordat het systeem in de workflow kan aflezen wie lees- en schrijfrechten heeft binnen een bepaalde status, en de inhuurdesk staat daar in dit geval niet tussen.
De aanvraag ligt nu bij de manager die twee opties heeft: 1) Hij kan de aanvraag afkeuren en terugsturen naar de inhuurdesk. 2) Hij kan de aanvraag accepteren en publiceren op hun website. Het systeem detecteert deze opties door wederom in het workflow-bestand te kijken welke overgangen in de status "verstuurd naar manager" zijn gedefinieerd en door te bepalen of de gebruiker de rechten heeft om deze overgangen te initiëren.
Als de manager de aanvraag afkeurt, kan hij hier een opmerking bij geven en ontvangt de inhuurdesk een mailtje dat het is afgekeurd in combinatie met het commentaar van de manager (overgang B). Wordt de aanvraag goedgekeurd, wordt deze naar een extern systeem gestuurd en automatisch op de website van het bedrijf geplaatst (overgang C).

Conclusie

Alle bovengenoemde statussen, overgangen, acties, rechten en opties worden vastgelegd in het workflow-bestand voor dit bedrijfsproces. Dit bedrijfsproces ging over het aanmaken van een nieuwe aanvraag, maar er zijn voor elk bedrijf nog verschillende andere processen vast te leggen in workflows, zoals die voor contractmanagement, urenregistratie en facturatie.
Ook is bovenstaande workflow zeer vereenvoudigd. Een normale workflow heeft vaak al snel tien tot twintig statussen waartussen overgangen plaats kunnen vinden, elk weer met zijn eigen rechten, acties, en gebruikers die er bij zijn betrokken.

2.2.2 Creatie

Alle informatie die in een workflow-bestand moet worden opgenomen, moet voldoen aan de specificaties van het bijbehorende bedrijfsproces. Het aanmaken van workflows is daarom een langdurig proces dat met uiterste zorgvuldigheid moet worden uitgevoerd. In dit hoofdstuk zal dit proces worden uitgewerkt zoals dit gebeurde voor de aanvang van het project. Dit proces is schematisch weergegeven in Figuur 6.



Figuur 6: Huidige configuratietraject nieuwe klant

Als Nétive een nieuwe klant aanneemt, worden alle door de VMS ondersteunde bedrijfsprocessen van de klant op papier vastgelegd. Een consultant analyseert het betreffende bedrijf en deelt deze informatie met Nétive. Aan de hand van deze specificaties schrijft ze documentatie en maakt ze Visio-diagrammen waarin wordt beschreven hoe de bedrijfsprocessen van de klant plaatsvinden.
Vervolgens gaan de ontwikkelaars van Nétive aan de slag om de bedrijfsprocessen tot in detail te realiseren. Een deel van dit proces bestaat uit het vastleggen van deze processen in een workflow-bestand. Zij schrijven de workflows met de hand in XML-formaat vergelijkbaar met die in Figuur 7. In dit figuur, afgeleid uit de workflow van hoofdstuk 2.2.1, is een status te zien, gemarkeerd met geel, met twee overgangen, gemarkeerd met groen. Deze overgangen zijn respectievelijk overgang B en C uit Figuur 5.

```
<workflow>
	<status naam="verstuurd_naar_manager">
		<leesrechten>Medewerker.Manager</leesrechten>
		<schrijfrechten>Medewerker.Manager</schrijfrechten>

		<overgang naam="afkeuren" naar="verstuurd_naar_inhuurdesk">
			<overgangsrechten>Medewerker.Manager</overgangsrechten>
			<invoeroptie naam=”commentaar”/>
			<mail naar="Medewerker.Inhuurdesk" met=”commentaar”>
				Aanvraag is afgekeurd.
			</mail>
		</overgang>

		<overgang naam="goedkeuren" naar="aanvraag_gepubliceerd">
			<overgangsrechten>Medewerker.Manager</overgangsrechten>
			<mail naar="Medewerker.Inhuurdesk">
				Aanvraag is goedgekeurd.
			</mail>
		</overgang>
	</status>
</workflow>
```

Figuur 7: XML van een Workflow (vereenvoudigd)

Het resulterende XML-bestand wordt geïmporteerd in een Force.com testomgeving en ondervindt daar een aantal functionele tests. Vervolgens wordt aan de klant gevraagd de testomgeving te gebruiken en aan te geven of het acceptabel is of niet. Als de klant tevreden is, wordt de testomgeving overgezet naar een productieomgeving en kan de klant gebruik maken van het VMS. Zo niet, analyseert de consultant wat er fout is en gaan de ontwikkelaars verder met het configureren van de testomgeving tot deze wel acceptabel is.
Vaak neemt de klant later weer contact op met Nétive omdat bepaalde onderdelen van het systeem niet kloppen, of krijgt men in de loop van de ontwikkeling hernieuwd inzicht in de bedrijfsprocessen. Dan moet een ontwikkelaar wederom het workflow-bestand induiken om de fouten te verbeteren, waarna het moet worden getest voor het weer kan worden gedistribueerd.

2.3 Opdracht

2.3.1 Aanleiding

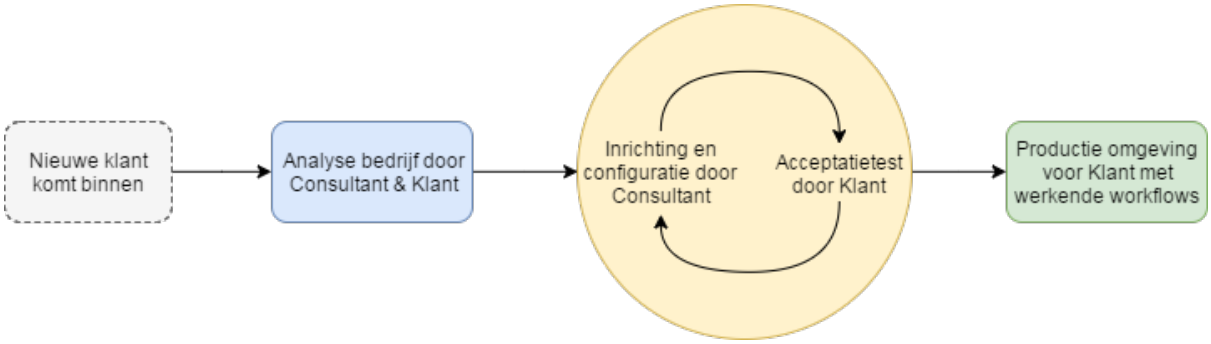
Het Vendor Management Systeem (VMS) is, zoals eerder beschreven, het software-product van Nétive. De workflows die hierin zijn opgenomen, leggen de basis voor alle bedrijfsprocessen die in het VMS worden doorlopen. In de workflows worden alle rechten, regels en taken vastgelegd die hiervoor van belang zijn, zodat gebruikers deze processen volgens hun bedrijfsstandaard kunnen doorlopen.
De exacte opzet van deze workflows is uniek voor elk bedrijf en daarom moest een configuratietraject worden afgelegd voordat een klant van zijn VMS-omgeving gebruik kon maken. Bij dit traject waren ook een consultant en meerdere ontwikkelaars van Nétive betrokken. Het management van Nétive had geconstateerd dat bijna de helft van de gewerkte uren van ontwikkelaars werd gespendeerd aan het configureren van het VMS en vond dat dit moest worden teruggedrongen.
De oplossing die hiervoor was gekozen, was om software te ontwikkelen waarmee de consultants zelf een VMS-omgeving konden configureren, zodat de tussenstap naar de ontwikkelaars kon worden voorkomen. Deze software zou bekend komen te staan als de Self Service Desk (SSD) en zou deels bestaan uit een tool waarmee consultants zelf workflows konden samenstellen. De resterende configuratie van de VMS was, gezien de omvang daarvan, buiten de scope van dit afstudeerproject gezet en werd in een later stadium door de ontwikkelaars van Nétive opgepakt.

2.3.2 Doelstelling

Het doel van dit afstudeerproject was om software te ontwikkelen waarin consultants een workflow konden vastleggen en configureren. Met het bestaan van deze software moest de tussenkomst van een programmeur bij het configureren van workflows zo veel mogelijk worden geëlimineerd.
De software zou bekend komen te staan als de workflow-configuratietool. Deze tool moest de consultants van Nétive de mogelijkheid geven om zelfstandig workflows op te stellen in de Self Service Desk zodat deze in VMS-omgevingen konden worden geïmporteerd. Het grootste belang bij deze ontwikkeling was dat het alle betrokken partijen een significant aantal werkuren moest besparen en tegelijkertijd de consultants meer controle zou geven.
Het was gewenst om de consultants een visuele representatie van de workflows te geven in de configuratietool. Zo zouden zij eenvoudig kunnen achterhalen welke statussen er in de workflow beschikbaar waren en hoe deze met elkaar waren verbonden. Voor het toevoegen van een workflow-onderdeel zou enkel een nieuw object hoeven worden getekend, waarna de gebruiker deze met het bestaande diagram kan verbinden.
Het was de bedoeling dat consultants de workflows konden opslaan in de SSD zonder deze naar een VMS-omgeving te moeten uploaden. Zo kon alle informatie bewaard blijven en kon de consultant later verder werken wanneer nodig. Pas als een workflow moest worden geëxporteerd, moest deze naar XML-formaat worden geconverteerd omdat dit de standaard is die het VMS gebruikt.
De consultants die de workflow-configuratietool zouden gaan gebruiken, beschikten niet over technische kennis van de werking van workflows. Hier moest rekening mee worden gehouden in de ontwikkeling van de configuratietool door duidelijke besturingselementen in de tool weer te geven en te voorkomen dat een foute workflow konden opstellen. Ook moest worden gezorgd dat de tool gebruiksvriendelijk bleef, ondanks de hoeveelheid informatie die de consultant in de tool zou moeten invoeren.

2.3.3 Resultaat

Met het bestaan van de workflow-configuratietool zou de oplevertijd van nieuwe workflows en updates voor workflows significant worden ingekort. Ook worden de ontwikkelaars hiermee uit het proces gehaald (zie Figuur 8).
Als Nétive een nieuwe klant binnenhaalt, kan een consultant vrijwel direct beginnen met het maken van workflows in de configuratietool. Zij hoeven dan geen tekeningen te maken in Visio en programmeurs zijn niet dagen bezig een XML-bestand op te zetten. Tevens kunnen de workflow-diagrammen dan als documentatie dienen voor de klant.
De persoon die de workflows gaat configureren, kan de informatie in de Self Service Desk opslaan om daar later mee verder te werken. Als de workflow klaar is en is getest, kan deze vanuit de SSD worden gedistribueerd naar de productieomgeving van de klant. Wanneer later nog wijzigingen nodig zijn in de workflow-configuratie, kan de consultant deze direct via de configuratietool aanbrengen en distribueren.



Figuur 8: Nieuw configuratietraject nieuwe klant

2.3.4 Afbakening

Dit project had als doel om workflows in het Nétive VMS configureerbaar te maken voor de consultants. Dit project was daarbij deel van een groter project binnen Nétive genaamd de Self Service Desk, die ook nog in de beginfase zat toen het afstudeerproject werd opgestart. Wanneer de gehele SSD klaar was, zou de workflow-configuratietool echter pas in gebruik worden genomen. Het zal dan eerst alleen door de interne consultants worden gebruikt en later ook door consultants van externe bedrijven waar Nétive mee werkt.
Omdat de workflow-configuratietool binnen de afstudeerperiode nog niet naar door de consultants zou worden gebruikt, was de functionaliteit voor distributie en versiebeheer minder van belang. Deze onderdelen van de tool hebben daarom een lagere prioriteit gekregen in de Sprint planning. De focus is voornamelijk blijven liggen op het kunnen tekenen en configureren van workflows, en het opslaan van de configuratie in een database.
Tot slot moest rekening worden gehouden met de aanwezigheid van verschillende typen workflows in het VMS. Elke type workflow heeft dezelfde functie, maar heeft vaak net een andere informatiestructuur.
Het kunnen ondersteunen van alle workflow-types was minder belangrijk dan het ontwikkelen van de configuratietool zelf en daarom hebben wij gekozen om één type workflow, de “aanvraag,” als uitgangspunt te gebruiken. Workflows van dit type worden namelijk het meest gebruikt in het VMS en bevatten vrijwel alle informatie die in een workflow kan worden opgenomen. Hierdoor kon de focus blijven liggen op de functionele aspecten van de workflow-configuratietool en kon worden voorkomen dat te veel tijd zou worden besteed aan het faciliteren van alle mogelijke informatiestructuren.

3. Aanpak

Inleiding
In dit hoofdstuk zal worden uitgelegd hoe de gekozen aanpak voor het project in de eerste week is gekozen. Er zal worden verteld hoe het plan van aanpak tot stand is gekomen en waarom er voor deze aanpak is gekozen in de context van het project. Ook de gekozen software-ontwikkelmethode zal hierbij aan bod komen.

3.1 Taken

Voordat een plan van aanpak kon worden opgesteld, moest ik achterhalen welke taken moesten worden uitgevoerd om het project te voltooien. Uit de opdrachtbeschrijving die Nétive mij had gegeven waren de volgende taken al duidelijk:

- Er moest een workflow-configuratietool worden ontworpen en ontwikkeld in een Force.com omgeving.
- De tool moest in de vorm van een webapplicatie worden opgezet en een grafische weergave van de workflows kunnen bieden.
- De workflows opgesteld in de tool moesten met versiebeheer worden beheerd en naar test- en productieomgevingen kunnen worden gedistribueerd.
- Het was aan mij om te bepalen en te verantwoorden hoe en met welke hulpmiddelen ik deze tool in deze omgeving zou gaan realiseren.

Uit deze informatie kon ik nog niet opmaken wat een workflow precies was of wat de Force.com omgeving inhield. Daardoor was het ook lastig te bepalen hoe de workflow-configuratietool moest worden opgezet.
Om deze informatie te achterhalen ben ik in overleg gegaan met mijn bedrijfsmentor en de Product Owner van Nétive, waarin ik vragen heb gesteld over de inhoud van de opdracht, de werking van het bestaande systeem en de wensen voor het nieuwe systeem.
Uit deze gesprekken werd duidelijk wat de functie van workflows was in het systeem en waaruit deze is opgemaakt. Ook werd duidelijk dat de gebruiksvriendelijkheid van de configuratietool een belangrijk kwaliteitscriterium was, gezien de doelgroep van de tool. Voor een gedetailleerde opdrachtbeschrijving, zie hoofdstuk 2.3 "Opdracht".
Met deze kennis over de opdracht heb ik kunnen bepalen welke taken ik zou moeten vervullen om het project te kunnen voltooien. Deze taken zijn hieronder beschreven.

3.1.1 Requirements

De eerste taak die ik zou moeten uitvoeren, was het eliciteren, vastleggen en reviewen van een lijst met requirements. Met deze requirements zou ik de mogelijkheid hebben om nauwkeurig en correct de specificaties van het gewenste product vast te stellen. Verschillende details over de structuur van workflows en de gewenste werking van de applicatie waren bij de opstart van het project nog niet duidelijk vastgelegd en daarom was het belangrijk om deze informatie te achterhalen.
De requirements moesten een houvast zijn voor de verdere voortgang van het project en zorgden ervoor dat de andere taken daarbinnen met zekerheid konden worden uitgevoerd. Zonder requirements zou het lastig zijn om overzicht te houden over de eisen die aan de workflow-configuratietool zijn gesteld en is de kans groot dat het uiteindelijke product niet acceptabel is.
Er was besloten een lijst met requirements op te zetten en bij te houden, achterhaald uit verschillende interviews met de stakeholders. Deze requirements zouden op verschillende categorieën worden ingedeeld, waaronder Business-, User- en Functionele requirements, en er moest herleidbaar zijn welke requirements uit elkaar zijn voortgekomen. Verder moesten de requirements ook worden gereviewd door de stakeholders, zodat er zekerheid kon ontstaan over de juistheid daarvan.

Op basis van de requirements konden ook ontwerpen worden gemaakt van de workflow-configuratietool. Het doel hiervan was om een duidelijker beeld te vormen van de functionaliteit van de tool, zodat deze sneller en met minder fouten kon worden ontwikkeld.
Er was gepland om een klassendiagram te maken van alle objecten en informatie die zich in de configuratietool bevonden en deze volgens met UML te tekenen.

3.1.2 Onderzoek

Het was reeds bekend dat de workflow-configuratietool in een Force.com omgeving moest kunnen draaien. Dit is een Platform-as-a-Service (PAAS) ontwikkelomgeving, gemaakt door Salesforce, waarin ontwikkelaars zelf applicaties samen kunnen stellen voor implementatie in een Salesforce omgeving.
Wat echter nog aan mij was om te bepalen, was hoe ik op de webpagina een module kon maken waarin grafisch objecten konden worden weergegeven en bewerkt. Deze functionaliteit zit niet kant-en-klaar in HTML, Javascript of Salesforce ingebouwd maar het was ook niet wenselijk dit van de grond af op te bouwen. Dit zou veel tijd in beslag nemen en daardoor minder tijd overlaten voor de andere requirements.
In de planning van het project is daarom besloten om een onderzoek op te starten als taak binnen het project. In dit onderzoek zou ik gaan zoeken naar software waarmee in een browser diagrammen konden worden getekend waarin de elementen uit de workflow kunnen worden ondergebracht. Uit deze verzameling tekentools zou ik vervolgens één tool selecteren die het meest geschikt is voor de specificaties van dit project.
Een deel van het onderzoek zou bestaan aan het opstellen van selectiecriteria. Als de tekentools goed op basis van deze criteria werden beoordeeld, was de kans kleiner dat een ongeschikte tekentool werd gekozen.
Als later in het ontwikkelingstraject zou blijken dat de gekozen tekentool niet aan de opgestelde criteria voldeed, zou een andere tool moeten worden gekozen en had de ontwikkeling weer deels opnieuw moeten beginnen. Daarom was het efficiënter om eerst een goede tekentool te kiezen middels een onderzoek, voor daadwerkelijk aan de programmeerwerkzaamheden werd begonnen. Als hierin zou worden geconcludeerd dat er geen geschikte tekentool beschikbaar was, konden wij ook ruim op tijd de scope van het project hierop aanpassen.
Deze taak zou worden afgesloten met een onderzoeksrapport waarin verschillende diagram-tekentools tegen elkaar worden afgewogen en waarin de keuze van de geselecteerde tekentool wordt onderbouwd.

3.1.3 Programmeren

De belangrijkste taak binnen dit project, het programmeren van de workflow-configuratietool, zou ook de meeste tijd in beslag nemen. Hiervoor moest een programmeeromgeving worden ingericht op het Force.com platform waarin ik de ontwikkelwerkzaamheden kon uitvoeren. De ontwikkelingen moesten worden uitgevoerd aan de hand van de opgestelde requirements en er moest gebruik worden gemaakt van de diagram-tekentool gekozen in het onderzoek. Het ontwikkeltraject moest ook met behulp van een bepaalde software-ontwikkelmethode worden gefaseerd. Hiervoor is het Scrum-framework gekozen. Deze keuze wordt verder uitgewerkt in hoofdstuk 3.3.
Er moest voornamelijk in HTML, CSS en Javascript worden geprogrammeerd, aangezien dit de talen zijn waarmee de frontend van applicaties op het Force.com platform worden opgebouwd. Daarnaast gebruikt Salesforce ook hun eigen programmeertalen waarmee onder andere een verbinding kan worden gelegd tussen de applicatie en de Salesforce database. Dit zijn Visualforce voor de frontend en Apex voor de backend.
Voor de ontwikkeling van een Force.com applicatie is gebruik gemaakt van de Force.com IDE plug-in voor Eclipse. Hiermee konden de bestanden op het Cloud-platform worden geïmporteerd naar een lokale ontwikkelomgeving en daar worden aangepast en geüpload. Ook bood de plug-in hulpmiddelen voor het gebruik van Visualforce en Apex in de code.

3.1.4 Testen

Parallel aan het programmeren van de workflow-configuratietool moest het testtraject plaatsvinden. De belangrijkste tests die moesten worden uitgevoerd waren de acceptatietests, waarin het product werd geverifieerd aan de hand van de opgestelde requirements. Daarnaast moesten ook unit-tests worden uitgevoerd om correcte werking en fouttolerantie van de applicatie op codeniveau te testen. De toepassing van unit-tests was belangrijk, omdat er in de applicatie veel opties voor gebruikersinvoer moesten komen waarin potentieel foute informatie zou kunnen worden ingevoerd. Het uitvoeren van de tests moest gelijktijdig met het ontwikkelen van de configuratietool plaatsvinden, omdat er veel belang was bij het volledig en correct implementeren van de requirements. Om deze reden is ook veel aandacht gegeven aan de acceptatietests, zodat kon worden achterhaald of de requirements volledig en acceptabel zijn gerealiseerd.
Om de acceptatietests uit te voeren zijn er acceptatiecriteria nodig om te kunnen evalueren. Daarom was besloten om, wanneer een requirement werd opgepakt om te ontwikkelen, hier eerst acceptatiecriteria bij te schrijven in overleg met de stagebegeleider. Aan de hand van deze criteria konden dan, wanneer de ontwikkeling van een specifiek requirement af was, acceptatietests worden gemaakt en uitgevoerd.
Omdat unit-tests voornamelijk de informatiestroom door de applicatie testen konden niet alle requirements een unit-test ondergaan. Er moest worden geanalyseerd welke stukken code gevoelig waren voor fouten zodat daar unit-tests voor konden worden geschreven.
Alle uitgevoerde tests moesten ook worden gedocumenteerd. Van de acceptatietests werd opgeschreven op welke requirements zij betrekking hadden, hoe de tests moesten worden uitgevoerd, en wat de resultaten van de test moesten zijn. Van de unit-tests werd opgeschreven welk stuk code werd getest, waarom dit een risicogebied was, en hoe de code met de test is gedekt.

3.2 Risicofactoren

Aan dit project zaten ook bepaalde risico's verbonden die gedurende het project in acht moesten worden genomen. Deze risico's waren met name van invloed op de planning van de ontwikkeling van de workflow-configuratietool, waarin zij regelmatig werden gecontroleerd (zie hoofdstuk 3.3.4).

3.2.1 Complexiteit workflows

Het grootste risicofactor binnen deze opdracht was de complexiteit van de workflows waar de configuratietool in moest voorzien. Er waren een groot aantal requirements te bedenken die moesten worden gerealiseerd om volledige en correcte workflows op te kunnen stellen binnen de configuratietool. De kans bestond daarom dat niet alle requirements binnen het afstudeertraject gerealiseerd konden worden. In de planning van de ontwikkeling zou dan ook een prioriteit worden gelegd op het ontwikkelen van een werkend en acceptabel prototype (zie hoofdstuk 2.3.4 "Afbakening") van de workflow-configuratietool. Daarna kon dieper op de andere requirements worden ingegaan.
Het probleem bij deze ontwikkeling trad op wanneer onverhoopt de minder belangrijke requirements te veel aandacht kregen. Dit probleem kon ontstaan doordat bepaalde requirements belangrijker werden geacht dan ze daadwerkelijk waren, maar het kon ook gebeuren dat bepaalde belangrijkere requirements over het hoofd waren gezien.
Dit kon worden voorkomen door zorgvuldig de lijst met requirements op te stellen en te prioriteren. Hiervoor moest goed worden vergaderd met de workflow-experts en de Product Owner. De opgestelde eisen konden vervolgens worden geprioriteerd op basis van het belang daarvan binnen het systeem en de tijd die benodigd was om deze onderdelen te realiseren. Tijdens het ontwikkelproces kon dit worden gecontroleerd door eerst goed te bepalen welke taken prioriteit hadden, alvorens deze werden opgepakt.

3.2.2 Gebruiksvriendelijkheid configuratietool

Tegenover de complexiteit van de workflows stond het feit dat de configuratietool zelf niet te complex mocht worden in zijn gebruik. Het was vastgesteld dat de gebruikers van de configuratietool weten wat een workflow is en wat ze moeten invullen om een correcte workflow te maken. Het belang lag er echter bij dat zij de tool ook zelfstandig en intuïtief konden gebruiken.
De gebruikers hoorden ook niet in het bezit te hoeven zijn van technische kennis over het achterliggende workflow-systeem. Zij zouden de tool op een voor hen begrijpelijke manier moeten kunnen gebruiken en niet verward raken door alle opties. Als een gebruiker steeds contact op moest nemen met Nétive voor ondersteuning, had de configuratie van workflows net zo goed bij de ontwikkelaars kunnen blijven liggen.
Het was dus van belang om niet te overschatten wat voor taken de gebruiker wil en kan uitvoeren. Dit kon worden voorkomen door een goed beeld te vormen van de kennis en mogelijkheden van de Service & Delivery afdeling van Nétive. Als een van de eindgebruikers van de tool konden zij namelijk duidelijk aangeven wanneer de tool te complex of onoverzichtelijk werd.

3.2.3 Beschikbaarheid tekentool

Een ander risicofactor was de beschikbaarheid van een bruikbare diagram-tekentool. Het was voor de ontwikkeling van de configuratietool van belang dat een goede en toepasbare tekentool werd gekozen als basis voor de configuratietool. Om deze te vinden zou een onderzoek worden uitgevoerd, maar de kans bestond dat hieruit geen acceptabel resultaat zou komen.
Als definitief was bepaald dat er geen diagram-tekentool beschikbaar was, zou een alternatief moeten worden verzonnen om workflows te presenteren. Ik had bijvoorbeeld zelf een infrastructuur kunnen ontwikkelen waarin de gebruiker met behulp van visuele representatie workflow-diagrammen kon tekenen, maar er waren ook andere presentatiemogelijkheden te bedenken waarbij niks werd getekend.
Het ontwikkelproces zou in deze situatie echter meer tijd in beslag nemen dan wanneer een bestaande tekentool kon worden gebruikt. Het doel van het project bleef echter dat in ieder geval een prototype kon worden gemaakt van de workflow-configuratietool.

3.3 Software-ontwikkelmethode

Voor het ontwikkeltraject moest ook een software-ontwikkelmethode worden gekozen om de ontwikkeling van de workflow-configuratietool te waarborgen. Er zal hier worden uitgelegd hoe deze is gekozen en hoe gepland was deze toe te passen in het ontwikkelingstraject.

3.3.1 Keuzefactoren

Om een toepasselijke ontwikkelmethode te kunnen kiezen, moest in kaart worden gebracht welke criteria waren voor de manier waarop het ontwikkeltraject moest worden uitgevoerd. Met de gekozen ontwikkelmethode moest het volgende kunnen worden bereikt:

- Zekerheid over de keuze van opgepakte requirements.
- Regelmatige evaluatie van opgeleverd werk en voortgang product.
- Voorkomen problemen door risicofactoren.

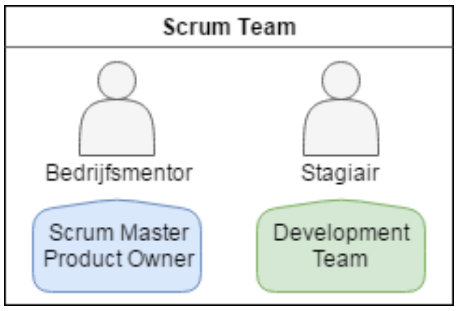
Er waren tijdens het opzetten van de planning nog geen requirements opgezet. Desondanks was door overleg met de bedrijfsmentor duidelijk geworden dat er een grote hoeveelheid functionele eisen voor de workflow-configuratietool bestonden, meer dan er waarschijnlijk binnen het afstudeertraject konden worden voltooid. Om deze reden was het belangrijk een prioritering aan te leggen in de requirements die ik zou oppakken. Bovendien zou deze prioritering ook tussentijds kunnen worden aangepast op basis van nieuwe inzichten die zijn verkregen over de wensen van de workflow-configuratietool.
Om de kwaliteit van het verrichte werk en het product te evalueren, zouden ook regelmatige reviews met de bedrijfsmentor plaats moeten vinden. Hierin kon worden beoordeeld of het werk kwalitatief in orde was en of het voldeed aan de requirements. Ook kon uit de staat van het product worden afgeleid of de ontwikkeling de goede kant op gaat of dat er wijzigingen in de prioritering van de requirements moesten worden gemaakt.
Naast vast te stellen of het product aan de requirements voldoet, moest ook worden vastgesteld dat het product niet een van de risicogebieden in gevaar bracht. Bij de review zou daarom ook moeten worden gekeken of er geen problemen zijn ontstaan op deze gebieden. Als dit wel is gebeurd, zouden maatregelen moeten worden genomen in de planning van de ontwikkeling. Dan zouden er nieuwe requirements kunnen worden gemaakt of zou de prioritering daarvan kunnen worden aangepast.

3.3.2 Scrum

Op basis van de hierboven genoemde projectspecificaties heb ik besloten om het Scrum framework toe te passen in mijn ontwikkeltraject. Met de Agile aanpak van Scrum te gebruiken kon ik beter garanderen dat ik de juiste workflow-configuratietool zou gaan opzetten. Door regelmatig te peilen of het ontwikkelde product overeenkwam met de requirements, kon worden verzekerd dat het product acceptabel en bruikbaar is.
Met de toepassing van Sprints kon deze controle ook consistent en in overleg met de stakeholders worden uitgevoerd. Door elke Sprint incrementele ontwikkelingen uit te voeren konden deze ook snel getest en geëvalueerd worden.
Ik heb een Scrum-team gevormd met mijn stagebegeleider, waarbij hij de rol van Scrum Master en Product Owner had en ik die van de Developer (zie Figuur 10). Voor de fasering van het ontwikkelproces heb ik gekozen om eenweekse sprints op te zetten, waarin wij aan het begin van elke sprint een Sprint Review en een Sprint Planning hielden (zie hoofdstuk 3.3.4 “Indeling Sprints”). Een onderdeel van het Scrum-proces is het bijhouden van een catalogus met taken die moeten worden uitgevoerd. Een zogeheten Product Backlog heb ik ook op moeten zetten voor mijn project (zie hoofdstuk 6.1 “Sprint Een”).

Nétive maakt zelf al langer gebruik van Scrum en heeft daarvoor een digitale omgeving ingericht met behulp van de applicatie JIRA (https://www.atlassian.com/software/jira). In deze omgeving kon een digitale Backlog worden opgezet en kon inzicht worden verkregen over de voortgang van sprints met onder andere burndown charts.

Figuur 9: Logo JIRA software



Figuur 10: Scrum Team indeling

3.3.3 Indeling Product Backlog

De Product Backlog moest worden ingevuld met User Stories. Deze Stories representeerden verschillende onderdelen van de workflow-configuratietool die moesten worden voltooid om deze volledig te kunnen realiseren en moesten elk zijn terug te leiden naar de requirements. De User Stories werden beschreven vanuit het perspectief van de gebruiker van de tool, geformuleerd volgens de structuur "Als <gebruiker> wil ik <functionaliteit>." Het was vervolgens de verantwoordelijkheid van het Scrum team om te bepalen welke taken hiervoor moesten worden uitgevoerd. Dit gebeurde in de Sprint Planning (zie "Indeling Sprints"). De functionele- en acceptatiecriteria die hieruit voort moesten komen, vormden samen met de opgestelde subtaken de zogeheten Definition of Done. Elke Story moest een complete Definition of Done bevatten zodat er geen onduidelijkheden waren over wat moest worden uitgevoerd, hoe en waarom.
Naast taken kreeg elke User Story ook een aantal Story Points mee. Dit puntenaantal gaf een indicatie van de relatieve omvang van de User Story en diende als richtlijn voor de planning van de sprints. Het bepalen van deze punten was een kwestie van inschatten hoeveel tijd een bepaalde User Story zou kosten en te kijken hoe deze zich verhoudt tot de andere Stories. De puntenaantallen waaruit gekozen kon worden waren 1, 2, 3, 5, 8, 13, en 20. Hierbij was ten eerste geschat dat mijn Velocity, de hoeveelheid werk die ik in een Sprint kon uitvoeren, op de dertien punten lag. Deze Velocity zou echter later kunnen worden bijgesteld op basis van de resultaten van een Sprint, met als gevolg dat er meer of minder User Stories konden worden ingepland.

3.3.4 Opzet Sprints

Als Scrum team hadden wij gepland om elke sprint te beginnen met een Sprint Planning en deze te beëindigen met een Sprint Review en Retrospective (zie Figuur 11: Sprint Opzet). Wij hebben besloten om de planning en review op dezelfde dag te doen, waarbij de review direct werd gevolgd door de planning. In dit hoofdstuk worden alle taken beschreven die wij hadden gepland in elke sprint op te pakken.

Sprint Planning

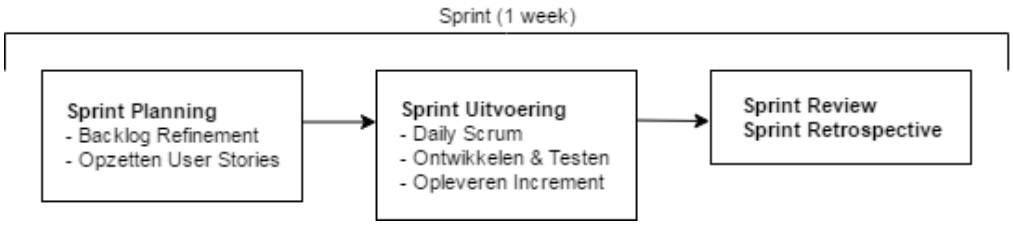
- De Scrummaster en ik kijken naar de Backlog en valideren of de opgestelde User Stories nog toereikend zijn voor het voltooien van het project. Als voor bepaalde gewenste functionaliteit nog geen User Story bestaan, worden hier een of meer User Stories voor geschreven. Waar nodig worden User Stories ook aangepast om gewenste functionaliteit beter te vertegenwoordigen. Wijzigingen of toevoegingen in de Backlog worden na de Sprint planning doorgevoerd naar de requirements zodat deze representatief blijven voor de wensen van de workflow-configuratietool.
- De Scrummaster en ik evalueren of de onderlinge prioritering van User Stories nog klopt. Dit wordt gedaan door de huidige focus van het project, zoals bepaald in de Sprint Review, te vergelijken met de User Stories bovenaan de Backlog. Als deze niet overeenkomen, wordt de volgorde in de Backlog aangepast zodat de meest belangrijke User Stories bovenaan komen te staan.
- Ik kijk naar de hoogst geprioriteerde User Stories en bepaalde hoeveel Story Points ik die Sprint op wil pakken op basis van mijn Velocity. Als een individuele Story te groot lijkt te zijn om in één sprint af te ronden, kan deze worden opgesplitst in kleinere User Stories. Dit is echter alleen mogelijk als ook uit deze Stories een tastbaar en werkend stuk functionaliteit voort kan komen.
- De Scrummaster en ik brainstormen per gekozen User Story over manieren om de betreffende functionaliteit te realiseren en stellen daar een aantal subtaken bij op. Waar mogelijk geeft de Scrummaster mij toegang tot reeds beschikbare informatie of code die mij zou kunnen helpen bij de ontwikkeling van de User Stories.

Sprint Uitvoering

- Aan het begin van elke dag wordt een Daily Scrum-meeting gehouden met een van de Scrum-teams van Nétive. Hierin bespreken de aanwezigen welke taken wij de voorgaande dag hebben uitgevoerd en welke taken we van plan zijn de komende dag uit te voeren. Hier is ook de gelegenheid om belemmeringen in de Sprint aan te kaarten en waar toepasselijk hulp te vragen aan anderen.
- Gedurende de dag voer ik de subtaken uit zoals die zijn opgesteld in de User Stories, met de opgestelde functionele criteria als richtlijn. Ontwikkelingen vinden plaats in de daarvoor opgestelde ontwikkelomgeving en worden door mij zelfstandig uitgevoerd. Als ik ergens hulp bij nodig heb, kan ik daarvoor bij de stagebegeleider terecht.
- In de afronding van elke User Story vinden testwerkzaamheden plaats om de acceptatiecriteria uit de Definition of Done te verifiëren en de correcte werking van de code te toetsen. Deze worden gedocumenteerd in een testrapport en zijn terug te leiden naar de User Stories. Als wordt vastgesteld dat de User Story nog niet "Done" was, wordt verder gewerkt tot dit wel geval is.
- Gedurende de Sprint kan de voortgang daarvan in de gaten worden gehouden met behulp van burndown charts. Deze schema's worden in JIRA gegenereerd en geven aan hoeveel Story Points nog moeten worden voltooid tegenover de resterende tijd voor de Sprint. Hierdoor kan worden vastgesteld of de Sprint volgens schema verloopt of dat er harder doorgewerkt moet worden.

Sprint Review & Retrospective

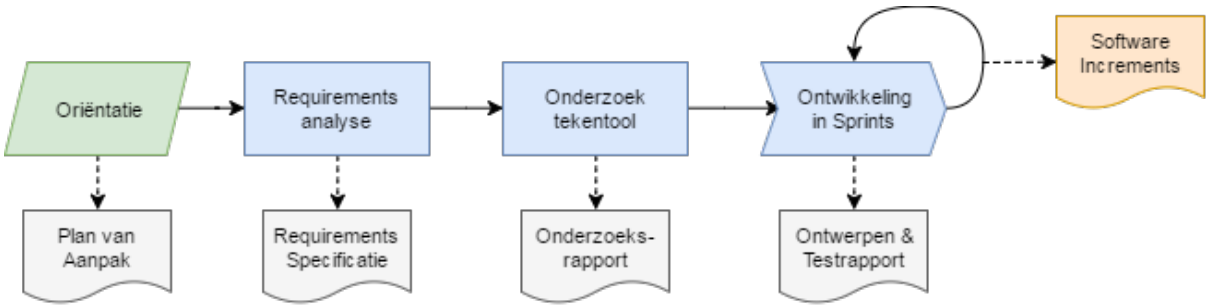
- Er wordt een demo gegeven aan de Product Owner over wat de afgelopen Sprint is ontwikkeld, waarop feedback wordt gegeven. De Product Owner kan hier vaststellen of het opgeleverde product voldoet aan zijn wensen en wat er nog gedaan moet worden om het product acceptabel te maken. Het is ook mogelijk dat hier nieuwe wensen naar voren komen, welke vervolgens in User Stories worden vastgelegd. Alle informatie die uit deze review naar voren komt wordt opgeschreven en kan als input worden gebruikt voor de Sprint planning.
- Bij de evaluatie van de ontwikkelingen worden ook de risicofactoren in de gaten gehouden. Zo kan worden bepaald of er door de uitgevoerde ontwikkelingen een probleem lijkt te ontstaan op een van deze gebieden. In dat geval maken wij nieuwe User Stories aan in de Backlog als maatregel om deze op te lossen. Waar nodig veranderen wij ook de prioritering van de User Stories om deze maatregelen voorrang te geven op anderen.
- Als in de afgelopen Sprint bepaalde User Stories niet konden worden afgerond, moeten hier maatregelen voor worden genomen. Dan wordt er gekeken waarom de Story niet is afgemaakt en wordt zo nodig de Velocity voor de Sprints aangepast. De incomplete User Story wordt dan in de daaropvolgende Sprint meegenomen en krijgt een aangepast aantal Story Points om de resterende hoeveelheid werk aan te geven.
- Tot slot is er de gelegenheid voor een Retrospective, waarin het team kan aankaarten wat goed en minder goed was verlopen in de afgelopen Sprint. Problemen die hier naar voren komen, worden aangepakt zodat deze niet in volgende Sprints terug zouden komen.



Figuur 11: Sprint Opzet

3.4 Gekozen aanpak

Met kennis van de taken, de opzet voor de ontwikkelings-werkzaamheden en de risicofactoren kon een planning voor het project worden opgezet. In de eerste week van het project heeft een oriëntatiefase plaatsgevonden waarin de opdracht verder is uitgewerkt en het plan van aanpak is opgezet. Deze planning is als volgt (zie Figuur 12).
In week twee van het project zouden de initiële requirements voor het project worden opgezet. Op basis van interviews met stakeholders en bedrijfsexperts zou een lijst met requirements worden opgezet en gereviewd. Uit dit proces moest een Requirements Specificatie-document voortkomen met een beschrijving van (de totstandkoming van) alle requirements.
In week drie en vier zou het onderzoek naar de diagram-tekentool starten. Hierin moesten eerst selectiecriteria worden gemaakt en moest een lijst met potentiële tekentools worden opgesteld. De criteria worden opgesteld aan de hand van de requirements, zodat de juiste tekentool kan worden gekozen. Daarom moest het onderzoek na de requirements analyse plaatsvinden. Na het opstellen van de selectiecriteria kon de software-selectie plaatsvinden, waaruit een onderzoeksrapport zou voortkomen waarin de keuze van de tekentool wordt uitgewerkt.
Nadat een diagram-tekentool was gekozen, kon de ontwikkeling van de configuratietool beginnen. Gepland was dit tot het eind van het project door te laten lopen, van week vijf tot en met week veertien. De ontwikkeling zou met het Scrum-framework in Sprints van één week worden uitgevoerd, volgens de planning opgezet in hoofdstuk 3.3.4. Als deel van het ontwikkeltraject zou ook een ontwerp worden gemaakt van het systeem en zou een testrapport worden gemaakt van de uitgevoerde acceptatie- en unittests. De reden dat het ontwerp pas na het onderzoek wordt gemaakt, is dat de gekozen diagram-tekentool deels van invloed is op het ontwerp.



Figuur 12: Plan van Aanpak (schematisch)

4. Requirements

Inleiding

De eerste stap in het ontwikkelen van de workflow-configuratietool is het achterhalen en documenteren van de productspecificaties. Deze requirements geven van een abstract tot een specifiek niveau aan welke wensen aan het product worden gesteld en hoe deze moet worden volbracht. In dit hoofdstuk zal worden verteld hoe deze requirements zijn geëliciteerd en hoe ze in een overzicht zijn vastgelegd. De volledige lijst met requirements is terug te vinden in de bijlage "Requirements Specificatie".

4.1 Stakeholders

Elk requirement, ongeacht het niveau, is gemaakt om de wensen van een of meer stakeholders te vervullen. Deze stakeholders bestonden uit mensen zowel binnen als buiten Nétive en hadden elk op hun eigen manier belang bij de realisatie van de workflow-configuratietool.

Producteigenaren

De producteigenaren waren de managers van Nétive en waren de eindverantwoordelijken voor het Vendor Management System en de workflow-configuratietool. Zij hadden duidelijke business doelen bij het project en wisten wat de wensen van de eindklanten waren.

Consultants

De consultants waren de uiteindelijke gebruikers van de workflow-configuratietool. Deze groep bestond uit mensen van twee verschillende afdelingen: de medewerkers van Service & Delivery van Nétive, en de partners van Nétive die de tussenpersoon waren voor het contact tussen Nétive en andere bedrijven.

Eindklanten

De eindklanten waren de mensen die gebruik maakten van het VMS en de workflows die daarin zijn opgenomen. Zij maakten dus geen gebruik van de workflow-configuratietool zelf, maar hadden wel baat bij het bestaan daarvan, dankzij de reductie in de tijd waarin consultants de klanten kunnen helpen.

4.2 Opzet requirements

Voor het opzetten van de requirements moest ik eerst weten welke soort requirements ik nodig had. Requirements kunnen uiteenlopen van algemene requirements over de business doelen van de software tot specifieke uitwerkingen over hoe een bepaalde functie in de software moet worden geïmplementeerd.
De requirements zouden worden opgedeeld in business-, gebruikers-, functionele- en niet-functionele requirements, en constraints. Elke requirement hoorde bij één categorie, maar vaak ontstonden uit één requirement verschillende andere requirements waardoor deze over meerdere categorieën werd uitgewerkt. Om deze reden is in het requirementsoverzicht gezorgd dat requirements altijd zijn terug te leiden naar hun oorsprong.
Omdat er is gekozen om met het Scrum-framework te werken, waren de requirements in deze fase nog niet geprioriteerd. De requirements zijn opgesteld zodat direct duidelijkheid was over de gewenste functionaliteit voor de workflow-configuratietool en zijn vervolgens gebruikt als input voor het opstellen van de Backlog in de Sprints. De Backlog zelf werd vervolgens wel geprioriteerd.

4.3 Elicitatie

Om een lijst met requirements op te kunnen stellen, is het nodig om met mensen te praten die kennis hebben van de eisen van het product. Verschillende mensen hebben vaak andere ideeën over het product maar door deze allemaal mee te nemen en af te wegen kan uiteindelijk een beter product worden opgezet.
Voor dit project zijn verschillende mensen geïnterviewd van verschillende afdelingen, elk op een andere manier betrokken bij het project. Voor elk persoon zal hier worden verteld hoe dit interview is aangepakt en wat daar de resultaten van waren. Voor met de interviews was begonnen, had ik al een conceptlijst requirements opgebouwd op basis van de kennis die ik over de opdracht had. Aan de hand van deze lijst kon ik meteen zien welke informatie nog mistte en waar nog onduidelijkheden waren.
Tijdens het uitvoeren van de interviews heb ik alle relevante informatie opgeschreven die daarin naar voren kwam en achteraf heb ik deze informatie verwerkt in de lijst met requirements.

4.3.1 Product Owner

De Product Owner van de workflow-configuratietool is de Product Development Manager. Ik heb deze persoon als eerste geïnterviewd over zijn wensen betreffende de workflow-configuratietool omdat hij degene was die het project beheerde. De Product Owner had voornamelijk kennis van de business kant van het project en kon daarom veel informatie vertellen over de business requirements.

4.3.2 Ontwikkelaar

Ook heb ik met een van de ontwikkelaars van Nétive gesproken, in dit geval mijn stagebegeleider, over de manier waarop de configuratietool kon worden opgebouwd. Deze persoon had ook veel algemene kennis over de werking van workflows en de mogelijkheden van de Force.com omgeving waar de tool in moest worden gebouwd. Uit deze informatie konden verschillende (niet-)functionele requirements en constraints worden opgemaakt.

4.3.3 Eindgebruiker

Tot slot heb ik een van de toekomstige eindgebruikers van de tool geïnterviewd, namelijk een medewerker van de Service & Delivery afdeling van Nétive. Deze persoon kon voornamelijk duidelijk maken waar de limieten lagen wat betreft de expertise van de eindgebruikers en had goede ideeën over manieren om de applicatie gebruiksvriendelijk te maken. Deze informatie gaf een belangrijk perspectief op de workflow-configuratietool en had daardoor veel invloed op de (niet-)functionele requirements.

4.4 Review

Na alle interviews had ik een volledige lijst met requirements opgebouwd. Ik had alle requirements uitgewerkt tot op functioneel niveau, elk terug te leiden naar de bijbehorende business en gebruikers requirements. Nu moesten de requirements nog worden aangescherpt, zodat deze voldoende duidelijk waren voor de opstart van het onderzoek.
Ik heb de requirements gereviewd met de Product Owner en de stagebegeleider. In deze gesprekken werden de requirements punt voor punt doorgenomen en werden fouten aangekaart. Na het proces van eliciteren en reviewen had ik voldoende vertrouwen in de requirements om het project voort te zetten. De requirements zijn later in het project gebruikt als input voor de selectiecriteria en de Scrum Backlog.

4.5 Requirements lijst

Om te demonstreren hoe de lijst met requirements is opgezet, zullen hier de meest belangrijke business requirements worden weergegeven met verschillende gebruikers en (niet-) functionele requirements die daaruit zijn voortgekomen.

Business Requirements

De volgende business requirements beschrijven de manier waarop de workflow-configuratietool workflows moet kunnen produceren en hoe de verschillende stakeholders daarbij zijn betrokken. De kolom "ID" heeft hier geen specifieke betekenis, maar wordt gebruikt om terug te kunnen refereren vanuit andere requirements. Deze ID komt overeen met die in de bijlage "Requirements Specificatie" voor referentie.

ID	Beschrijving
BR-01	De Product Owner wil dat consultants de mogelijkheid hebben om workflows op te stellen, te beheren en te distribueren met behulp van een configuratietool.
BR-02	De Product Owner wil dat de workflows opgesteld in de configuratietool op dezelfde manier kunnen worden gebruikt als die van bestaande workflows.
BR-03	De Product Owner wil dat consultants de workflow-configuratietool makkelijk en zelfstandig kunnen gebruiken.
BR-04	De Product Owner wil dat consultants de workflow-configuratietool kunnen gebruiken voor documentatie doeleinden.
BR-05	De Product Owner wil dat programmeurs minder tijd spenderen aan het opzetten van workflows.
BR-06	De Product Owner wil dat klanten minder lang hoeven te wachten totdat zij gebruik kunnen maken van hun workflows.

Gebruikers Requirements

De volgende gebruikers requirements zijn een paar voorbeelden van requirements die voortkwamen uit de business requirements. In de kolom "Uit" is te zien van welke requirement deze afkomstig zijn. Deze requirements beschrijven hoe de gebruiker de workflow-configuratietool moet kunnen gebruiken. Deze gebruikswijzen worden uitgewerkt in functionele requirements die de functionaliteit op systeemniveau beschrijven.
De Gebruikers Requirements gevolgd uit BR-01 gaan verder in op hoe de configuratietool gebruikt moet worden en hoe workflows zelf betrokken zijn bij het systeem. De requirements uit BR-02 geven aan welke objecten en informatie de gebruiker in de tool moet kunnen toevoegen, overeenkomstig met de informatie in bestaande workflows. Tot slot geeft requirement 11 aan welke hulpmiddelen de gebruiker heeft bij het tekenen van workflows.

ID	Beschrijving	↳Uit
UR-01	Consultants moeten bestaande workflows kunnen openen.	BR-01
UR-02	Consultants moeten nieuwe workflows kunnen aanmaken.	BR-01
UR-04	Consultants moeten workflows die niet met de workflow-configuratietool zijn gemaakt kunnen importeren.	BR-01
UR-07	Consultants moeten statussen aan de workflow kunnen toevoegen, bewerken en verwijderen.	BR-02

UR-08	Consultants moeten namen, beschrijvingen en toegangsrechten van statussen kunnen invoeren, wijzigen en verwijderen.	BR-02
UR-09	Consultants moeten transities tussen statussen kunnen vastleggen in de workflow en deze kunnen bewerken en verwijderen.	BR-02
UR-10	Consultants moeten labels, condities, rechten en acties van transities kunnen invoeren, wijzigen en verwijderen.	BR-02
UR-11	Consultants moeten losstaande labels en lijnen aan het workflow-diagram kunnen toevoegen.	BR-04

Niet-functionele Requirements

De volgende niet-functionele requirements komen voort uit de business requirements en geven kwalitatieve criteria over de manier waarop het systeem moet werken.
De eerste requirement gaat wederom in op de manier waarop workflows moeten worden opgezet, namelijk dat ze enkel valide zijn wanneer ze voldoen aan de structuur van bestaande workflows. De requirements gevolgd uit BR-03 geven kwalitatieve eisen aan waarmee de gebruiker de tool makkelijk en zelfstandig kan gebruiken, zoals beschreven in de Business Requirement.

ID	**Beschrijving**	**↳Uit**
NF-01	Het systeem moet valide workflows produceren die voldoen aan de structuur van bestaande workflows.	BR-02
NF-02	Het systeem moet makkelijk en intuïtief zijn om te gebruiken.	BR-03
NF-03	Het systeem moet volgens de "What You See Is What You Get" techniek zijn vormgegeven.	BR-03
NF-04	De schematische opmaak van workflows moet bewaard blijven tussen sessies.	BR-03
NF-06	De workflows getekend in het systeem moeten voor documentatie doeleinden zijn te gebruiken.	BR-04

Constraints

De volgende constraints komen voort uit de business requirements en geven harde grenzen aan de manier waarop deze worden in het systeem worden gerealiseerd.
De bestaande workflows worden in XML-formaat in het bestaande systeem gebruikt en daarom is deze constraint terug te zien in C-01. Het Salesforce Lightning Design System (SLDS) is een gestandaardiseerde vormgeving voor websites opgezet door Salesforce. Nétive heeft gekozen deze toe te passen in de gehele Self Service Desk en is daarom ook een constraint voor dit project.

ID	**Beschrijving**	**↳Uit**
C-01	Het systeem moet workflows in XML-formaat distribueren.	BR-02
C-02	Het systeem moet door mensen zonder technische achtergrond te gebruiken zijn.	BR-03
C-05	Het systeem moet met het Salesforce Lightning Design System zijn vormgegeven.	

Functionele Requirements

Tot slot worden hier de belangrijkste functionele requirements beschreven zoals die zijn ontstaan vanuit de bovenstaande requirements. Deze functionele requirements zijn in de Sprints omgezet in Backlog items en in de ontwikkeling meegenomen.

Workflow-configuratietool algemeen

De requirements in de volgende tabel geven aan hoe het systeem de workflows moet beheren. De workflow-configuratietool moet workflows kunnen openen en tonen aan de gebruiker, maar moet ook bestaande workflows kunnen importeren.

ID	Beschrijving	↳Uit
FR-01	Het systeem moet een workflow-configuratie kunnen inladen.	UR-01
FR-02	Het systeem moet elementen uit een workflow grafisch kunnen weergeven.	UR-01
FR-03	Het systeem moet een overzicht van bestaande workflow-configuraties kunnen geven.	UR-01
FR-04	Het systeem moet een nieuwe workflow-configuratie kunnen aanmaken.	UR-02
FR-05	Het systeem moet workflows die niet met de configuratietool zijn gemaakt kunnen importeren.	UR-04

Statussen & Transities

De volgende requirements gaan in op de informatie die de gebruiker in het systeem moet invoeren met betrekking tot de statussen en transities. Er wordt aangegeven welke informatie dit is en waar deze vandaan moet worden gehaald.

ID	Beschrijving	↳Uit
Statussen		
FR-18	Het systeem moet statussen aan een workflow-diagram kunnen toevoegen.	UR-07
FR-19	Het systeem moet de naam, beschrijving en fase van een status kunnen laten invullen en wijzigen.	UR-08
FR-20	Het systeem moet de naam, beschrijving en fase van een status kunnen weergeven.	UR-08
FR-21	De beschikbare statusnamen moeten uit de gekoppelde Salesforce-database worden ingelezen.	NF-01
FR-23	Het systeem moet de lees- en schrijfrechten van een status kunnen laten invullen en wijzigen.	UR-08
FR-24	Het systeem moet de lees- en schrijfrechten van een status kunnen weergeven.	UR-08
FR-26	De beschikbare gebruikersrollen moeten uit de gekoppelde Salesforce-database worden ingelezen.	NF-01
FR-27	Het systeem moet een status in de vorm van een rechthoek in het diagram weergeven.	FR-18
Transities		
FR-30	Het systeem moet een transitie kunnen toevoegen door een lijn van een status naar een andere status te trekken.	UR-09
FR-33	Een status moet met meerdere verschillende statussen en met zichzelf kunnen zijn verbonden.	FR-30

FR-34	Het systeem moet de richting van een transitie aan kunnen geven en wijzigen.	FR-30
FR-35	Het systeem moet de naam, beschrijving en modus van een transitie kunnen laten invullen en wijzigen.	UR-10
FR-36	Het systeem moet de naam, beschrijving en modus van een transitie kunnen weergeven.	UR-10
FR-37	Het systeem moet een of meer transitierechten, condities en acties aan transities kunnen toekennen.	UR-10
FR-38	Het systeem moet de transitierechten en acties van een transitie kunnen weergeven.	UR-10

Gebruiksvriendelijkheid

De volgende requirements zijn bedacht om de tool meer gebruiksvriendelijk te maken. Zij geven aan hoe bepaalde informatie in het systeem moet worden getoond en het systeem zich moet gedragen om de gebruiker een betere ervaring te geven.

ID	**Beschrijving**	**↳Uit**
FR-25	Lees- en schrijfrechten moeten kunnen worden ingevuld door een of meerdere gebruikersrollen in een lijst aan te vinken.	NF-02
FR-28	Het systeem moet de naam en beschrijving van een status in het diagram laten zien.	NF-03
FR-29	Het systeem moet de optie geven om de grootte van een status aan te passen.	NF-02
FR-31	Het systeem moet op een bestaande lijn meerdere transities toe kunnen laten voegen.	NF-02
FR-39	Transitierechten moeten kunnen worden ingevuld door een of meerdere gebruikersrollen in een lijst aan te vinken.	NF-02
FR-40	Het systeem moet op elke status meerdere contactpunten weergeven van waaruit transities kunnen worden getrokken en waarop transities kunnen worden gekoppeld.	NF-02
FR-56	Het systeem moet markeren welk object in de workflow is geselecteerd.	NF-02
FR-60	Het systeem moet met behulp van context-menu's toegang geven tot alle functionaliteit waarmee objecten in het systeem kunnen worden toegevoegd, gewijzigd en verwijderd.	NF-02

Validatie & Export
De requirements hieronder beschrijven hoe het systeem moet zorgen dat er valide workflows worden gemaakt en hoe het deze moet exporteren naar het VMS.

ID	Beschrijving	↳Uit
FR-50	Het systeem moet kunnen controleren of de workflow-onderdelen en -informatie in de workflow aan de opgegeven validatieregels voldoen.	NF-01
FR-51	Het systeem moet weerhouden dat bepaalde workflow-onderdelen en -informatie kunnen worden verwijderd aan de hand van de opgegeven validatieregels.	NF-01
FR-52	Het systeem moet kunnen controleren of het eindpunt van de workflow vanuit het beginpunt bereikt kan worden.	NF-01
FR-53	Het systeem moet kunnen aanwijzen welk onderdeel van een workflow invalide is of mist.	NF-02
FR-55	Het systeem moet voorkomen dat een niet-valide workflow via de configuratietool gedistribueerd kan worden.	NF-01
FR-07	Het systeem moet de informatie in de workflows kunnen converteren van en naar XML-formaat.	C-01
FR-08	Het systeem moet workflows naar een test- of productieomgeving kunnen distribueren.	UR-05

Conclusie
Bovenstaande requirements zijn maar een selectie van alle requirements die zijn opgesteld, maar zij geven wel aan hoe vanuit enkele business requirements verschillende andere requirements zijn ontstaan. De requirements zijn ook elk terug te leiden naar het business requirement waar zij uit zijn voortgekomen.

5. Onderzoek

Inleiding
Aansluitend op het onderdeel requirements heeft een onderzoek plaatsgevonden naar de beschikbaarheid van software waarmee in een webapplicatie een grafische representatie van een workflow kon worden gegeven. In dit hoofdstuk zal worden uitgelegd wat de eisen zijn die aan de tekentool zijn gesteld, en hoe dit onderzoek is opgestart en uitgevoerd. Tot slot zullen de resultaten van het onderzoek worden uitgelegd en zal duidelijk worden waarom de gekozen software is geselecteerd. Het volledige onderzoeksrapport dat uit dit onderdeel heeft geresulteerd is terug te vinden in de bijlage "Onderzoeksrapport".

5.1 Algemene software-specificaties

De diagram-tekentool die met het onderzoek moest worden geselecteerd, moest aan een aantal eisen voldoen. Deze eisen stammen af van de opgestelde requirements, achterhaald in overleg met de Product Owner en Stakeholders van het project. Uit deze specificaties zijn selectiecriteria gemaakt, die duidelijk afbakenen waar de software aan moet voldoen en welk belang dit heeft.

5.1.1 Implementatie

Het was in de doelstelling van de opdracht vastgesteld dat de workflow-configuratietool deel zou gaan uitmaken van de Self Service Desk (SSD). Nétive wilde de gebruikers van de SSD geen extra software laten installeren maar, net als het VMS, een webapplicatie bieden.
Dit betekende voor de workflow-configuratietool en de bijbehorende diagram-tekentool dat zij ook in een web-omgeving moesten kunnen draaien. Tekentools die dit niet konden, vielen daarom per definitie meteen al af. Ook kon de tekentool geen op zichzelf staande webapplicatie zijn. De tekentool moest geïntegreerd worden in een Salesforce omgeving en contact maken met de bijbehorende database. Een standalone tekentool zit echter ingebouwd in een aparte website en kon daarom niet gebruikt worden.

5.1.2 Workflow-structuur

Zoals gedefinieerd in Business Requirement nummer 2 moeten de workflows gemaakt in de workflow-configuratietool aan de informatiestructuur van bestaande workflows voldoen. Deze structuur moest worden afgedwongen in de programmatuur van de applicatie.
In de requirements is bijvoorbeeld beschreven dat statussen een naam moeten krijgen (FR-19). Als de gebruiker in de configuratietool een status tekent, moet de optie aanwezig zijn om aan dit specifieke statusobject een naam toe te voegen. Het vierkantje op het scherm representeert het object, maar de gebruiker moet achter dit object alle informatie kunnen toevoegen die in een workflow hoort te zitten.
Om dit mogelijk te maken moet er toegang zijn tot de backend van de objecten getekend in de tekentool. Hiervoor zijn bepaalde functies nodig ingebouwd in de tekentool, die mij toegang moeten geven tot de getekende objecten en hun achterliggende informatiestructuur. Zo kan gebruikersinput automatisch worden verwerkt en opgeslagen. Deze informatie kan niet ergens los worden opgeslagen in het systeem, omdat dan niet meer is te achterhalen welke informatie bij welke objecten hoort.

5.1.3 Gebruikers

Zoals vastgesteld in Business Requirement nummer 3 beschikten de gebruikers van de workflow-configuratie niet over een technische achtergrond. Zij wisten wat workflows zijn en welke informatie erin moest zitten, maar konden niet verwacht worden een geheel XML-bestand zelf op te zetten met alle technische informatie die zich daarin bevindt. Om deze reden moet de tekentool zodanig kunnen worden aangepast dat deze makkelijk kan worden gebruikt en niet meer componenten in de interface heeft dan nodig is.
Verder zijn er bepaalde stukken functionaliteit die in de uiteindelijke workflow-configuratietool moeten komen, die reeds in een diagram-tekentool zouden kunnen zijn geïmplementeerd. Voorbeelden hiervan zijn functies die kunnen worden aangeroepen om elementen aan het workflow-diagram toe te voegen of met elkaar te verbinden, of een manier om elementen in het diagram een kleur te kunnen geven. Idealiter bevat een tekentool deze functies uit zichzelf, maar als dit niet het geval is moet er in ieder geval een manier zijn om deze functies toe te voegen aan de tekentool.

5.1.4 Kosten

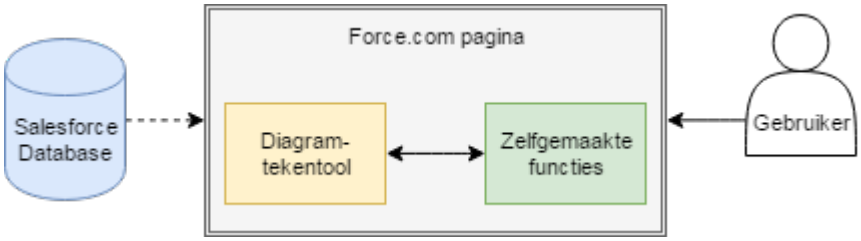
Tot slot wilde Nétive voor dit project het liefst nog geen software-licenties aanschaffen, tenzij het echt noodzakelijk bleek te zijn. Dit betekende dat er voor een gratis tekentool of een tool met een proefperiode moest worden gekozen. Voor tekentools die enkel met een proefperiode beschikbaar waren, moest ook worden gekeken dat deze niet verliep voor het eind van het project. Hierbij stond vooropgesteld dat het product gemaakt met de tool uiteindelijk wel commercieel mocht worden ingezet door Nétive.

5.2 Selectiecriteria

Uit de opgestelde eisen voor de diagram-tekentool zijn een aantal selectiecriteria opgesteld. Aan de hand van deze criteria kon elke tekentool worden beoordeeld en afgewogen tegenover de andere tools. Van elk criterium zal hier een inhoudelijke beschrijving worden gegeven en wordt aangegeven hoe gemeten kan worden in welke mate een tekentool de criteria vervuld.

5.2.1 Implementatiemogelijkheden

De belangrijkste functionele eis van de software is dat deze in een webpagina moest kunnen worden geïmplementeerd in een Force.com omgeving. Als dit niet mogelijk was, zou de gebruiker een externe applicatie moeten installeren om gebruik te kunnen maken van de workflow-configuratietool en dat was geen optie. Het was hierbij voldoende als de software op een aparte pagina stond, los van de andere Force.com pagina's. Belangrijker was dat de applicatie kon communiceren met de Salesforce omgeving waar het zich in bevond, zodat het toegang had tot de bijbehorende database.
De programmeertaal waarin de tekentool was opgezet was daarom ook niet relevant als selectiecriterium. Als een bepaalde tekentool in een webapplicatie kon worden ingebouwd, was dat voldoende.
Om dit criterium te evalueren moest de tekentool worden geïmporteerd in een Force.com omgeving waarin een testapplicatie werd opgezet. In deze applicatie moesten ook enkele elementen kunnen worden getekend met behulp van de functies ingebouwd in de tool. Als de tool op dit aspect faalde, was de tool per direct uitgesloten voor de selectie.



Figuur 13: Context implementatie tekentool

5.2.2 Aanpasbaarheid

De software moest zodanig kunnen worden aangepast dat het aan de opgestelde requirements voldeed betreffende de informatievoorziening en gebruiksvriendelijkheid. Zoals genoemd in de requirements moesten verschillende variabelen, waaronder namen, beschrijvingen en rechten, aan de objecten in het diagram kunnen worden toegevoegd. Dit was alleen mogelijk als de tekentool functies bood waarmee toegang kon worden verkregen tot de objectstructuur achter deze elementen.
Ook moesten verschillende functies in de tekentool kunnen worden ingebouwd om een betere gebruikerservaring te creëren. Zo moest het systeem kunnen detecteren wanneer de gebruiker op een diagram-element klikt, zodat informatie over het element in de applicatie kon worden weergegeven. Ook context-menu's waren gewenst voor de tool, met daarin functies specifiek gemaakt voor de workflow-configuratietool.
Als een bepaalde tekentool niet de mogelijkheid voor dit soort aanpassingen bood, zou een oplossing moeten worden bedacht buiten de tool om. Dit zou nadelig zijn voor de ontwikkeling omdat dit meer werk oplevert.

Om dit criterium te evalueren moest worden geprobeerd de structuur en opmaak van elementen te veranderen in een testimplementatie van de tekentool. De aanwezigheid van ingebouwde functies van de tool en de mogelijkheid voor handmatige uitbreidingen zou hierbij naar voren moeten komen. De score van een tool op dit criterium was dus afhankelijk van het wel of niet kunnen aanpassen van elementen in de tool en de mate waarin dit gedaan kon worden.

5.2.3 Tekenfuncties

De software moest van zichzelf zo veel mogelijk functionaliteit bieden dat aansloot op de opgestelde requirements. Hoe minder tekenfunctionaliteit handmatig hoefde te worden toegevoegd, hoe meer tijd er kon worden besteed aan de andere requirements van de configuratietool.
Het was bijvoorbeeld gewenst dat de tool opties aan de gebruiker bood om objecten aan het workflow-diagram toe te voegen en deze met elkaar te verbinden. Ook waren functies gewenst om objecten te kunnen kleuren of vervormen, en om het gehele tekenveld te kunnen verschuiven of uit te vergroten.
Als te veel van dit soort functionaliteit handmatig zou moeten worden gemaakt, zou er minder tijd beschikbaar zijn voor andere onderdelen van de tool. De uiteindelijk kwaliteit van het product zou dan lager zijn.
Om dit criterium te evalueren moest de documentatie van de diagram-tekentool worden doorgenomen. Hierdoor kon een beeld worden gevormd van de hoeveelheid functionaliteit die in de tekentool zat ingebouwd en van de functionaliteit die handmatig zou moeten worden geprogrammeerd. De score van de tool op dit criterium was dus afhankelijk van het aantal functies ingebouwd in de tekentool dat betrekking had op het tekenen van elementen in de tool en dat relevant was voor de requirements. Niet alle soorten functionaliteit hadden even veel prioriteit, dus was het van belang om eerst te kijken naar de meer fundamentele functionaliteit voor het tekenen van objecten.

5.2.4 Uitleesmogelijkheden

De informatie opgeslagen in het diagram moest ook kunnen worden uitgelezen. De workflow-configuratietool moest niet alleen workflows kunnen opstellen, maar moest deze ook kunnen opslaan in de Force.com omgeving en in XML-formaat distribueren naar verschillende omgevingen in het VMS. Hiervoor zou een tussenstap nodig zijn om de informatie te vertalen naar de structuur die reeds in het VMS gebruikt wordt, maar daarvoor moest eerst de informatie uit de tekentool zelf kunnen worden onttrokken.
De structuur van deze informatie was niet zeer van belang, aangezien deze sowieso al moest worden geconverteerd. De informatie moest wel via de programmatuur van de applicatie kunnen worden verwerkt, dus een foto of pdf-bestand zou niet voldoende zijn. Een XML- of JSON-bestand zou daarentegen wel door het systeem verwerkt kunnen worden. Als een bepaalde tekentool van zichzelf geen uitleesmogelijkheden bood, zou deze functionaliteit handmatig moeten worden gemaakt.
Om dit criterium te evalueren moest in de documentatie van de tekentool worden gezocht naar functies waarmee de elementen in de tool konden worden geprogrammeerd. Als een bepaalde tool niet over dergelijke functionaliteit beschikte, was dat een significant tegenargument voor het gebruik daarvan.

5.2.5 Prijsstelling en Licentietype

Het was niet gewenst om een licentie aan te moeten schaffen om gebruik te kunnen maken van de software. Nétive wilde liever eerst een prototype van de workflow-configuratietool zien, voor zij zouden besluiten een licentie aan te schaffen. Daarom moest worden gekozen voor een diagram-tekentool met een proefperiode of een die volledig gratis te gebruiken was.
Nétive wilde uiteindelijk ook de workflow-configuratietool aan hun klanten gaan verkopen als deel van een softwarepakket. Een diagram-tekentool waarvan de licentie weerhoudt dat deze voor commerciële doeleinden mag worden gebruikt, was daarom ongewenst.
Om deze criteria te evalueren moest op de site van de eigenaar van de tekentool worden gezocht naar informatie over de kosten en licenties van het gebruik van de tool. Een gratis tool was daarbij een vereiste. Een tool met een proefperiode langer dan het project termijn was ook voldoende. Als een tool niet commercieel mocht worden gebruikt, was dat een hard afvalcriteria. Verder was het gewenst dat de tool open source was, maar dit was niet vereist.

5.2.6 Documentatie en Ondersteuning

Naast bovenstaande functionele criteria waren er ook kwalitatieve criteria van toepassing voor de keuze van de diagram-tekentool. Het was van belang dat er makkelijk geprogrammeerd kon worden met de software en dat er geen bugs in de software zaten. Daarvoor was het gunstig dat de software op een overzichtelijke en object-georiënteerde manier was opgezet, goed was gedocumenteerd, en dat er een actief ontwikkelingsteam achter zat die bugfixes uitvoerde en updates uitbracht.
Een goede documentatie van software betekende dat alle functionaliteit in de software tot in detail is beschreven. Aan deze documentatie kon ook worden afgeleid of de klassen en functies een duidelijke naamgeving hebben gekregen en of duidelijk is beschreven hoe deze in de praktijk moeten worden gebruikt. Als in de ontwikkeling van de workflow-configuratietool zou blijken dat de documentatie van een functie tekortschiet, zou hulp kunnen worden gezocht door contact op te nemen met de supportdesk van de eigenaar van de tool. Ook zou hulp kunnen worden gezocht in secundaire bronnen, zoals Stack Overflow. Het is daarvoor wel van belang dat er een actieve gemeenschap aanwezig is die deze bronnen relevant houdt en reageert op vragen.
Om deze criteria te evalueren moest de documentatie van de tekentool worden doorgenomen en beoordeeld. Ook moest worden bepaald in hoeverre primaire en secundaire ondersteuningsbronnen aanwezig zijn voor de tool. Daarnaast moest op het internet worden gezocht naar de aanwezigheid van bugs in de tool en moest worden beoordeeld hoe de ontwikkelaars van de tool hiermee omgingen. Een tool met goede documentatie en support, met een actieve community, en met weinig bugs, scoorde dus hoger op dit criterium.

5.3 Beoordelingssysteem

Om te kunnen bepalen welk diagram-tekentool de beste optie was, moesten deze worden beoordeeld aan de hand van bovengenoemde selectiecriteria. Om deze beoordeling objectief en consistent uit te kunnen voeren, is een systeem opgesteld waarin punten konden worden gegeven aan de verschillende criteria. Aan de hand van deze punten kon worden bepaald welke diagram-tekentool het meest geschikt was als basis voor de workflow-configuratietool.

5.3.1 Weging criteria

Van alle selectiecriteria moest een onderlinge weging worden opgesteld, om het belang daarvan bij de keuze van de software vast te leggen. Sommige criteria waren belangrijker dan anderen en daarom was het belangrijk om hier onderscheid in te maken. Er waren ook een paar criteria waar een tekentool direct op zou afvallen als daar niet aan werd voldaan. Deze afvalcriteria zijn apart genoemd en waren gebruikt om de longlist te reduceren naar een shortlist.
Van de overige criteria is een procentuele verhouding opgesteld op basis van de significantie daarvan binnen de keuze van de diagram-tekentool. Een hoger percentage overeenkomst met de criteria duidt op een betere tekentool.
De aanpasbaarheid van de tekentool en de aanwezigheid van tekenfuncties waren hierbij de belangrijkste criteria, want zij hadden de meeste invloed op de hoeveelheid werk die in de requirements moest worden gestoken. De uitleesbaarheid van het diagram en de kwestie of de tool open source was hadden hier ook betrekking op, maar in mindere mate. De aanwezigheid van goede documentatie en ondersteuning zou een leuke toevoeging zijn, maar was minder significant dan de daadwerkelijke functionaliteit van de tekentool.

ID	**Naam**	
Afvalcriteria		
A	Implementeerbaar in webapplicatie	
B	Gratis software	
C	Commercieel bruikbaar	
Kwalitatieve criteria		**Weging**
D	Aanpasbaarheid tekentool	30%
E	Aanwezigheid tekenfuncties	30%
F	Uitleesbaarheid diagram	15%
G	Open Source	15%
H	Documentatie	5%
I	Ondersteuning	5%
Totaal:		**100%**

5.3.2 Punten

Om de mate van overeenkomst tussen een diagram-tekentool en de criteria aan te geven, is gebruik gemaakt van een puntensysteem. Deze punten lopen van 1.0 tot -1.0 en geven per criterium aan of de tekentool daar wel of niet aan voldoet. Zie het tabel hieronder voor de scores die aan een criterium konden worden meegegeven.

Score	**Betekenis**
1.0	De software is volledig in overeenstemming met de eisen van het selectiecriterium.
0.5	De software is deels in overeenstemming met de eisen van het selectiecriterium.
0.0	De software is noch in overeenstemming noch in strijd met de eisen van het selectiecriterium.
-0.5	De software is deels in strijd met de eisen van het selectiecriterium.
-1.0	De software is volledig in strijd met de eisen van het selectiecriterium.

5.4 Uitvoering onderzoek

Met kennis van de software-specificaties en de criteria waarop ik deze moest beoordelen, kon het onderzoek van start gaan. Hiervoor is eerst een kandidatenlijst opgesteld met diagram-tekentools. Deze lijst is vervolgens door middel van een beoordelingstraject teruggebracht tot één tekentool die het best als basis voor de workflow-configuratietool kon worden gebruikt.

5.4.1 Opstellen longlist

Voor het opstellen van een longlist ben ik op het internet op zoek gegaan naar websites die een diagram-tekentool aanboden. Hiervoor zijn zoektermen gebruikt als "diagram drawer", "javascript diagram", "draw diagram api" en "draw diagram library".

De lijst met gevonden diagram-tekentools is als volgt:

Activiti BPMN	(Alfresco Software; https://www.activiti.org/)
Cacoo	(Nulab Inc.; https://cacoo.com/)
Creately	(Cinergix Pty. Ltd.; https://creately.com/)
D3JS	(Mike Bostock; https://d3js.org/)
Draw2D	(Andreas Herz; http://www.draw2d.org/draw2d/)
Draw.IO	(JGraph Ltd.; https://www.draw.io/)
FlowchartJS	(Adriano Raiano; http://flowchart.js.org/)
Gliffy	(Gliffy Inc.; https://www.gliffy.com/)
GoJS	(Northwoods Software; https://gojs.net)
JointJS	(Client.IO; https://www.jointjs.com/opensource)
JsPlumb CE	(jsPlumb Inc.; https://jsplumbtoolkit.com/)
Lucidchart	(Lucid Software; https://www.lucidchart.com/)
Mermaid	(Knut Sveidqvist; https://knsv.github.io/mermaid/)
MxGraph	(JGraph Ltd.; https://jgraph.github.io/mxgraph/)
Raphael	(Dmitry Baranovskiy; http://dmitrybaranovskiy.github.io/raphael/)
Rappid	(Client.IO; https://www.jointjs.com/)
Visio	(Microsoft; https://products.office.com/nl-nl/visio/flowchart-software)
Visual Paradigm	(VP International; https://www.visual-paradigm.com/)

Alle bovenstaande diagram-tekentools zijn gekozen met als eis dat er flowcharts mee konden worden getekend. Er had nog geen beoordeling plaatsgevonden op basis van de selectiecriteria.

5.4.2 Longlist naar shortlist

Na het opstellen van een longlist met diagram-tekentools, was het de bedoeling om deze te reduceren tot een shortlist. Deze reductie is gedaan met behulp van de afvalcriteria, opgesteld in hoofdstuk 5.3.1.

Uitvoering

Om te bepalen of een tekentool in een webapplicatie kon worden geïmporteerd, was soms enkel een kwestie van kijken naar de website van het product. Daar stond dan al aangegeven dat een desktopapplicatie of standalone webapplicatie was. Van de tekentools waar wel een losstaande versie beschikbaar van was, werd eerst gekeken of deze ook gratis en commercieel beschikbaar waren. Als dit het geval was, werd de tool geïmporteerd in een Salesforce testomgeving die ik had opgezet. Hierin werd geprobeerd de tool als deel van de webpagina in te laden en om enkele tekenfuncties uit te voeren.

Onderzoeksresultaten

De volgende tabel laat van elke tool in de longlist zien aan welke afvalcriteria het wel en niet voldeed. De diagram-tekentools die aan alle drie de criteria voldeden (hieronder dikgedrukt), werden opgenomen in de shortlist.

Naam tekentool	A. Implementatie web	B. Gratis applicatie	Commercieel bruikbaar
Activiti BPMN		✓	✓
Cacoo			
Creately			
D3JS	✓	✓	✓
Draw2D	✓		✓
Draw.IO		✓	✓
FlowchartJS	✓	✓	✓
Gliffy			
GoJS	✓		✓
JointJS	✓	✓	✓
JSPlumb CE	✓	✓	✓
Lucidchart			
Mermaid	✓	✓	✓
MxGraph	✓	✓	✓
Raphael	✓	✓	✓
Rappid	✓		✓
Visio			
Visual Paradigm			

5.4.3 Beoordeling shortlist

De shortlist met diagram-tekentools bestond nu uit D3JS, FlowchartJS, JointJS, JSPlumb CE, Mermaid, MxGraph en Raphael. Het was nu de taak deze resterende tekentools te bestuderen en te testen. Hiervoor is wederom gebruik gemaakt van de Salesforce testomgeving die ik had opgezet. Hierin zijn alle diagram-tekentools in de shortlist ingeladen en is voor elke tool een testapplicatie gemaakt.
Voor elk selectiecriterium zijn specifieke onderdelen van de tool onderzocht, zoals beschreven in hoofdstuk 5.2. Deze hebben vervolgens een score gekregen om aan te geven in hoeverre zij overeenkomen (of conflicteren) met de selectiecriteria. De scores zijn gebaseerd op wat de tekentool van zichzelf bood ten tijde van het onderzoek, niet op wat in theorie handmatig gemaakt zou kunnen worden.

Beoordeling

Hieronder zijn twee tabellen gegeven met de resultaten van het onderzoek. Alle zeven diagram-tekentools hebben voor elk criterium een score gekregen, waarmee een procentuele weging werd gecalculeerd. De percentages zijn bij elkaar opgeteld en onderaan de tabel neergezet. Een hoger percentage betekent meer overeenkomst met de criteria en een meer geschikte tool.

Criteria			**Software opties**					
			D3JS		**FlowchartJS**		**JointJS**	
ID	**Naam**	**Weging**	**Score**	**Resultaat**	**Score**	**Resultaat**	**Score**	**Resultaat**
D	Aanpasbaarheid tekentool	30%	-0.5	-15%	-1.0	-30%	1.0	30%
E	Aanwezigheid tekenfuncties	30%	1.0	30%	0.5	15%	0.5	15%
F	Uitleesbaarheid diagram	15%	1.0	15%	-1.0	-15%	1.0	15%
G	Open Source	15%	1.0	15%	1.0	15%	1.0	15%
H	Documentatie	5%	1.0	5%	0.0	0%	0.5	2.5%
I	Ondersteuning	5%	1.0	5%	0.5	2.5%	1.0	5%
	Totaal:	**100%**		**55%**		**-12.5%**		**82.5%**

	Software opties							
	JSPlumb CE		Mermaid		MxGraph		Raphael	
ID	**Score**	**Resultaat**	**Score**	**Resultaat**	**Score**	**Resultaat**	**Score**	**Resultaat**
D	-0.5	-15%	-1.0	-30%	0.5	15%	1.0	30%
E	1.0	30%	0.5	15%	0.5	15%	0.5	15%
F	1.0	15%	-1.0	-15%	1.0	15%	0.0	0%
G	1.0	15%	1.0	15%	1.0	15%	1.0	15%
H	1.0	5%	0.5	2.5%	1.0	5%	1.0	5%
I	1.0	5%	0.5	2.5%	1.0	5%	0.5	2.5%
		55%		**-10%**		**70%**		**67.5%**

Bevindingen

Uit bovenstaande tabellen is op te maken dat de tekentool JointJS het hoogste scoorde. Deze tool werd echter vrij nauw gevolgd door MxGraph en Raphael en daarom was het verstandig deze nog extra te evalueren.
De tools D3JS en JSPlumb CE scoorden redelijk maar waren te afgesloten geprogrammeerd, waardoor ondersteuning voor de informatiestructuur van workflows lastig werd. De tools FlowchartJS en Mermaid eindigden op een negatieve score. Diagrammen werden in deze tools met de zogenaamde "markdown" taal opgezet, wat betekende dat er geen toegang was tot de objectstructuur van de getekende elementen.

5.5 Resultaat onderzoek

De diagram-tekentool die middels dit onderzoek was gekozen is JointJS. Dit is een open source Javascript library met functies om verschillende soorten diagrammen te tekenen, waaronder flowcharts. Deze tool is als basis gebruikt voor de verdere ontwikkeling van de workflow-configuratietool. Alle tekentools in de shortlist zijn getest om te zien welke functionaliteit en uitbreidingsmogelijkheden zij boden en JointJS kwam hierbij naar voren als de tool met het meeste potentieel.
Verschillende tools boden enkel een manier om figuren in een diagram te tekenen, maar bevatten geen functionaliteit om deze te verslepen, toe te voegen, of anderzijds aan te passen. In theorie was het mogelijk om deze functionaliteit zelf te programmeren, maar JointJS beschikte al over deze functionaliteit.
Andere tools die dit ook konden, hadden weer als nadeel dat zij niet genoeg aangepast konden worden. Deze tools waren gemaakt om als kant-en-klaar product in een website geïmplementeerd te worden. JointJS was echter compleet open source en beschikte over verschillende opties om zelfgemaakte elementen aan het diagram toe te voegen die nog steeds van de beschikbare JointJS functionaliteit gebruik kunnen maken.
Van alle geëvalueerde tools uit de shortlist, had JointJS de beste balans tussen de beschikbare functionaliteit en de aanpassingsmogelijkheden voor verdere ontwikkelingen. De documentatie van JointJS was voldoende en er bestond een actieve community die de tool ontwikkelde en ondersteuning leverde.
Een ander interessant aspect van JointJS was dat er ook een betaalde variant van bestaat, genaamd Rappid. Rappid is in wezen een functioneel uitbreidingspakket op de gratis JointJS, welke functionaliteit biedt voor onder andere wijzigingsmanagement en keyboard- afhandeling. Mocht ooit de workflow-configuratietool in gebruik worden genomen, kan worden besloten om de Rappid software aan te schaffen zodat minder tijd hoeft worden besteed aan verdere ontwikkelingen van de tool.

6. Realisatie

Inleiding

In week vijf van het afstudeertraject kon het ontwikkeltraject daadwerkelijk beginnen. De realisatie van de workflow-configuratietool gebeurde in sprints van één week. Elke sprint werd een nieuw onderdeel van de tool opgeleverd en gereviewd, en op basis daarvan werd bepaald wat de daaropvolgende sprint werd ontwikkeld. In dit hoofdstuk zal van elke sprint worden beschreven welke keuzes zijn gemaakt en wat op basis van die keuzes is ontwikkeld.

6.1 Sprint Eén - Initiatie

In de eerste sprint van het project zijn voornamelijk voorbereidingen gedaan op de ontwikkeling van de workflow-configuratietool. De Backlog moest nog worden opgezet en er moest nog software worden gekozen om de unit-tests mee uit te voeren. Ook zijn in deze sprint ontwerpen gemaakt van de workflow-configuratietool over de opzet van de informatiestructuur en de layout van de tool.

User Stories

Voor het opzetten van de Backlog is enkel naar de functionele requirements gekeken. Deze requirements representeerden namelijk de fundamentele functionele aspecten van de workflow-configuratietool en waren specifiek genoeg om als Backlog item te worden gebruikt.
De Sprint Backlog werd opgezet in de applicatie JIRA, welke de ontwikkelaars van Nétive al langere tijd gebruiken voor het beheer van Scrum. In deze applicatie heb ik User Stories gemaakt van de functionele requirements, met voor elke Story een titel, beschrijving, en acceptatiecriteria. Ik en mijn bedrijfsmentor hebben de Backlog gereviewd en hebben aan elke User Story een hoeveelheid Story Points toegekend om de relatieve hoeveelheid benodigd werk aan te geven. Ook is een initiële prioritering gemaakt van alle User Stories, zodat deze direct konden worden opgepakt in de tweede sprint.

Test Framework

Er was gepland om de workflow-configuratietool te testen met unit-tests. Deze tests moesten individuele software-functies testen en daarom was het nodig deze geautomatiseerd met behulp van een unit-test framework uit te voeren. In deze sprint moest daarom een unit-test framework worden gekozen waarmee unit-tests konden worden uitgevoerd. De keuze van dit framework was niet zeer significant. Het enige belang was dat er herbruikerbaar tests konden worden opgezet die snel uitgevoerd konden worden.
Er is gekozen om gebruik te maken van het programma QUnit. Dit is een op Javascript gebaseerde library van de makers van JQuery, waarmee Javascript functies geautomatiseerd kunnen worden getest.

Ontwerpen

Aan de hand van de opgestelde requirements en de gekozen diagram-tekentool konden ontwerpen worden gemaakt over de opzet van de workflow-configuratietool. Deze ontwerpen bestonden uit een diagram met daarin de informatie- en componentenstructuur van de configuratietool, en een layout van de pagina zelf.

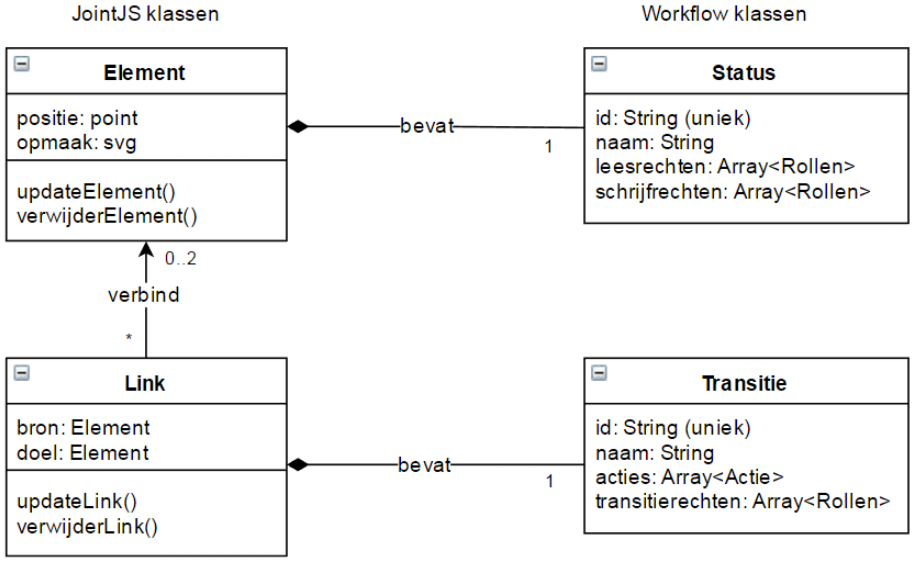
Klassendiagram

Aangezien de workflow-configuratietool is opgezet in Javascript, waren er geen strikte regels over de datatypes. Desondanks is voor elk workflow-object een "class" opgezet met verplichte variabelen en functies. JointJS bevatte zelf ook enkele klassen met daarin functies om deze te tekenen binnen het canvas op de webpagina.

In Figuur 14 is aangegeven hoe de klassen van JointJS zich verhouden tot de status en transitie klassen van de workflow. Er zijn meer klassen opgenomen in de configuratietool, zoals de eerdergenoemde acties, condities, en gebruikersrollen, maar die zijn allemaal ondergeschikt aan onderstaande klassen.

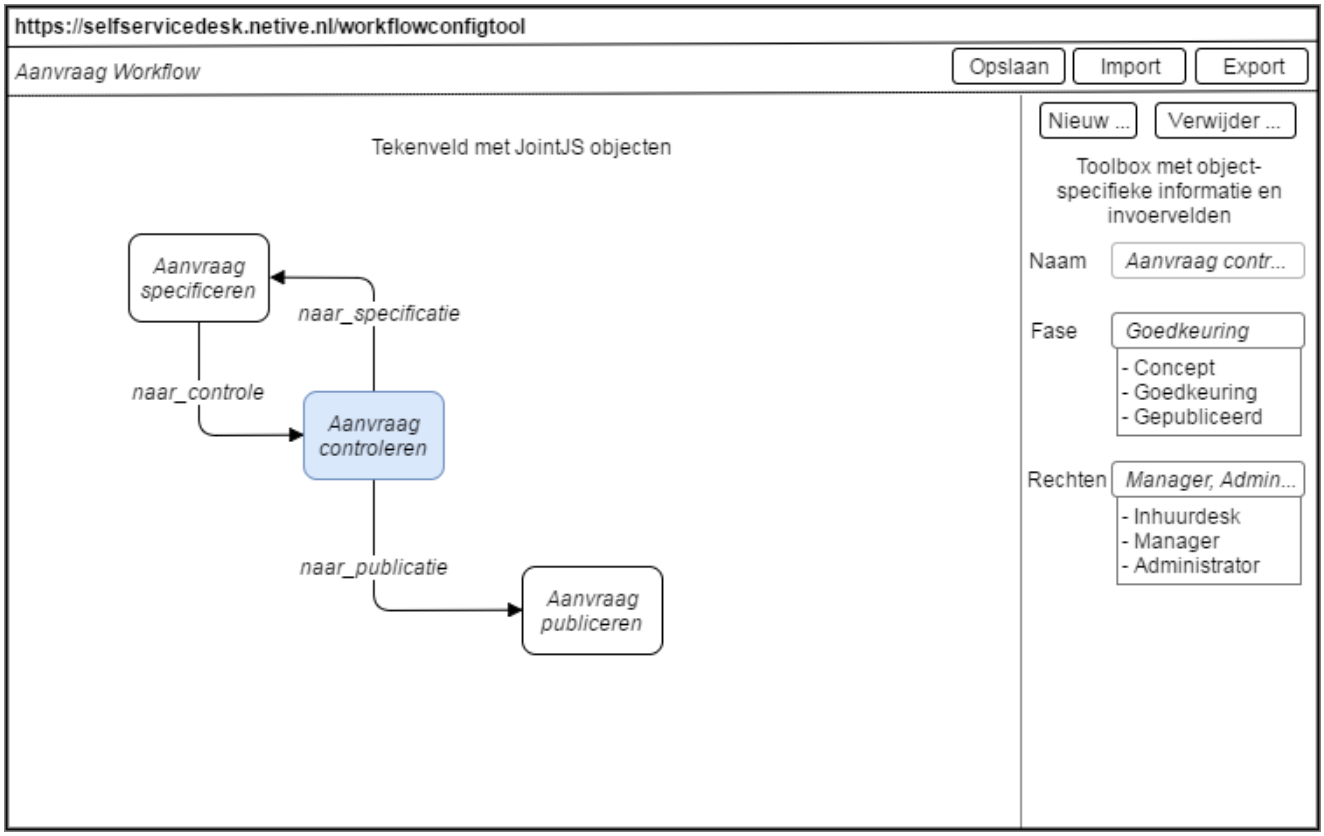
De Element en Link klassen waren van JointJS en moesten worden aangepast om Status en Transitie objecten op te kunnen slaan. De Element klasse is een object dat op het scherm wordt getekend, met een opmaak en positie aangegeven door de gebruiker. De Link klasse is een object op het scherm in de vorm van een lijn die twee Elementen verbindt.

De Status en Transitie klassen waren enkel modellen voor de data die de gebruiker in moest voeren. Hierin stonden onder andere de naam van het object en de rechten voor het gebruik in het VMS. De Status en Transitie klassen waren ook een compositie met hun bijbehorende JointJS objecten. Dit geeft aan dat er zonder Elementen en Links geen Statussen of Transities konden bestaan in het systeem. Deze structuur was gekozen omdat de gebruiker een visueel element nodig had om op te klikken zodat hij de informatie in de workflow klassen kon aanpassen. Als er geen visueel element beschikbaar was, kon de gebruiker de status of transitie dus niet meer aanpassen.



Figuur 14: Klassendiagram Workflow-objecten

Website Mockup
Van de workflow-configuratietool is ook een mockup gemaakt. De tool zelf moest uit één pagina bestaan dus het was vooral een kwestie van bepalen waar de verschillende knoppen moesten komen in verhouding tot het tekenveld.
In Figuur 15 is de gemaakte mockup te zien. In dit diagram is de workflow aangehouden van het voorbeeld in hoofdstuk 2.2. Aan de rechterkant van het figuur is een toolbox te zien met daarin informatie over de geselecteerde status of transitie, in dit geval de status "Aanvraag controleren." In de toolbox heeft de naam een vrij invoerveld, de fase een enkelvoudige keuzelijst, en de rechten een meervoudige keuzelijst. In de praktijk moesten er nog meer invoervelden komen voor statussen en transities, maar die waren verder niet van invloed op de layout van de pagina.



Figuur 15: Web Mockup van de workflow-configuratietool

6.2 Sprint Twee – Tekenen

In sprint twee is het daadwerkelijke programmeren van de workflow-configuratietool van start gegaan. De JointJS testapplicatie die is geschreven voor het software-selectie onderzoek is hiervoor deels overgenomen.
Sprint twee is begonnen met de eerste officiële sprint planning. Hierin hebben ik en de Scrum master gekeken naar de Backlog en ontwerpen die ik had opgesteld. Ik had besloten om bovenaan de Backlog te beginnen, aangezien er geen reden was de prioritering nu al te wijzigen. Dit waren de User Stories over het toevoegen en bewerken van statussen en transities in de configuratietool.

Layout van de applicatie

Om de workflow-configuratietool eruit te laten zien als in Figuur 15, moest ook een CSS-bestand worden geschreven. Salesforce pagina's worden normaal geleverd met een voorgemaakte structuur volgens de Salesforce-standaarden, maar deze voorzag niet in de layout die ik wilde maken. Salesforce bood gelukkig de optie om handmatig de CSS van de pagina te schrijven en dus heb ik voor deze optie gekozen.

Tekenen van interactieve objecten

JointJS bood kant-en-klare functies om objecten te tekenen op een webpagina. Om een werkend canvas te creëren hoefde enkel de naam van een html-container en een hoogte en breedte mee te worden geven.

Door voorgemaakte JointJS-objecten te instantiëren, konden deze vervolgens worden toegevoegd aan het canvas. Aan deze objecten konden attributen worden meegegeven, onder andere over de opmaak en positie daarvan. Deze opmaak was in SVG-formaat en de positie in de vorm van een x- en y-coördinaat op het canvas.

Een van de eisen was om de naam van een status in het visuele object te laten zien. JointJS ondersteunde dit door programmeurs de optie te geven zelfgemaakte HTML-code aan de objecten toe te voegen. Vervolgens kon, met behulp van de JQuery library, de naam van elke status in het bijbehorende visuele object worden geplaats.

Bewerken van getekende objecten

Om de JointJS objecten interactief te maken is gebruik gemaakt van zogenaamde "events." Het event die ik heb gebruikt is de "on click" event voor de Elementen en Links. Hiermee wordt, wanneer de gebruiker op een JointJS object klikt, een event afgevuurd waarmee door mij geschreven code kan worden uitgevoerd.

Ik heb een functie geschreven die met dit event wordt aangeroepen, waarmee de informatie van het geklikte object in de toolbox terecht komt. Deze informatie kan vervolgens door de gebruiker worden bewerkt en wordt vervolgens automatisch opgeslagen in het status- of transitie-object. Voor lege variabele wordt ook een invoerveld getoond, en kan de gebruiker zelf een waarde invoeren.

Unit-tests

In deze sprint waren voornamelijk CRUD-functies gebouwd, wat goede kandidaten waren om unit-tests op uit te voeren. Voor elk van deze functies is een unit-test opgezet en gedocumenteerd. Voor het bepalen van de testcriteria is gekeken naar de requirements en de acceptatiecriteria van de bijbehorende User Story. Hierin stond namelijk beschreven hoe deze functies zouden moeten werken, waaruit kon worden opgemaakt welke testcases ik moest maken en welke data ingevoerd zou moeten worden in het systeem.

Voor elke unit-test zijn twee verzamelingen testdata opgezet, een om in te voeren in de functie, en een om de resultaten daarvan te vergelijken met de wensen volgens de requirements. In de invoerdata was ook rekening gehouden met de mogelijkheid voor niet-valide of missende data.

De tests zelf zijn geprogrammeerd met behulp van QUnit functies. QUnit bood een voorgeschreven HTML-pagina die bij het openen automatisch alle opgezette unit-tests uitvoerde. Hier werd ook aangegeven welke tests goed of fout gingen en wat de verschillen waren tussen de resulterende en verwachte data.

Acceptatietests

Tot slot zijn er deze sprint enkele acceptatietests opgezet. Met deze black-box tests kon worden achterhaald of de acceptatiecriteria van de User Stories volledig zijn gerealiseerd. Voor elke User Story is een testcase opgezet met daarin beschreven wat de acceptatiecriteria waren, welke stappen de tester moest doorlopen om deze te verifiëren, en wat de verwachte testresultaten waren. Ook werden waar nodig precondities gegeven over de aanwezigheid en werking van bepaalde onderdelen in de applicatie.

Deze unit- en acceptatietests zijn stuk voor stuk door mij uitgevoerd, en waar nodig zijn wijzigingen gemaakt in het systeem om de fouten op te lossen. Alle tests zijn gedocumenteerd en terug te vinden in het testrapport.

Review

Aan het einde van de sprint hebben ik en de bedrijfsmentor de software en werkzaamheden gereviewd. Een van de dingen die hier naar voren kwam was dat de getekende workflows erg onoverzichtelijk werden als er een groot aantal statussen en transities in voorkwamen. Ik had namelijk een workflow getekend met daarin 12 statussen en 44 transities, afgeleid uit een workflow-bestand die de bedrijfsmentor mij had gegeven. In het diagram liepen veel lijnen door elkaar heen en er kon niet goed worden afgelezen hoe de workflow precies doorlopen kon worden.
Om deze reden was besloten de focus van het project te verschuiven. De focus lag op dat moment nog vooral op het kunnen invoeren van alle mogelijke informatie die in een workflow kan worden opgenomen, maar aan de hand van dit nieuwe inzicht werd duidelijk dat nu de aandacht moest worden gelegd op het overzichtelijk maken van de workflow-diagrammen. Hoe dit werd opgelost wordt in de planning van sprint drie behandeld.

6.3 Sprint Drie – Overzichtelijkheid

In de planning van sprint drie heb ik besloten om de aandacht te leggen bij het overzichtelijk maken van de workflow-configuratietool. Deze was op dat moment niet overzichtelijk en daardoor minder gebruiksvriendelijk. Gebruiksvriendelijkheid was een belangrijk risicogebied en daarom is gekozen dit probleem meteen te adresseren. Verder is ook functionaliteit ingebouwd waarmee de richting van een transitie kan worden omgewisseld.

Minder Links

De eerste stap die is gezet om dit op te lossen was het reduceren van het aantal lijnen op het scherm. Dit was mogelijk omdat er soms meerdere transities waren tussen twee statussen die elk een eigen link hadden. Dit kwam voor wanneer tussen twee statussen heen en terug kon worden gestapt, of wanneer simpelweg transities met verschillende acties of rechten plaats konden vinden.
De oplossing was om achter één link in het diagram meerdere transities te plaatsen. Zo hoeft er maar één lijn te worden getekend tussen twee statussen, ook als de gebruiker daar meerdere transities tussen moet vastleggen.
Deze functie is een nieuw item geworden in de Backlog, en heb ik in deze sprint opgepakt, gerealiseerd en getest. Deze ontwikkeling bracht ook wijzigingen met zich mee voor het klassendiagram. Waar een Link namelijk eerst altijd één Transitie had, kon het er nu een of meer hebben.
Met deze ontwikkeling is ook een functie toegevoegd waarmee gebruikers individuele transities uit een link kunnen verwijderen, in plaats van meteen de gehele link te verwijderen.

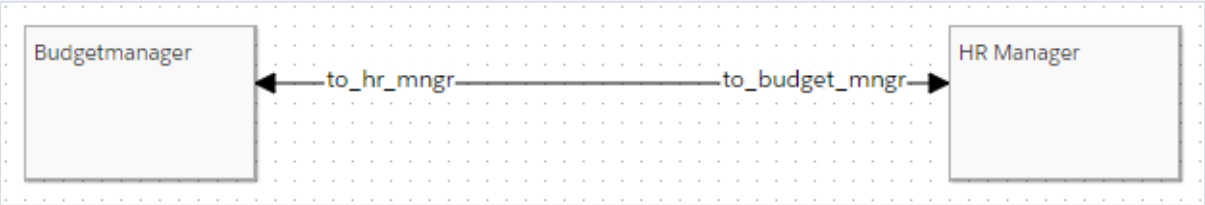
Minder Elementen

De tweede oplossing was om het aantal elementen op het scherm te reduceren. In eerdergenoemde voorbeelden kwam de status “publiceer aanvraag” naar voren. Dit is een voorbeeld van een status waarin de informatie in de workflow (in dit geval de aanvraag) naar de backend van het VMS wordt gestuurd. Dit publicatieproces bevat is een technisch proces met meerdere statussen waar geen bedrijfsmedewerkers bij aan de pas komen. Het valt daarom ook buiten het kennisgebied van de eindgebruiker.
Om de workflow-configuratietool overzichtelijker en begrijpelijker te maken zouden deze statussen daarom uit de tool kunnen worden gehaald en via een geautomatiseerde functie aan de XML van de workflow worden toegevoegd. Dit was mogelijk omdat deze backend processen over alle workflows bijna geheel hetzelfde zijn. Het deel dat niet overal overeenkwam waren de acties die bij een transitie in het backend proces plaatsvonden, zoals de mails naar verschillende VMS-gebruikers. Hier was echter een oplossing voor gevonden door het vastleggen van acties ook uit de workflow-configuratietool zelf te halen en in een apart dialoog te zetten.
Voor het realiseren van deze functionaliteit zijn enkele nieuwe User Stories gemaakt in de Backlog maar deze waren nog niet in de sprint opgepakt. Pas als wij echte valide workflows wilden gaan exporteren, zou deze functionaliteit van belang worden. Tot die tijd was het voldoende om

simpelweg niet deze statussen te tekenen in de tool, omdat dan alsnog inzicht kan worden verkregen in hoe overzichtelijk de diagrammen worden voor de gebruiker.

Richting Transitie

Omdat gebruikers nu meerdere transities aan één link konden toevoegen, was het nodig hen ook de optie te geven de richting van een transitie aan te passen. In Figuur 16 is te zien hoe een dergelijke link er in de workflow-configuratietool uit zou gaan zien.
Aan de frontend hoefde hiervoor enkel een knop te worden toegevoegd die de optie gaf de richting om te wisselen, maar aan de backend moesten grotere wijzigingen plaatsvinden. Omdat er eerst één transitie per link werd opgeslagen, stond de informatie over de bron- en doel-statussen alleen in de link zelf opgeslagen. Om de richting van een transitie onafhankelijk van de link om te kunnen draaien, moet deze informatie ook in de transities beschikbaar zijn en gewijzigd kunnen worden.
Dit bracht echter een extra probleem met zich mee wanneer een link werd losgekoppeld. Als een link van zijn doel-status wordt losgekoppeld, worden omgewisselde transities in de link juist van hun bron-status losgekoppeld. De link had echter geen kennis over de richting van zijn transities en daardoor was het lastiger te bepalen of een transitie aan de kant van de bron- of doel-status werd losgekoppeld.
Om dit op te lossen is een nieuwe variabele toegevoegd aan het transitie-object om aan te geven welke richting het op ging. Zo kon ook makkelijker worden bekeken aan welke kant van de link een pijltje moest worden getekend.



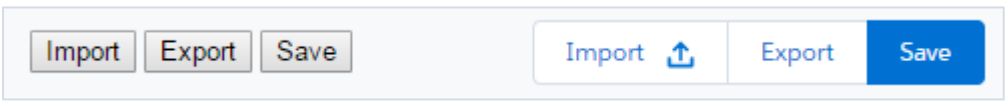
Figuur 16: Twee statussen met twee transities op één link

6.4 Sprint Vier – Styling & Slimme Links

Nu de workflow-configuratietool meer overzichtelijk was, heb ik gekozen om mij te gaan focussen op de styling van de tool. De gebruiker had al verschillende opties om diagrammen te tekenen en data in te voeren, maar het was ook van belang dat deze er netjes en consistent uit zagen. Ook zijn poorten toegevoegd aan de JointJS elementen, waarmee de gebruiker links op verschillende punten aan het object kan koppelen.

Salesforce-styling

Nétive houdt in het VMS de standaard Salesforce opmaak aan. Om deze opmaak in websites met zelfgemaakte HTML te kunnen implementeren heeft Salesforce het zogenaamde "Lightning Design System" opgezet (SLDS; https://www.lightningdesignsystem.com/). In dit gratis pakket, waar Nétive in het SSD ook gebruik van wil maken, is alle CSS en SVG-pictogrammen gebruikt in Salesforce beschikbaar gesteld voor het publiek.
In Figuur 17 is het verschil te zien tussen drie normale HTML-knoppen en dezelfde knoppen opgemaakt met SLDS-klassen. In de HTML zijn dit nog steeds button-elementen, maar met de CSS en pictogrammen van Salesforce zien ze er een stuk beter uit. SLDS geeft knoppen ook geen extra functionaliteit; het is puur een verzameling aan CSS-klassen die, waar nodig op basis van Javascript functies, aan bestaande HTML-elementen kan worden toegevoegd.



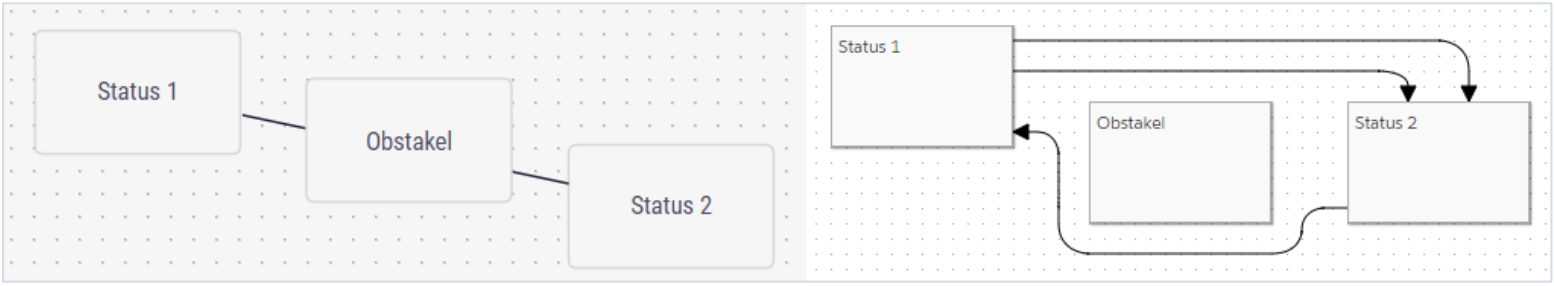
Figuur 17: Standaard HTML-knoppen (links); knoppen met SLDS (rechts)

De SLDS-klassen bestonden uit meer dan alleen styling voor knoppen. Ook formulieren, menu's, pop-ups, en andere elementen in de workflow-configuratietool zijn opgemaakt met het SLDS (zie Figuur 18). Hiervoor moesten echter wel enkele wijzigingen in de HTML-code plaatsvinden. Waar bijvoorbeeld voor een tekst-inputveld eerst alleen een input- en label-element nodig was, moesten daar nu ook verschillende div-elementen omheen worden geplaatst om aan SLDS duidelijk te maken dat er een label-input-combinatie aanwezig was.

Figuur 18: SLDS-styling voor verschillende input-elementen

Poorten & algoritmes

Voor het bestaan van poorten waren JointJS links altijd direct met de element-objecten verbonden. Dit betekende dat de X-Y-positie van het begin en eind van een link precies in het midden van de element-container zat, zoals te zien in Figuur 19 aan de linkerzijde. Hierdoor kon niet worden gezien of er meerdere lijnen tussen twee statussen liepen, omdat deze over elkaar heen vielen. Ook liepen lijnen recht door andere objecten heen.
De oplossing die JointJS hiervoor bood was het gebruik van poorten en zoekalgoritmes voor de positionering van links. De poorten konden aan elementen worden toegevoegd om connectiepunten te creëren voor de links. In Figuur 19 aan de rechterkant bij Status 2 kan worden gezien hoe twee links naast elkaar aan het element kunnen worden geplaatst. Hierdoor kan de gebruiker overlappingen in de lijnen voorkomen.
In JointJS zit een zoekalgoritme ingebouwd die automatisch de kortste route berekent voor lijnen tussen hun begin- en eindpunt. Hierdoor worden ook obstakels in het pad van de lijn ontweken. Dit algoritme is toegepast in de workflow-configuratietool, terug te zien in Figuur 19 aan de rechterkant bij de lijn van Status 2 naar Status 1, waar de lijn om het obstakel heen loopt.



Figuur 19: JointJS-links zonder (links) en met (rechts) zoekalgoritmes en poorten

Review

De algoritmes, toegepast in deze sprint, maakten de applicatie soms wat trager, voornamelijk wanneer het algoritme op meerdere lijnen tegelijkertijd moest worden toegepast, bijvoorbeeld als een status met veel lijnen daaraan gekoppeld door het tekenveld werd verschoven. Ik had daarom een nieuw Backlog item toegevoegd om dit probleem aan te pakken. De bedrijfsmentor en ik vonden dit probleem nog niet ernstig genoeg om deze in de volgende sprint mee te nemen, dus is de prioritering van de Backlog niet veranderd.

6.5 Sprint Vijf - Opslag

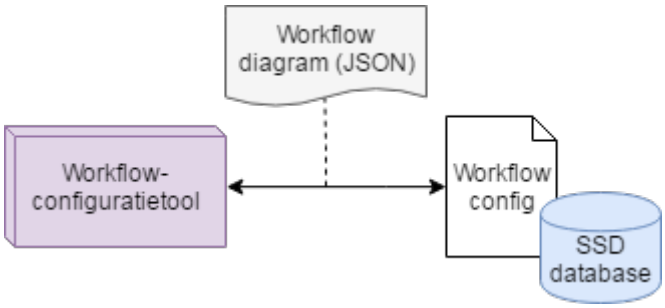
Dankzij de ontwikkeling in de afgelopen twee sprints was de workflow-configuratietool gebruiksvriendelijk en zag het er overzichtelijk uit. Ik wilde graag feedback krijgen van andere medewerkers van Nétive en daarom heb ik een demonstratie gegeven van de tool. Ik heb deze sprint ook functionaliteit geschreven waarmee workflows kunnen worden opgeslagen en opgehaald uit de Salesforce database. Hiermee kunnen gebruikers hun werk tussentijds opslaan zonder deze direct naar een VMS-omgeving te hoeven exporteren.

Demonstratie

Tijdens de Sprint Review van de ontwikkelteams van Nétive geven zij altijd presentatie over het werk dat zij hebben opgeleverd aan alle andere medewerkers. Ik heb van deze gelegenheid gebruik gemaakt om zelf ook een presentatie te geven.
Hierin heb ik kort uitgelegd wat ik tot op dat moment had gedaan en heb ik vervolgens een demo gegeven van de workflow-configuratietool. Ik heb laten zien hoe een status kan worden getekend en hoe daar een transitie tussen kan worden getrokken, welke informatie de gebruikers in de tool kunnen invoeren, en een voorbeeld gegeven van hoe een volwaardige workflow er in de configuratietool uit zou zien.
Na afloop van de presentatie was er de gelegenheid voor vragen. Veel vragen konden worden beantwoord met "Dit staat al in de Backlog en willen we in de toekomst in de tool gaan verwerken," maar een paar mensen hadden ook ideeën waar ik nog niet eerder over had nagedacht. Een van de wensen die hier naar voren kwam was om de objecten in het diagram kleuren te kunnen geven, zodat daar beter onderscheid tussen kan worden gemaakt.
Een andere wens kwam van een bedrijfsmanager, die aankaartte dat een van de klanten van Nétive het belangrijk vond om te kunnen zien welke wijzigingen er in de loop van de tijd plaatsvinden in de workflow-configuratie. Dit zou betekenen dat elke informatiewijziging die de gebruiker uitvoert, moet worden opgeslagen. Een oplossing die een ontwikkelaar naar voren bracht was het gebruik van het Event Sourcing design pattern.
Deze suggesties zijn in de Backlog opgenomen en er is besloten in de planning van sprint zes te kijken in hoeverre ik deze op zou gaan pakken.

Workflow opslag

Er waren twee manier om een workflow op te slaan: geconverteerd naar het XML-formaat gebruikt in het VMS, met daarin enkel de informatie over statussen en transities; en onveranderd in een nieuw formaat, met daarin ook alle informatie over de positionering en het uiterlijk van de getekende JointJS elementen. In deze sprint heb ik mij gericht op de tweede methode: het opslaan van het workflow-diagram zoals beschreven in Figuur 20.



Figuur 20: Dataflow Workflow-object (deel 1)

Tot op dit punt had ik bijna alleen gewerkt aan de frontend van de webapplicatie, maar nu moest ik ook gebruik gaan maken van de Salesforce backend. Ik heb daarin een Apex Controller geprogrammeerd, welke vanuit mijn Javascript-code kan worden aangeroepen, met daarin functies die met de Salesforce database verbinding maken.
Om een diagram getekend in de configuratietool op te kunnen slaan, heb ik een van de uitleesfuncties van JointJS gebruikt waarmee de gehele informatiestructuur kon worden omgezet naar JSON-formaat. Dit JSON-object kon vervolgens worden geconverteerd naar een string, welke aan de Apex Controller kon worden doorgegeven die het vervolgens opsloeg in de Salesforce database. Dit proces was in principe niet heel complex; de meeste tijd ging hier zitten in het leren gebruiken van de Salesforce backend en de communicatie daartussen met Javascript.
Nu de workflow-diagrammen konden worden opgeslagen, moest er ook functionaliteit gemaakt worden om deze uit de database op te halen en in de configuratietool te openen. Hiervoor is ook een functie gemaakt in de Apex Controller, die de JSON-string ophaalde en doorgaf aan de Javascript runtime. Middels een JointJS functie kon de JSON direct worden omgezet naar Element en Link objecten, die direct aan het diagram konden worden toegevoegd.

Unit-tests

Met het uitvoeren van de unit-tests bleek dat er iets fout ging in de conversie van JSON-string naar workflow-diagram. Wat er namelijk gebeurde was dat de typering van de Status en Transitie Javascript-klassen verloren ging bij de conversie naar JSON. Dit werden bij de conversie primitieve Javascript-objecten en bevatten daardoor niet de functies die normaal in een Status- of Transitie-object zitten. Om dit op te lossen moesten deze primitieve objecten codematig worden omgezet naar de workflow-objecten en weer aan hun bijbehorende Elementen en Links worden toegevoegd.

6.6 Sprint Zes - Beheer

In sprint vijf is functionaliteit gemaakt om workflow-diagrammen op te slaan, alleen er bestond nog geen menu waar de gebruikers hun opgeslagen workflows in konden selecteren. Hierin moest het ook mogelijk worden om nieuw workflow-configuraties aan te maken en te benoemen. Dit was de eerste geplande taak voor deze sprint. Daarnaast is ook besloten de suggestie voor het kunnen kleuren van objecten te ontwikkelen. Er is geëxperimenteerd met het toepassen van de Event Sourcing design pattern maar dit is niet serieus opgepakt.

Workflow menu

Voor het selectiemenu is een nieuwe Salesforce pagina opgezet naast die van de workflow-configuratietool. De workflow-configuratietool is een handgemaakte pagina, maar Salesforce biedt ook pagina's met voorgemaakte componenten. Een van deze componenten is die voor een lijst met objecten van één specifiek type, welke ik in deze pagina heb gebruikt. Dit component bevatte ook knoppen waarmee objecten van dat type konden worden toegevoegd of aangepast.
Ik heb in dit component zelf een knop toegevoegd waarmee de gebruiker, nadat hij een workflow heeft geopend, naar de workflow-configuratietool wordt doorgeleid. Bij het openen van de tool wordt ook meteen de opgeslagen JSON-string opgehaald (als die bestaat) en ingeladen in het tekenveld.

6.7 Sprint Zeven – Conversie naar XML

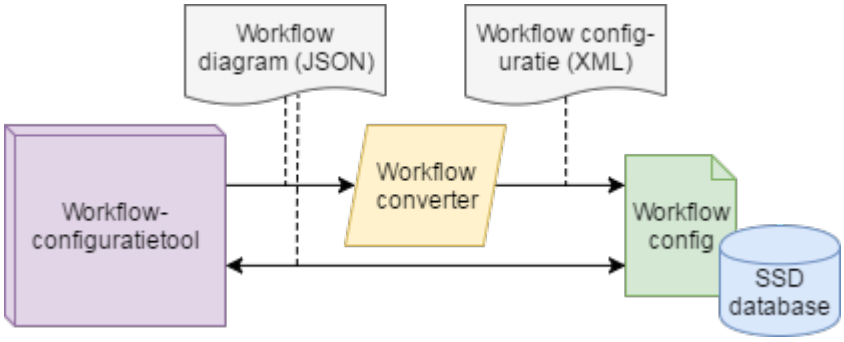
Het was inmiddels mogelijk om de belangrijkste informatie van workflows vast te leggen met de configuratietool. Aangezien het einde van het project naderde, wilde ik mij richten op het creëren en opslaan van XML-bestanden, omgezet uit workflow-diagrammen. Deze XML-bestanden konden vervolgens in een later stadium naar een VMS-omgeving worden geëxporteerd, maar dit viel buiten de scope van het project.

Informatiestructuur omzetten

De objecten in het workflow-diagram die in de XML moeten worden opgeslagen zijn de statussen en transities. In de tool staan statussen en transities los van elkaar opgeslagen, maar in de XML moet een transitie worden opgeslagen in de status van waaruit deze begint. Ook de acties en rechten waren aparte objecten, waarvan enkel een referentie in de statussen en transities werd opgeslagen. Deze moesten echter ook volledig in de statussen en transities terugkomen.
Om deze conversie uit te voeren heb ik een Apex klasse gemaakt waarin de informatiestructuur van het workflow-diagram wordt omgezet naar de structuur van de XML.

XML-bestand schrijven

Nu de informatiestructuur van de workflow correct was, kon het schrijven van het XML-bestand plaatsvinden. Hiervoor is gebruik gemaakt van de Apex klasse XMLStreamWriter, waarmee codematig een XML String kan worden opgebouwd. Om de XML te schrijven moest door de status- en transitie-objecten worden geïtereerd, met elk object genest in de XML van zijn ouder-object.
Het XML-bestand werd vervolgens als string opgeslagen in het workflow-object, zoals aangegeven in Figuur 21.



Figuur 21: Dataflow Workflow-object (deel 2)

6.8 Sprint Acht – Importeren van XML

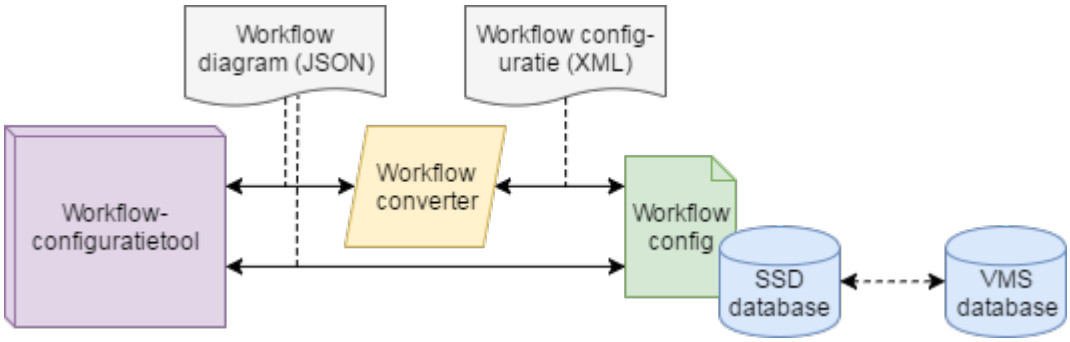
In de laatste sprint van het afstudeerproject heb ik functionaliteit gemaakt om een XML-bestand van een workflow om te zetten naar een diagram in de configuratietool. Deze functionaliteit zou alleen gebruikt worden om bestaande workflows te importeren; workflows die van de grond af met de configuratietool zijn gemaakt zullen deze functie niet gebruiken.

XML naar JSON

Om de XML te converteren naar objecten in de backend van Salesforce, is gebruik gemaakt van de XMLStreamReader. Hiermee konden alle elementen en attributen van de XML worden ingelezen, waarna ik ze handmatig een de workflow-objecten kon toevoegen. Deze objecten werden vervolgens aan de Javascript runtime doorgegeven, waar ze via een JointJS functie tot een diagram werden getekend.

Conclusie

Na het opzetten van de workflow-configuratietool, de workflows in de SSD en de conversie laag daartussen, ziet het systeem eruit als in Figuur 22. In de database staan workflow-configuraties opgeslagen, met daarin een JSON-bestand en een XML-bestand



Figuur 22: Dataflow Workflow-object (deel 3)

7. Evaluatie

Inleiding

In dit hoofdstuk zal ik een evaluatie geven van het gehele afstudeertraject. Hierin zal ik een beoordeling geven van de opgeleverde producten, de uitvoering ten opzichte van de geplande aanpak, en het niveau waarop de beroepstaken zijn uitgevoerd.

7.1 Opgeleverde producten

Workflow-configuratietool

Om te beginnen kan ik zeggen dat ik persoonlijk trots ben op de workflow-configuratietool die ik heb ontwikkeld. De voornaamste reden hiervoor is dat mijn collega's bij Nétive erg enthousiast leken te zijn over het product. Er kwamen regelmatig mensen naar mij toe die vroegen of ze de tool al konden gebruiken of die om een kleine demo vroegen. Mede dankzij het interne succes van de workflow-configuratietool hebben de ontwikkelaars besloten de Self Service Desk als een serieus item in de Sprints op te nemen en ik denk dat dat er wel voor mag spreken dat Nétive echt waarde hecht aan de applicatie die ik heb ontwikkeld.
Ik vind de applicatie zelf ook mooi vormgegeven en gebruiksvriendelijk. Het design is erg strak, zoals te zien in hoofdstuk 6.4, en alle invoervelden en functies zijn met één druk op de knop toegankelijk. De applicatie presteert ook goed, en het tekenen en configureren van workflows gaat vrij vlot. Dankzij de creatieve vrijheid die ik had en de vele uitdagingen die het project met zich mee bracht, vond ik het ook een leuk proces om de workflow-configuratietool te ontwikkelen.

Productdoelen

Het product heeft de opgestelde doelen in het afstudeerplan voor het grootste deel kunnen vervullen. Er was al verwacht dat niet alle denkbare requirements binnen het afstudeertraject konden worden volbracht. Daarom was gekozen om in eerste instantie te focussen op de belangrijkste functionaliteit: het tekenen en configureren van workflows, en het opslaan, inlezen en exporteren daarvan.
Deze doelen zijn behaald en voldoen aan de wensen die de opdrachtgever had opgesteld. Volgens mijn bedrijfsmentor gaat op korte termijn de ontwikkeling van de configuratietool verder worden opgepakt als deel van het SSD-project om de resterende requirements te vervullen.

Code

Naar mijn mening is het product codematig ook goed opgezet. Ik heb met de bedrijfsmentor tijdens de sprint reviews regelmatig naar mijn code gekeken en hij vond dat het goed herbruikbaar en uitbreidbaar was opgezet. Ik heb in de ontwikkeling gebruik gemaakt van nieuwe programmeertalen, Visualforce en Apex, waarmee ik succesvol met de Salesforce database heb kunnen communiceren. Ook is mijn kennis van Javascript wezenlijk uitgebreid in dit project, mede dankzij het vele gebruik van JQuery en JointJS.

Requirements

De requirements, opgesteld in week twee van het project en gedurende de sprints verder uitgebreid, hebben mij veel geholpen bij het ontwikkelen van de workflow-configuratietool. Bij de aanvang van het project wist Nétive eigenlijk alleen nog maar dát ze een tool wilden waar workflows in konden worden geconfigureerd. Hoe deze tool eruit moest gaan zien en hoe deze door de gebruiker moest worden gebruikt, was aan mij overgelaten.
Door de requirements op te stellen en te reviewen met de stakeholders, kon ik voor mijzelf en de stakeholders een duidelijk beeld creëren over hoe de tool eruit moest gaan zien. Deze requirements heb ik gedurende het gehele project veelvoudig gebruikt en er waren vrijwel geen wijzigingen in nodig later in het project, afgezien van enkele aanvullingen.

7.2 Gebruikte Aanpak

In het plan van aanpak waren drie hoofdtaken gepland in onderstaande volgorde:

1. Opstellen van requirements (1 week)
2. Selecteren van diagram-tekentool (2 weken)
3. Ontwikkelen van workflow-configuratietool met Scrum (resterende 10 weken)

Requirements

Het opzetten van de requirements is naar mijn mening goed volgens plan verlopen. In week twee ben ik begonnen met verschillende interviews en tegen het eind van de week had ik een lijst met requirements opgesteld en gereviewd. Ik heb met alle stakeholders kunnen praten en heb daardoor voldoende nieuwe inzichten kunnen krijgen om acceptabele requirements op te zetten.
Ik had voor extra zekerheid ook een interview willen houden met een andere toekomstige eindgebruiker van de tool: de externe consultants. Ik heb nu met een interne S&D-medewerker gesproken, maar ik kon mij voorstellen dat iemand die niet bij Nétive werkt toch andere ideeën heeft over workflows en de wijze waarop die zouden moeten worden opgesteld.

Tool-selectie

Het selecteren van de diagram-tekentool is goed afgerond. Er is een tool gekozen die goed aansluit op de requirements en die mij veel heeft geholpen bij het opzetten van gebruiksvriendelijke en krachtige tekenfuncties.
Ik ben echter wel van mening dat het gehele tool-selectie onderdeel te veel aandacht heeft gekregen binnen het project. Hiermee bedoel ik niet dat het niet belangrijk is om een goede en passende tool te selecteren, maar de tijd die ik hier binnen dit project in heb gestoken, had veel korter kunnen zijn. Dit had gemakkelijk één week kunnen zijn als ik minder tijd had besteed het aan testen en documenteren van de gevonden tools.
Al vrij vroeg in het onderzoek kwam ik tot de conclusie dat JointJS de ideale tool was om voor de workflow-configuratietool te gebruiken. Met de andere tools konden diagrammen worden gemaakt, maar met JointJS konden diagram-editors worden gemaakts. Het was speciaal ontworpen om aanpasbaar te zijn, en niet om als een kant-en-klaar product geïmplementeerd te worden. Desondanks heb ik, mede met het idee om de bijbehorende beroepstaak op een goed niveau uit te voeren, twee weken lang een volledig onderzoek uitgevoerd.

Ontwikkeltraject

Het ontwikkeltraject is ook goed volgens plan verlopen. Er was voldoende tijd om de belangrijkste requirements te realiseren. Het ontwikkeltraject was uiteindelijk twee weken korter dan gepland, omdat ik de laatste twee weken van het stageproject enkel aan de scriptie heb gewerkt.

Scrum

De keuze om Scrum te gebruiken voor het ontwikkeltraject was naar mijn mening een goede keuze. Ik denk dat Scrum een zeer behulpzame techniek is voor projecten zoals deze waar er redelijke onzekerheid is over wat er precies geprogrammeerd moet worden.
Door elke week een sprint planning en review te houden, kon de ontwikkeling van de tool goed worden gecontroleerd. Hierdoor is de kwaliteit van het product hoog gebleven en waren er voldoende contactmomenten met de bedrijfsmentor. Sprint-lengtes langer dan één week waren ook niet ideaal geweest gezien de korte duur van dit project.
Gedurende de ontwikkeling kwamen soms problemen naar boven over de kwaliteit van de configuratietool, maar dankzij de sprint reviews konden deze snel worden gedetecteerd en vervolgens worden opgelost. Scrum was geen nieuwe techniek voor mij, maar ik heb het tot voor dit project nog nooit met deze diepgang gebruikt. Gelukkig had mijn bedrijfsmentor genoeg ervaring met Scrum om mij hierin te helpen.

7.3 Gekozen Beroepstaken

1.4 Uitvoeren analyse door definitie van requirements

Deze beroepstaak is uitgevoerd in week 2 van het project, voor het eliciteren, opzetten, en reviewen van een lijst met requirements. Deze activiteit is op niveau 3 (lastig) afgerond.
Er is één lijst met requirements opgesteld. Ik heb de requirements volledig en dekkend op kunnen zetten door verschillende stakeholders te interviewen, en door ze van hoog tot laag niveau te categoriseren en uit te werken. De requirements zijn hierbij traceerbaar opgezet, en consistent en SMART geformuleerd.
De requirements zijn in week 2 nog niet geprioriteerd, maar zijn in week 5 omgezet naar User Stories in de Scrum Backlog, waar zij ook zijn geprioriteerd. De Backlog en de prioritering daarvan is vervolgens tijdens elke Sprint Planning gecontroleerd en bijgewerkt op basis van nieuwe inzichten verkregen over het product.

1.3 Selecteren van standaardsoftware

Deze beroepstaak is uitgevoerd in week 3 en 4 van het project, voor het selecteren van een diagram-tekentool voor gebruik in de workflow-configuratietool. Deze activiteit is op niveau 2 (simpel) afgerond.
Er is één applicatie geselecteerd op basis van selectiecriteria opgemaakt uit de requirements, de limitaties van het doelplatform, en de limitaties opgelegd vanuit bedrijf. Er is een longlist opgesteld, gereduceerd tot een shortlist aan de hand van afvalcriteria. De applicaties in de shortlist zijn beoordeeld op basis van de selectiecriteria, getest in een software-omgeving, en tegen elkaar afgewogen om de meest geschikte tool te kunnen kiezen.

3.2 Ontwerpen systeemdeel

Deze beroepstaak is uitgevoerd in week 5 van het project, voor het vastleggen van de informatiestructuur en de layout van de workflow-configuratietool. Deze activiteit is op niveau 2 (simpel) afgerond.
Er zijn twee systeemontwerpen gebouwd: een klassendiagram en een GUI-diagram. Deze waren redelijk eenvoudig van aard; het klassendiagram beschrijft niet alle klassen, variabelen en functies, en het GUI-diagram beschrijft enkel de hoofdpagina van de applicatie. Het klassendiagram is opgezet volgens de UML-regels.
Gezien de Agile-natuur van dit project is vrij weinig aandacht gelegd aan het ontwerpen van het systeem. De informatiestructuur en layout van de applicatie werd gedurende het project pas echt duidelijk en daarom was er niet veel aanleiding om voorafgaand aan de ontwikkeling volledige ontwerpen te maken.

3.3 Bouwen applicatie

Deze beroepstaak is uitgevoerd van week 5 tot en met week 12 van het project, voor het ontwikkelen van de workflow-configuratietool. Deze activiteit is op niveau 3 (lastig) afgerond.
Er is één applicatie opgeleverd, de workflow-configuratietool. Deze is geprogrammeerd met verschillende web-based programmeertalen, waaronder talen uniek aan het applicatie-framework. De code is volledig herbruikbaar en uitbreidbaar opgezet. De code is ook testbaar opgezet ten behoeve van de unit-tests die erop worden uitgevoerd. Er is gebruik gemaakt van het Event Sourcing design pattern om de functionaliteit te segregeren en versiebeheer toe te passen.
De applicatie is geprogrammeerd in Eclipse met daarin de Force.com IDE plug-in voor remote access tot de Salesforce-omgeving.

3.4 Initiëren en plannen van het testproces

Deze beroepstaak was gepland voor week 5 van het project, voor het vastleggen van het testproces. Deze activiteit is echter niet volgens de richtlijnen van de beroepstaak afgerond.
Voorafgaand aan het afstudeertraject was gepland deze beroepstaak op te pakken, met de gedachte dat daardoor een beter testtraject kon worden doorlopen. Achteraf bleek dat deze aanpak te gedetailleerd op de planning in ging voor dit project. Het testproces kon niet tot in detail worden vastgelegd, omdat er simpelweg nog niet genoeg bekend was over de te testen onderdelen. Daarom is deze beroepstaak aan de kant gezet en is besloten om het testproces in de loop van het ontwikkeltraject vast te leggen.

3.5 Uitvoeren van en rapporteren over het testproces

Deze beroepstaak is uitgevoerd van week 5 tot en met week 12 van het project, voor het testen van de workflow-configuratietool. Deze activiteit is op niveau 3 (lastig) afgerond.
Ondanks dat er geen testplan is opgezet, hebben er wel testwerkzaamheden plaatsgevonden. Deze zijn ontworpen naarmate nieuwe User Stories in de Sprints werden opgenomen. Er zijn tests opgezet van de types acceptatietest en unit-test. Voor elke User Story is een acceptatietest opgezet met acceptatiecriteria, precondities, en gewenste resultaten. De unit-tests zijn codematig opgezet met het QUnit test-framework, waardoor ze herhaalbaar konden worden uitgevoerd.
Van de uitgevoerde tests is een testrapport opgezet met daarin alle informatie benodigd om de tests uit te voeren.

Literatuurlijst

Activiti BPMN. (n.d.). Retrieved from Alfresco Software: https://www.activiti.org/

Alex, A. (2014). *9 Flowchart Tools for Creating Charts and Diagrams*. Retrieved from Codecondo: http://codecondo.com/9-flowchart-tools-creating-charts-diagrams

Baranovskiy, D. (n.d.). *Raphael.* Retrieved from http://dmitrybaranovskiy.github.io/raphael/

Bostock, M. (n.d.). *D3JS.* Retrieved from https://d3js.org/

Cacoo. (n.d.). Retrieved from Nulab Inc.: https://cacoo.com/

Creately. (n.d.). Retrieved from Cinergix Pty. Ltd.: https://creately.com/

Delgado, C. (2016). *Top 5: Best Free Diagrams Javascript Libraries*. Retrieved from Our Code World: http://ourcodeworld.com/articles/read/159/top-5-best-free-diagrams-javascript-libraries

Draw.IO. (n.d.). Retrieved from JGraph Ltd.: https://www.draw.io/

Ed-Douibi, H. (2015). *10 Javascript Libraries to Draw your own Diagrams*. Retrieved from Modeling Languages: http://modeling-languages.com/javascript-drawing-libraries-diagrams

Gliffy. (n.d.). Retrieved from Gliffy Inc.: https://www.gliffy.com/

GoJS. (n.d.). Retrieved from Northwoods Software: https://gojs.net/

Herz, A. (n.d.). *Draw2D.* Retrieved from http://www.draw2d.org/draw2d/

JointJS. (n.d.). Retrieved from Client.IO: https://www.jointjs.com/opensource

JsPlumb CE. (n.d.). Retrieved from JsPlumb Inc.: https://jsplumbtoolkit.com/

Lucidchart. (n.d.). Retrieved from Lucid Software: https://www.lucidchart.com/

MxGraph. (n.d.). Retrieved from JGraph Ltd.: https://jgraph.github.io/mxgraph/

Raiano, A. (n.d.). *FlowchartJS.* Retrieved from http://flowchart.js.org/

Rappid. (n.d.). Retrieved from Client.IO: https://www.jointjs.com/

Sveidqvist, K. (n.d.). *Mermaid.* Retrieved from https://knsv.github.io/mermaid/

Visio. (n.d.). Retrieved from Microsoft: https://products.office.com/nl-nl/visio/flowchart-software

Visual Paradigm. (n.d.). Retrieved from VP International: https://www.visual-paradigm.com/

Bijlage A – Afstudeerplan

Afstudeerplan

Informatie afstudeerder en gastbedrijf

Afstudeerblok: 2017-1.1
Startdatum uitvoering afstudeeropdracht: 6 februari 2017
Inleverdatum afstudeerdossier volgens jaarrooster: 2 juni 2017

Studentnummer: 13092197
Achternaam: dhr. Jansen
Voorletters: J.
Roepnaam: Jasper
Adres: Eikenlaan 16
Postcode: 2691ED
Woonplaats: ‘s-Gravenzande
Telefoonnummer: 0174 418 077
Mobiel nummer: 06 3833 9497
Privé e-mailadres: jasperjansen94@gmail.com

Opleiding: Informatica
Locatie: Den Haag
Variant: voltijd

Naam studieloopbaanbegeleider: B.M. Derks
Naam begeleidend examinator: O. Zor
Naam tweede examinator: A.M.J.J. Lousberg-Orbons

Naam bedrijf: Nétive
Afdeling bedrijf: Software Engineers
Bezoekadres bedrijf: Patrijsweg 20, Rijswijk
Postcode bezoekadres: 2289EX
Postbusnummer:
Postcode postbusnummer:
Plaats: Rijswijk
Telefoon bedrijf: 015 251 18 40
Telefax bedrijf:
Internetsite bedrijf: http://www.netive.nl/

Achternaam opdrachtgever: dhr. van der Wal
Voorletters opdrachtgever: M.
Titulatuur opdrachtgever:
Functie opdrachtgever: Product Development Manager
Doorkiesnummer opdrachtgever:
Email opdrachtgever: mike.van.der.wal@netive.nl

Achternaam bedrijfsmentor: dhr. Kuiper
Voorletters bedrijfsmentor: S.
Titulatuur bedrijfsmentor:
Functie bedrijfsmentor: Software Engineer
Doorkiesnummer bedrijfsmentor:
Email bedrijfsmentor: sven.kuiper@netive.nl

Doorkiesnummer afstudeerder:
Functie afstudeerder: Software Engineer

Titel afstudeeropdracht:
Ontwerpen en ontwikkelen van een workflow-configuratiesysteem bij Nétive.

Opdrachtomschrijving

1. Bedrijf

Nétive is een bedrijf dat software ontwikkelt dat zij aan andere bedrijven verkopen. Deze software valt onder de categorie Human Resource Management (HRM) software en wordt via een Cloud-omgeving gedistribueerd. De klanten van Nétive gebruiken deze software voornamelijk om processen omtrent flexibele arbeid te beheren. Nétive is opgericht in 2003 en heeft inmiddels 23 werknemers en meer dan 140 klanten aan wie zij software levert. Nétive roemt zich in het feit dat zij de enige volledig onafhankelijke Vendor Management System (VMS) aanbieder is in Nederland. Zij besteedt dan ook veel aandacht aan het personaliseren van de software aan de eisen van haar klanten. Mijn opdracht als software engineer zal voortbouwen op dit proces van het personaliseren van deze software-omgeving.

2. Probleemstelling

De software die Nétive biedt maakt het mogelijk om verschillende processen binnen een bedrijf te beheren. De klanten van Nétive gebruiken deze software voornamelijk voor het beheren van de inhuur van werknemers. De weergave van een dergelijk proces binnen de software van Nétive wordt een workflow genoemd. Een voorbeeld hiervan is de workflow die het proces van het inhuren van een nieuwe werknemer beschrijft. In deze workflow worden de verschillende stappen uitgelijnd die moeten worden ondernomen om dit proces goed te laten verlopen. De eerste stap van dit proces begint bijvoorbeeld bij een afdelingsmanager die een concept opstelt van een vacature. Deze persoon verstuurt de vacature naar een bedrijfsmanager ter controle, die aan de volgende stap begint. De VMS van Nétive biedt hierbij een weergave van de stand van zaken van dit proces in de vorm van een workflow. Hierin wordt onder andere ook weergegeven welk personeel en welke documenten er bij de workflow zijn betrokken.

Het probleem bij het vastleggen van deze workflows is dat geen enkel bedrijf exact dezelfde workflow aanhoudt voor hetzelfde proces. De volgorde van de af te lopen stappen en de betrokken personen en acties bij elke stap variëren voor elk bedrijf. Nétive kan hierdoor dus niet één workflow ontwikkelen die door al hun klanten kan worden gebruikt. Daarom gaat Nétive altijd in overleg met de klant om vast te leggen welke processen in hun bedrijf plaatsvinden. Na dit overleg gaat een software engineer van Nétive aan de slag om hier een workflow bij te ontwikkelen. Dit proces neemt al gauw een hele werkweek in beslag. Ook moeten er na deze periode vaak nog correcties gedaan worden aan de workflows.

Nétive heeft geconstateerd dat dit proces een stuk efficiënter kan verlopen als de klanten zelf de mogelijkheid hebben om hun workflows op te kunnen stellen binnen het VMS. Dit voorkomt ook dat er communicatieproblemen kunnen ontstaan tussen het bedrijf en de medewerkers van Nétive en kan alle betrokken partijen veel werkuren besparen. Nétive is een ambitieus bedrijf dat het liefst zo veel mogelijk klanten vergaart. De lange periode waarin de klant moet wachten voor hij volledig gebruik kan maken van de door hem aanschafte software, is hierbij echter een obstakel. Als Nétive dit probleem op kan lossen zal zij dus een betere service kunnen leveren.

3. Doelstelling van de afstudeeropdracht

Het doel van deze opdracht is om het bestaande workflow systeem van het VMS van Nétive uit te breiden. Deze uitbreiding moet het mogelijk maken voor mensen zonder kennis van programmeren om hun eigen workflows op te stellen. Hiervoor is het nodig om een omgeving te creëren waarin een visuele representatie gegeven kan worden van de door de gebruiker opgezette workflow-onderdelen. Deze omgeving zal bekend staan als de workflow-configuratietool. De workflow-onderdelen moeten binnen deze tool met behulp van een ”drag-and-drop” systeem kunnen worden toegevoegd. De tool moet ook uitbreidbaarheid zijn omdat nieuwe bedrijven vaak nieuwe eisen met zich mee brengen betreffende de functionaliteit van de workflows.

4. Resultaat

Een workflow-configuratietool maakt het mogelijk voor mensen om hun eigen workflows te creëren binnen het VMS van Nétive. Dit kan gebeuren zonder dat daar een programmeur van Nétive bij aan te pas hoeft te komen. Er zal echter nog wel een service-medewerker nodig zijn om de klant te helpen bij het opzetten van de workflows en om advies te geven. Desondanks kan deze toevoeging een significante tijdbesparing opleveren voor de medewerkers van Nétive.

Ook zal het prettig zijn voor de klanten om zelfstandig hun workflows op te stellen omdat ze dan complete controle en overzicht hebben over de ontwikkeling ervan. Zij zullen dan niet meer hoeven te wachten tot de medewerkers van Nétive de workflows voor hen gemaakt hebben. Ook zullen zij niet meer het risico lopen dat er communicatieproblemen optreden bij het beschrijven van de workflows.

Een voorbeeld van functionaliteit waar de workflow-configuratietool in moet voorzien, zijn de faseovergangen die binnen de workflow plaatsvinden. Afhankelijk van bepaalde opgegeven condities, moet de workflow van fase kunnen veranderen en moeten betrokken werknemers hier een bericht over krijgen. Ook moeten aan een workflow bepaalde rollen kunnen worden toegewezen. Deze geven aan welke gebruikers welke rechten hebben met betrekking tot het doorlopen van de stappen en het aanpassen van de workflow. Als een workflow-diagram af is, moet deze softwarematig worden vertaald zodat deze kan worden opgeslagen volgens de standaard van de bestaande workflow-engine.

5. Uit te voeren werkzaamheden

Het ontwikkelen van de workflow-configuratietool zal volgens de volgende vijf hoofdfases worden uitgevoerd. Dit project zal, zoals gebruikelijk binnen Nétive, via een iteratief proces worden uitgevoerd.

De eerste fase van het project zal worden gebruikt om mij te oriënteren op het bedrijf en de opdracht die ik uit zal voeren. Ik zal dan kennis maken met de software- en ontwikkelomgeving van Nétive en met het Cloud-platform waar ik met mijn opdracht op zal werken. Er zal in deze periode een plan van aanpak worden gemaakt waarin verder gedetailleerd wordt welke stappen zullen worden ondernomen om de doelstelling van deze opdracht te behalen. Hierbij zal ook worden gekeken welke problemen tijdens de opdracht naar voren kunnen komen en hoe ik deze op kan lossen. Deze fase zal ongeveer tien dagen in beslag nemen.

De tweede fase zal bestaan uit een onderzoek naar de beschikbaarheid van tekentools die mij kunnen helpen om de configuratietool te ontwikkelen. Hierbij moet worden gedacht aan software-platformen die een visuele representatie kunnen geven van processen, vergelijkbaar met flowchart-diagrammen. Een van deze tools moet dan geïntegreerd worden in de bestaande workflow engine van Nétive zodat gebruikers zelf hun workflows kunnen tekenen. Hier komt nog bij kijken dat deze tool moet worden aangepast zodat deze de mogelijkheden biedt die Nétive in hun workflows wil verwerken.

Er zal worden gekeken welke tools op dit moment beschikbaar zijn, waaruit een selectie zal worden gemaakt op basis van onder andere de implementatiemogelijkheden, aanpasbaarheid, en uitleesmogelijkheden van de diagrammen voor verwerking in de bestaande workflow-engine. Hierbij zal een rapport worden opgesteld waarin de verschillende aspecten van de software-platformen met elkaar worden vergeleken en afgewogen. Deze fase zal ongeveer vijf dagen in beslag nemen.

In de derde fase zullen de requirements van de te ontwikkelen software worden vastgelegd. Deze zullen onder andere afhangen van de tool die is gekozen in de fase twee. Bij het vaststellen van deze requirements zal worden overlegd met de stagebegeleider, die kennis heeft over het bestaande systeem en de mogelijke aanknooppunten daarin met het systeem dat ik ga ontwikkelen. De workflow-configuratietool is in feite een uitbreiding op een bestaand systeem. Daarom is het van belang dat goed wordt gekeken welke functionaliteit er al binnen het systeem beschikbaar is en hoe mijn software hierop aan kan sluiten. Ook moet rekening worden gehouden met het feit dat Nétive regelmatig nieuwe klanten binnenkrijgt, die op hun beurt weer andere workflows willen maken. De uitbreidbaarheid van de configuratietool is daarom een belangrijk aandachtspunt. Uit de opgestelde requirements kunnen vervolgens verschillende design-diagrammen en use-case beschrijvingen worden gemaakt. Hiermee kan ik voor mijzelf en de opdrachtgevers duidelijk maken wat precies geprogrammeerd gaat worden en aan welke onderdelen de meeste tijd zal worden gespendeerd. Deze fase zal ongeveer tien dagen in beslag nemen.

In de vierde fase zal het programmeren van de configuratietool van start gaan. Hier zal ik alle kennis die ik heb vergaard in de vorige fases moeten gebruiken om de eerder beschreven workflow-configuratietool te gaan maken. Als er tijd beschikbaar is, zal er ook een gebruiksaanwijzing worden opgeleverd die de eindgebruikers van de tool kunnen raadplegen. Deze fase zal ongeveer dertig dagen in beslagen nemen.

De vijfde fase van het project zit grotendeels verweven in de programmeerfase. Het testen van de workflow-configuratietool is een continu proces dat hand in hand gaat met het programmeren. Ook zullen er verschillende unit-tests worden opgezet, waarmee de software efficiënter getest kan worden. Naarmate verschillende onderdelen van de workflow-configuratietool klaar zijn voor gebruik, kunnen deze aan de hand van acceptatietests worden getest door de experts van Nétive en vervolgens door hun klanten. Deze onderdelen moeten ook worden getest op hoe goed zij aansluiten op de bestaande workflow-engine en dat zij daar geen fouten veroorzaken. Dit zal gebeuren met behulp van een integratietest. Als is geconcludeerd dat de workflow-configuratietool volledig klaar is voor gebruik, kunnen stappen worden gezet om deze definitief te implementeren in het huidige VMS van Nétive. Deze fase zal ongeveer vijftien dagen in beslag nemen.

6. Op te leveren producten

De producten die resulteren uit de verschillende fasen van de afstudeeropdracht zijn als volgt.

1. Een plan van aanpak over het verloop van de afstudeeropdracht in zijn geheel.
2. Een onderzoeksrapport waarin een keuze wordt gemaakt tussen verschillende diagram-tekentools.
3. Een document met een overzicht van de opgestelde requirements met betrekking tot de functionele- en gebruikseisen van de configuratietool.
4. Een document met de opgestelde klasse- en sequence-diagrammen en use-cases die de werking van de tool beschrijven, gemaakt op basis van de requirements.
5. Een testrapport waarin de uitgevoerde testprocessen en de resultaten daarvan zijn beschreven.
6. Een werkende workflow-configuratietool die de klanten van Nétive kunnen gebruiken om zelfstandig hun workflows op te kunnen zetten binnen de VMS.
7. Optioneel: Een gebruikershandleiding voor de workflow-configuratietool.

7. Te demonstreren competenties en wijze waarop

De beroepstaken die hieronder zijn beschreven, vertegenwoordigen de belangrijkste taken die ik tijdens mijn afstudeeropdracht uit zal voeren.

1.3 Selecteren van standaardsoftware

In de tweede fase van het project zal een keuze worden gemaakt voor een diagram-tekentool die gebruikt zal worden als basis voor de workflow-configuratietool. De keuze van deze diagram-tekentool is bepalend voor de mogelijkheden die ik heb voor het programmeren van de workflow-configuratietool. De software-ontwikkelmethoden, programmeertalen en ontwikkeltools die Nétive gebruikt zijn reeds vastgelegd, maar het is mogelijk dat ik voor het uitbreiden en implementeren van de configuratietool nieuwe technieken nodig heb die niet eerder gebruikt zijn bij Nétive.

1.4 Uitvoeren analyse door definitie van requirements

Het vastleggen van de requirements voor dit project zal in fase drie worden uitgevoerd. In overleg met de opdrachtgever en systeemexpert, zal hier gekeken worden naar de eisen waar de workflow-configuratietool aan moet voldoen om het opgestelde probleem op te lossen.

3.2 Ontwerpen systeemdeel

De software die in deze opdracht zal worden gemaakt, wordt eerst gemodelleerd aan de hand van de opgestelde requirements. Het betreft hier een compleet nieuwe uitbreiding op een bestaand systeem, die via een Cloud-platform aan de verschillende klanten van Nétive wordt geleverd. Hierbij zullen verschillende modellen worden gemaakt van de werking van het systeem en over de implementatie daarvan in het bestaande systeem.

3.3 Bouwen applicatie

Het bouwen van de workflow-configuratietool zal plaatsvinden in de reeds opgestelde ontwikkelomgeving van Nétive. Deze ontwikkelomgeving biedt verschillende omgevingen waarin gefaseerd kan worden geprogrammeerd, getest, en gedistribueerd. Het versiebeheer van de software is dan ook een belangrijk onderdeel van deze omgeving. Zoals eerder genoemd sluit de software die ik ga maken aan op een bestaand systeem van Nétive en daarom zal ik voor een groot deel ook dezelfde programmeertalen en -tools gebruiken.

3.4 Initiëren en plannen van het testproces

3.5 Uitvoeren van en rapporteren over het testproces

De belangrijkste aspecten van de software die moeten worden getest, zijn de foutloze werking ervan en het juist voldoen aan de verwachtingen van de klanten van Nétive. Er zal daarom veel aandacht worden besteed aan het maken van unittests en acceptatietests. Van deze tests zal eerst een testplan worden opgesteld om te verzekeren dat de verschillende onderdelen van de software de juiste hoeveelheid aandacht krijgen.

Bijlage B – Plan van Aanpak

1. Inleiding

In de periode van 6 februari tot 2 juni 2017 zal ik afstuderen bij het bedrijf Nétive, gevestigd in Rijswijk. In dit document wordt het plan van aanpak beschreven dat tijdens deze afstudeerstage zal worden gevolgd. De stageopdracht zal bestaan uit het ontwikkelen van een systeem, dat aansluit op een onderdeel van het Vendor Management System (VMS) van Nétive. Met behulp van dit systeem zal het mogelijk zijn voor de consultants van Nétive om zogeheten workflow-diagrammen vast te leggen. Dit is op het moment alleen mogelijk met tussenkomst van een programmeur en hier wil Nétive zo veel mogelijk van afzien.

2. Aanleiding

Nétive is een bedrijf dat software ontwikkelt dat zij aan andere bedrijven verkoopt. Deze software valt onder de categorie Human Resource Management (HRM) software en wordt via een Cloud-omgeving gedistribueerd. De klanten van Nétive gebruiken deze software voornamelijk om processen omtrent flexibele arbeid te beheren.
Een van de onderdelen van deze software zijn de workflows. Hierin wordt het gehele inhuurproces met bijbehorende voorwaarden, regels en taken schematisch weergegeven. De exacte opzet van deze workflows is uniek voor elk bedrijf, maar de klanten beschikken op het moment nog niet over een systeem waarmee zij zelfstandig deze workflows op kunnen stellen. Om deze reden is op het moment de tussenkomst van een programmeur altijd vereist voor het ontwikkelen van workflows.
Nétive is op het moment bezig met verschillende projecten die meer mogelijkheden moeten geven aan de klant op het gebied van configuratie binnen het VMS. Naast de workflows zijn er verschillende andere instellingen die niet direct door gebruiker zijn aan te passen. Daarom moet de klant nu elke keer naar Nétive moet bellen wanneer hij of zij een wijziging in deze instellingen wilt aanbrengen.
Het doel van mijn opdracht zal zijn om software te creëren die deze tussenkomst van een programmeur zo veel mogelijk moet elimineren op het gebied van workflow-configuratie. Deze software zal bekend komen te staan als de workflow-configuratietool. Deze tool moet de consultants van Nétive de mogelijkheid geven om zelfstandig hun workflows op te stellen zodat deze door hun klanten gebruikt kunnen worden. Het grootste belang bij deze ontwikkeling is dat het alle betrokken partijen een significant aantal werkuren kan besparen.

3. Opdrachtgever

De opdracht is opgesteld door Mike van der Wal. Hij is Product Development Manager binnen Nétive en overziet alle software-gerelateerde ontwikkelingen daarbinnen. Nétive is een bedrijf dat één specifiek product levert, namelijk het zogeheten Vendor Management System (VMS) en daar is hij de Product Owner van.

4. Probleemanalyse

Het primaire probleem bij het opstellen van workflows is dat dit een langdurig en foutgevoelig proces is. Voordat de programmeur aan de slag gaat, vindt veel conversatie plaats tussen de klant en een contactpersoon van Nétive. Aan de hand van de informatie die hier naar voren komt, gaat een programmeur aan de slag om een workflow te ontwikkelen. Deze wordt vervolgens online gezet en getest door de klant, die feedback geeft waar de programmeur mee verder kan werken. Dit proces gaat door totdat de klant volledig tevreden is met zijn workflows en kan al snel een gehele werkweek in beslag nemen.
Nétive heeft geconstateerd dat dit proces een stuk efficiënter kan verlopen als klanten zelf de mogelijkheid hebben om hun workflows op te kunnen stellen binnen het VMS. Dit voorkomt ook dat er communicatieproblemen kunnen ontstaan tussen het bedrijf en de medewerkers van Nétive en kan alle betrokken partijen veel werkuren besparen. De lange periode waarin de klant moet wachten voor hij volledig gebruik kan maken van het VMS, is ook een obstakel voor het werven van nieuwe klanten. Als Nétive dit probleem op kan lossen zal zij dus een betere service kunnen leveren.
In mijn afstudeeropdracht zal de workflow-configuratietool niet van de grond af worden opgebouwd, maar zal gepoogd worden om bestaande tools te gebruiken om het tekenen of anderzijds representeren van de workflows te faciliteren. De keuze van deze tekentool zal middels een onderzoek worden gemaakt. Hierbij is het ook mogelijk dat wordt bepaald dat er geen passende tekentool beschikbaar is. De nadruk zal hier worden gelegd op de begrijpelijkheid van de tool voor de klanten. De functionaliteit die in de configuratietool terecht komt, zal dus ook niet verder gaan dan wat redelijkerwijs te begrijpen is voor de gebruiker.

5. Doelstelling

Het doel van deze opdracht is om een systeemdeel te ontwikkelen waarin de klanten van Nétive met zo min mogelijk tussenkomst van een programmeur hun eigen workflows op kunnen stellen binnen het Nétive VMS.
Deze software moet een schematische weergave bieden van de workflows, zoals de gebruiker deze heeft opgesteld. Hierbij is het van belang dat de tool begrijpelijk is voor een persoon zonder technische achtergrond. Ook moet de gebruiker geen extra software hoeven te installeren om gebruik te kunnen maken van de tool. Deze moet namelijk op de Cloud-omgeving van Nétive of anderzijds op een webserver beschikbaar worden gesteld.
Nadat een gebruiker een workflow-diagram heeft getekend, moet deze worden vertaald naar een formaat dat door het VMS kan worden begrepen. Op deze manier kan de workflow worden geïmporteerd in het VMS en kunnen de klanten hun workflows blijven gebruiken zoals ze gewend zijn. Op vergelijkbare wijze moet een workflow-diagram ook vanuit het VMS kunnen worden geïmporteerd naar de diagram-tekentool om aangepast te kunnen worden. Waar nodig zal de bestaande workflow-engine ook worden aangepast om de nieuwe configuratietool te kunnen ondersteunen.

6. Aanpak

6.1 Oriëntatie & Requirements

Het ontwikkelen van de workflow-configuratietool zal in verschillende stappen gebeuren. Dit begint bij de oriëntatiefase, waarin zal worden gekeken hoe de workflow-configuratietool precies zal worden opgezet. Hierna zal ook worden vastgesteld welke functionaliteit de configuratietool precies moet bieden en hoe deze tool op het bestaande systeem aan kan sluiten. Hieruit ontstaat een requirementsoverzicht. Met de kennis van deze requirements zullen ook verschillende diagrammen worden opgesteld waarmee de werking van het systeem kan worden weergegeven.

6.2 Software-onderzoek

Het volgende deel van de opdracht zal bestaan uit een onderzoek naar bestaande diagram-tekentools die gebruikt kunnen worden als basis voor het tekenen van workflow- diagrammen. In dit onderzoek zullen verschillende tekentools worden geanalyseerd die beschikbaar zijn op het internet. Dit kunnen zowel gratis als betaalde tools zijn, zolang deze open-source zijn. Het is namelijk van belang dat de tekentool kan worden aangepast aan de hand van de opgestelde requirements.
De keuze van de diagram-tekentool zal worden gemaakt op basis van bepaalde eisen. Deze bestaan uit onder andere de aanpasbaarheid, implementatiemogelijkheden, uitleesmogelijkheden en prijs. Aan het eind van dit onderzoek zal een onderzoeksrapport beschikbaar worden gesteld waarin de keuze, of het afzien van een keuze, van de diagram-tekentool wordt onderbouwd.

6.3 Ontwikkelen

Als er eenmaal een tekentool is gekozen, of als is bepaald dat er geen toepasbare tekentools beschikbaar zijn, kan het programmeren van de configuratietool van start gaan. Het belangrijkste doel van het ontwikkelproces is het uitbreiden van de gekozen tekentool zodat deze aan de opgestelde requirements voldoet. Ook zal een brug moeten worden gelegd tussen de configuratietool en het bestaande workflow-systeem.
Het ontwikkeltraject van dit project wordt ingedeeld aan de hand van de Scrum methodiek. Er zal in de applicatie JIRA een Backlog worden opgezet met User Stories, die aan de hand van de requirements zijn opgesteld. Deze User Stories krijgen een aantal issue points die een indicatie geven van de hoeveelheid tijd die nodig zal zijn om het issue af te maken. De User Stories zullen in sprints worden opgepakt, welke elk een week zullen duren.
Aan het begin van elke sprint zal een sprint review plaatsvinden met de stagebegeleider, waarin wordt gekeken of alle issues van de afgelopen sprint zijn behaald en of het bestede puntenaantal correct is of moet worden bijgesteld. Bij een sprint review kan worden besloten om nieuwe issues aan te maken of andere issues een lagere prioriteit te geven. Zo kan worden ingespeeld op mogelijke wijzigingen in de product-requirements. Deze wekelijkse verfijning van de User Stories hebben met terugwerkende kracht ook invloed op de opgestelde lijst met requirements.
Na deze review volgt een sprint planning. Hierin gaan wij met de kennis opgedaan uit de voorgaande sprint de nieuwe sprint invullen met de issues die de hoogste prioriteit hebben. Deze sprints zullen tot het einde van het project doorlopen.
De issues die in elke sprint worden opgepakt, worden ook getest aan de hand van verschillende testmethodes. Deze testmethodes bestaan uit verschillende functionele- en systeem tests waarmee de algemene werking van de applicatie en de overeenkomst ervan met requirements wordt getoetst.

Hierbij zullen verschillende testmethodes worden gebruikt, waaronder unit-tests om de code te testen en acceptatietests om de resultaten van de User Stories te testen.
Gedurende elke sprint zal het systeemontwerp ook worden gedocumenteerd met behulp van klassendiagrammen en sequentiediagrammen. Deze diagrammen zullen de structuur en werking van het systeem beschrijven met als doel om deze aspecten van het systeem begrijpelijk en overzichtelijk te houden.

6.4 Weekindeling

Hieronder is een overzicht gegeven van de verschillende stappen die gedurende het afstudeertraject zullen worden doorlopen, in combinatie met de bijbehorende eindproducten en het aantal werkdagen dat ik hier ongeveer mee bezig zal zijn. In de ontwikkelfase van het project zullen wekelijks iteraties op het project plaatsvinden waarin de requirements, User Stories en de applicatie zelf worden verfijnd met behulp van de Scrum methodiek.

	Stap	**Eindproduct(en)**	**Duur (dagen)**
1	Oriëntatiefase	Plan van aanpak	5
2	Eliciteren requirements	Requirementsoverzicht	5
3	Onderzoek diagram-tekentool	Onderzoeksrapport	10
4	Ontwikkelfase a.d.h.v. Scrum	Workflow-configuratietool, Documentatie, Diagrammen, Testrapport	50

7. Risicoanalyse

Aan dit project zitten ook bepaalde risico's verbonden die gedurende het project in acht moeten worden genomen om deze tot een succesvol einde te brengen. Hier zal ook worden beschreven hoe deze gedurende het project kunnen worden vermeden.

7.1 Complexiteit workflows

Het grootste risicofactor binnen deze opdracht is de complexiteit van de workflows die met de configuratietool moeten worden gemaakt. Er zijn een groot aantal requirements te bedenken waar de tool aan moet voldoen en de kans bestaat dat niet alle requirements binnen het afstudeertraject gerealiseerd kunnen worden. Er zal dan ook een nadruk worden gelegd op het ontwikkelen van een werkend en acceptabel prototype van de workflow- configuratietool.
Het probleem bij deze ontwikkeling treedt op wanneer onverhoopt de minder belangrijke requirements te veel aandacht krijgen. Dit probleem kan ontstaan doordat er te weinig tijd beschikbaar is om alle belangrijke requirements te vervullen, maar het kan ook gebeuren dat bepaalde requirements over het hoofd zijn gezien.
Dit probleem kan worden voorkomen door eerst zorgvuldig vast te stellen welke functionele eisen er aan de configuratietool zijn verbonden. Hiervoor zal regelmatig worden vergaderd met de workflow-experts en Product Owners. De opgestelde eisen kunnen vervolgens worden geprioriteerd op basis van het belang daarvan binnen het systeem en de tijd die benodigd is om deze onderdelen te realiseren.

7.2 Gebruiksvriendelijkheid configuratietool

Tegenover de complexiteit van de workflows staat het feit dat de workflow-configuratietool zelf niet te complex moet worden. Het is de bedoeling dat de consultants van Nétive de tool zelfstandig en intuïtief kunnen gebruiken. Ook moet de gebruiker niet in het bezit hoeven te zijn van technische kennis over het achterliggende workflow-systeem. Als de gebruiker steeds contact op moet nemen met Nétive voor ondersteuning, had de ontwikkeling van workflows net zo goed bij de software engineers kunnen blijven liggen. Het is dus van belang om niet te overschatten wat voor taken de gebruiker wil en kan uitvoeren.
Dit probleem kan worden voorkomen door een goed beeld te vormen van wat de expertise is van de Service & Delivery afdeling van Nétive. Zij zijn namelijk een van de eindgebruikers van de tool en hebben het meeste inzicht in de kennis van de klanten.

7.3 Beschikbaarheid tekentool

Een andere risicofactor is de beschikbaarheid van een bruikbare diagram-tekentool. De tekentool moet aan een aantal zeer belangrijke eisen voldoen als wij deze willen gebruiken voor de workflow-configuratietool. Deze moet namelijk op een fundamenteel niveau aanpasbaar zijn, het moet geïntegreerd kunnen worden in een online-omgeving, en het moet uitgelezen kunnen worden zodat de diagrammen in het bestaande workflow-systeem kunnen worden opgenomen.
Als definitief wordt bepaald dat er geen diagram-tekentools beschikbaar zijn die gebruikt kunnen worden als basis voor de workflow-configuratietool, zal een alternatief moeten worden verzonnen om workflows te representeren. Ik kan bijvoorbeeld zelf een infrastructuur ontwikkelen waarin de gebruiker met behulp van visuele representatie workflow-diagrammen kan tekenen, maar er zijn ook andere presentatiemogelijkheden te bedenken waarbij helemaal niks getekend wordt. Het ontwikkelproces zal in deze situatie meer tijd in beslag nemen dan wanneer een bestaande tekentool kan worden gebruikt. Het doel blijft echter dat in ieder geval een concept wordt gemaakt van een systeem waarin workflows kunnen worden opgesteld. Deze keuzes zullen worden beschreven in het onderzoeksrapport.

7.4 Risico's vermijden

Van bovenstaande risicofactoren is beschreven welke stappen er moeten worden gezet om te voorkomen dat deze een probleem gaan vormen voor het product. Om het vermijden van deze risicofactoren te waarborgen, zal elke week tijdens de sprint review worden gekeken of de risicofactoren voldoende in acht zijn genomen. Als wij bijvoorbeeld zien dat de workflow-configuratietool te complex of onoverzichtelijk wordt, zullen er nieuwe issue points worden gemaakt om de configuratietool te simplificeren.
Door gebruik te maken van de Scrum methodiek kan snel worden gereageerd op deze opdoemende problemen en kan met meer zekerheid een acceptabel product worden opgezet.

8. Randvoorwaarden

Om de opdracht succesvol te kunnen doorlopen, moet aan bepaalde materiële en kennis-gerelateerde voorwaarden worden voldaan.
De belangrijkste voorwaarde van deze opdracht is dat de kennis en filosofie achter de workflows goed duidelijk wordt. Workflows zijn niet alleen groot en complex, maar ook in zulke mate gevarieerd dat het lastig is om alle functionaliteit te achterhalen die in de configuratietool beschikbaar moeten worden gesteld. Daarom is het van belang dat deze aspecten goed duidelijk worden gemaakt, als ik de workflow-configuratietool doeltreffend wil programmeren. Het is hiervoor ook van belang dat ik vaardig word met de programmeertaal en ontwikkelomgeving die Nétive gebruikt.
Naarmate de configuratietool wordt geprogrammeerd, moet deze ook worden getest. Een belangrijk onderdeel van dit testproces zijn de acceptatietests, die als doel hebben om vast te stellen of de functionaliteit juist is ontwikkeld. Om deze tests uit te kunnen voeren zijn medewerkers van Nétive nodig die ervaring hebben met workflows. Bij Nétive zijn dit voornamelijk de medewerkers van de Service & Delivery. Zij kunnen het best vaststellen of de configuratietool aan de verwachtingen voldoet die Nétive hiervoor heeft opgesteld. Als is vastgesteld dat een kwalitatief goede en bruikbare workflow-configuratietool beschikbaar is, kan worden besloten deze ook te laten testen door een van de partners van Nétive.
Verder is een computer met toegang tot de Cloud-omgeving van Nétive en software- ontwikkeltools vereist. Afhankelijk van de gekozen diagram-tekentool is het ook mogelijk dat er een investering gedaan wordt om de rechten te verkrijgen deze te mogen implementeren in het Nétive VMS. Mocht dit het geval zijn, zal er een investeringsvoorstel ingediend moeten worden bij het management van Nétive.

Bijlage C – Requirements Specificatie

1. Inleiding

In dit document zal een overzicht worden gegeven van de requirements zoals die zijn opgesteld voor het afstudeerproject. Gedurende dit project zal een workflow-configuratietool worden ontwikkeld voor het bedrijf Nétive in Rijswijk. Workflows worden al langer gebruikt binnen het bedrijf maar deze tool zal het makkelijker maken om deze op te zetten en moet ook de tussenkomst van programmeurs bij het configureren minimaliseren.

2. Project

2.1 Doelstelling

Er moet een tool gemaakt worden waarin workflows kunnen worden geconfigureerd. Deze workflow-configuratietool zal worden gebruikt door de consultants van Nétive die in overleg met de klanten een workflow op kunnen zetten.
Het product wordt een deel van een groter systeem. Nétive is bezig met het maken van een tool waarin een grote hoeveelheid instellingen zijn aan te passen door de klant. De workflow-configuratietool is hier een deel van, ook al zal deze in eerste instantie vooral door de consultants gebruikt worden en niet de klanten zelf.

2.2 Context

Het Vendor Management System (VMS) is een software-omgeving waarin de klanten van Nétive onder andere hun processen omtrent flexibele arbeid kunnen beheren. Workflows zijn een bestaand onderdeel van het VMS. Workflows worden tot op heden echter voornamelijk door de Software Engineers van Nétive gemaakt, nadat een consultant de specificaties hiervan heeft achterhaald in overleg met de klant.
De configuratietool moet het mogelijk maken om deze workflows door de consultants zelf te laten configureren. De rol van consultants wordt vervuld door medewerkers van de Service & Delivery afdeling van Nétive en hun partners. Deze mensen beschikken over het algemeen niet over een technische achtergrond en hier moet rekening mee worden gehouden bij het ontwikkelen van de configuratietool.

2.3 Stakeholders

Producteigenaars

De managers van Nétive zijn de producteigenaars van het VMS. Het is ook hun taak om het product aan potentiële klanten te verkopen. Hun wens is om de betrokkenheid van programmeurs bij het ontwikkelproces van workflows te minimaliseren en om het gehele configuratieproces sneller te laten verlopen.

Consultants

De consultants van Nétive bestaan uit medewerkers van de Service & Delivery afdeling van Nétive en verschillende partners van Nétive. Zij zijn het eerste aanspreekpunt van de klant en hebben als taak om te zorgen dat de wensen van de klant worden vervuld. De consultants zullen gebruik maken van de workflow-configuratietool om klant specifieke workflows op te stellen en aan te passen.

Klanten

De klanten van Nétive zijn de mensen die gebruik maken van de workflows, zoals deze zijn opgesteld binnen hun omgeving in het VMS. Zij hebben als taak om aan de consultants van Nétive duidelijk te maken hoe hun bedrijfsstructuur in elkaar zit, zodat de consultants hier workflows van kunnen maken. De workflow-configuratietool zal in niet door de klanten gebruikt worden, maar door de consultants van Nétive. Het belang van de klant bij de configuratietool is dat, met behulp van deze tool, workflows sneller naar de productieomgeving van de klant kunnen worden gedistribueerd en minder fouten zullen bevatten.

3. Vaststellen Requirements

3.1 Elicitatie

In de beginfase van het project ben ik meerdere malen in overleg gegaan met de Product Owner en verschillende software engineers van Nétive om de requirements van de workflow-configuratietool te eliciteren. De belangrijkste informatie die hieruit voortkwam bestond uit kennis over de werking van het huidige workflow-systeem en de ideeën die zij hadden over de manier waarop de workflow-configuratietool kon worden gebruikt. Aan de hand van deze gesprekken en de informatie die erin naar voren kwam is een verzameling concept-requirements gemaakt.

3.2 Review

De geëliciteerde requirements zijn in vergaderingen met de Product Owner en een software engineer geëvalueerd. Aan de hand van deze gesprekken zijn verschillende aspecten van het project aan het licht gekomen die eerder niet behandeld waren. Deze zijn vervolgens in het requirements overzicht aangevuld.

3.3 Prioritering & verfijning

De prioritering van de requirements is gedaan met als doel om zo spoedig mogelijk een werkende concept-applicatie op te stellen. Uit deze requirements worden User Stories opgezet, welke volgens de Scrum methodiek met sprints worden ingepland. Elke sprint vindt een sprint review plaats, waarin de opgezette User Stories worden geëvalueerd, geprioriteerd en desnoods uitgebreid. Deze wekelijkse verfijning van de User Stories hebben met terugwerkende kracht ook invloed op de opgestelde lijst met requirements.

4. Requirements

4.1 Business Requirements

De business requirements geven aan wat het bedrijf wil bereiken met het project. Deze requirements geven leven aan alle andere requirements.

ID	Beschrijving
BR-01	De Product Owner wil dat consultants de mogelijkheid hebben om workflows op te stellen, te beheren en te distribueren met behulp van een configuratietool.
BR-02	De Product Owner wil dat de workflows opgesteld in de configuratietool op dezelfde manier kunnen worden gebruikt als die van bestaande workflows.
BR-03	De Product Owner wil dat consultants de workflow-configuratietool makkelijk en zelfstandig kunnen gebruiken.
BR-04	De Product Owner wil dat consultants de workflow-configuratietool kunnen gebruiken voor documentatie doeleinden.
BR-05	De Product Owner wil dat programmeurs minder tijd spenderen aan het opzetten van workflows.
BR-06	De Product Owner wil dat klanten minder lang hoeven te wachten totdat zij gebruik kunnen maken van hun workflows.
BR-07	De Product Owner wil dat programmeurs kunnen vastleggen wie toegang heeft tot de workflow-configuratietool.
BR-08	De Product Owner wil dat consultants verschillende versies van een workflow in de workflow-configuratietool kunnen beheren.

4.2 Gebruikers Requirements

De functionele gebruikers requirements geven aan welke eisen de gebruiker aan het systeem stelt om hier gebruik van te kunnen maken. Deze worden opgesteld vanuit het oogpunt van de gebruiker en gaan niet veel dieper dan wat de gebruiker op zijn scherm ziet. Hieruit volgen zowel Functionele- als Niet-Functionele Requirements.

ID	Beschrijving	↳ Uit	Prioriteit
UR-01	Consultants moeten bestaande workflows kunnen openen.	BR-01	M
UR-02	Consultants moeten nieuwe workflows kunnen aanmaken.	BR-01	M
UR-03	Consultants moeten workflows kunnen aanpassen.	BR-01	M
UR-04	Consultants moeten workflows die niet met de workflow-configuratietool zijn gemaakt kunnen importeren.	BR-01	S
UR-05	Consultants moeten workflows naar een omgeving van een klant kunnen distribueren.	BR-01	S
UR-06	Consultants moeten geopende workflows kunnen opslaan in een lokale omgeving.	BR-01	S
UR-07	Consultants moeten statussen aan de workflow kunnen toevoegen, bewerken en verwijderen.	BR-02	M
UR-08	Consultants moeten namen, beschrijvingen en toegangsrechten van statussen kunnen invoeren, wijzigen en verwijderen.	BR-02	M
UR-09	Consultants moeten transities tussen statussen kunnen vastleggen in de workflow en deze kunnen bewerken en verwijderen.	BR-02	M

UR-10	Consultants moeten labels, condities, rechten en acties van transities kunnen invoeren, wijzigen en verwijderen.	BR-02	M
UR-11	Consultants moeten losstaande labels en lijnen aan het workflow-diagram kunnen toevoegen.	BR-04	C
UR-12	Beheerders moeten workflow-templates kunnen vastleggen.	BR-03	C
UR-13	Consultants moeten bij het maken van een workflow een workflow-template kunnen kiezen.	BR-03	C
UR-14	Beheerders moeten vastleggen welke personen welke workflows via de configuratietool mogen inzien en bewerken.	BR-07	C

4.3 Functionele Requirements

De functionele systeem requirements geven harde technische eisen aan het systeem. Deze hebben verschillende niveaus van diepte. Hoog niveau Systeem Requirements kunnen resulteren in meerdere lagere niveau Systeem Requirements.

ID	**Beschrijving**	**↳ Uit**	**Prioriteit**
Workflow-configuratietool			
FR-01	Het systeem moet een workflow-configuratie kunnen inladen.	UR-01	M
FR-02	Het systeem moet elementen uit een workflow grafisch kunnen weergeven.	UR-01	M
FR-03	Het systeem moet een overzicht van bestaande workflow-configuraties kunnen geven.	UR-01	S
FR-04	Het systeem moet een nieuwe workflow-configuratie kunnen aanmaken.	UR-02	M
FR-05	Het systeem moet workflows die niet met de configuratietool zijn gemaakt kunnen importeren.	UR-04	S
FR-06	Het systeem moet vastleggen welke personen welke workflows via de configuratietool mogen inzien en aanpassen.	UR-14	C
FR-07	Het systeem moet de informatie in de workflows kunnen converteren van en naar XML-formaat.	C-01	S
FR-08	Het systeem moet workflows naar een test- of productieomgeving kunnen distribueren.	UR-05	C
FR-09	Het systeem moet het mogelijk maken om de gedistribueerde versie van een workflow terug te draaien naar een oude versie.	NF-05	S
FR-10	Een gedistribueerde workflow moet een versienummer en distributiedatum toegewezen krijgen.	FR-09	S
FR-11	Als de distributie van een workflow wordt geüpdatet moet het systeem de oude versie van de distributie opslaan.	FR-09	S
FR-12	Het systeem moet laten zien welke versie van een workflow-configuratie op een bepaald moment online staat.	NF-05	C
FR-13	Het systeem moet door middel van tabjes tussen geopende workflows kunnen wisselen.	NF-02	W
FR-14	Het systeem moet screenshots kunnen exporteren van het workflow-diagram.	NF-06	W
Workflow-configuraties			
FR-15	Een workflow-configuratie moet bestaan uit een naam, de naam van de klant en een type.	UR-02	S
FR-16	Het systeem moet een workflow van XML-formaat kunnen importeren om een workflow-configuratie aan te maken.	UR-04	S
FR-17	Het systeem moet de inhoud van een workflow aan de hand van templates kunnen invullen.	NF-02	C
Statussen			

FR-18	Het systeem moet statussen aan een workflow-diagram kunnen toevoegen.	UR-07	M
FR-19	Het systeem moet de naam, beschrijving en fase van een status kunnen laten invullen en wijzigen.	UR-08	M
FR-20	Het systeem moet de naam, beschrijving en fase van een status kunnen weergeven.	UR-08	S
FR-21	De beschikbare statusnamen moeten uit de gekoppelde Salesforce-database worden ingelezen.	NF-01	C
FR-22	Het systeem moet, wanneer bepaalde statussen zijn toegevoegd, suggesties kunnen geven voor het automatisch toevoegen van andere statussen en transities.	NF-02	W
FR-23	Het systeem moet de lees- en schrijfrechten van een status kunnen laten invullen en wijzigen.	UR-08	M
FR-24	Het systeem moet de lees- en schrijfrechten van een status kunnen weergeven.	UR-08	S
FR-25	Lees- en schrijfrechten moeten kunnen worden ingevuld door een of meerdere gebruikersrollen in een lijst aan te vinken.	NF-02	M
FR-26	De beschikbare gebruikersrollen moeten uit de gekoppelde Salesforce-database worden ingelezen.	NF-01	C
FR-27	Het systeem moet een status in de vorm van een rechthoek in het diagram weergeven.	FR-18	M
FR-28	Het systeem moet de naam en beschrijving van een status in het diagram laten zien.	NF-03	S
FR-29	Het systeem moet de optie geven om de grootte van een status aan te passen.	NF-02	W
Transities			
FR-30	Het systeem moet een transitie kunnen toevoegen door een lijn van een status naar een andere status te trekken.	UR-09	M
FR-31	Het systeem moet op een bestaande lijn meerdere transities toe kunnen laten voegen.	NF-02	S
FR-32	Transities moeten kennis hebben van de statussen waar ze mee zijn verbonden.	FR-30	M
FR-33	Een status moet met meerdere verschillende statussen en met zichzelf kunnen zijn verbonden.	FR-30	M
FR-34	Het systeem moet de richting van een transitie aan kunnen geven en wijzigen.	FR-30	S
FR-35	Het systeem moet de naam, beschrijving en modus van een transitie kunnen laten invullen en wijzigen.	UR-10	M
FR-36	Het systeem moet de naam, beschrijving en modus van een transitie kunnen weergeven.	UR-10	S
FR-37	Het systeem moet een of meer transitierechten, condities en acties aan transities kunnen toekennen.	UR-10	M
FR-38	Het systeem moet de transitierechten en acties van een transitie kunnen weergeven.	UR-10	S
FR-39	Transitierechten moeten kunnen worden ingevuld door een of meerdere gebruikersrollen in een lijst aan te vinken.	NF-02	M
FR-40	Het systeem moet op elke status meerdere contactpunten weergeven van waaruit transities kunnen worden getrokken en waarop transities kunnen worden gekoppeld.	NF-02	S
Condities & Acties			
FR-41	Aan rechten en transities moeten condities kunnen worden gekoppeld.	FR-29	M
FR-42	De beschikbare condities moeten uit de gekoppelde Salesforce-database worden ingelezen.	FR-29	M
FR-43	Acties kunnen bestaan uit een of meer veld-updates.	UR-07	S
FR-44	Het systeem moet bij veld-update-acties een veldnaam en datatype kunnen laten invullen en verplichte velden kunnen laten aanvinken.	FR-34	S

Templates & Validatie			
FR-45	Templates moeten bij het aanmaken van een nieuwe workflow uit een vooraf gedefinieerde keuzelijst kunnen worden gekozen.	UR-13	S
FR-46	Templates moeten vastleggen wat de minimale verzameling workflow-onderdelen in een workflow is en wat de validatieregels zijn.	NF-01	S
FR-47	Het systeem moet workflows als templates kunnen opslaan voor later gebruik.	UR-12	W
FR-48	Het systeem moet opgeslagen templates kunnen aanpassen en verwijderen.	UR-12	C
FR-49	Het systeem moet aan de hand van het gekozen template weten welke validatieregels er moeten worden aangehouden.	UR-13	S
FR-50	Het systeem moet kunnen controleren of de workflow-onderdelen en -informatie in de workflow aan de opgegeven validatieregels voldoen.	NF-01	S
FR-51	Het systeem moet weerhouden dat bepaalde workflow-onderdelen en -informatie kunnen worden verwijderd aan de hand van de opgegeven validatieregels.	NF-01	S
FR-52	Het systeem moet kunnen controleren of het eindpunt van de workflow vanuit het beginpunt bereikt kan worden.	NF-01	S
FR-53	Het systeem moet kunnen aanwijzen welk onderdeel van een workflow invalide is of mist.	NF-02	C
FR-54	Het systeem moet automatisch controleren of een workflow valide is.	NF-02	W
FR-55	Het systeem moet voorkomen dat een niet-valide workflow via de configuratietool gedistribueerd kan worden.	NF-01	S
Usability			
FR-56	Het systeem moet markeren welk object in de workflow is geselecteerd.	NF-02	S
FR-57	Het systeem moet de optie geven om de weergave van statussen en transities te filteren op basis van de informatie die daarin is ingevuld.	NF-02	C
FR-58	Het systeem moet door middel van sneltoetsen kunnen worden bediend.	NF-02	C
FR-59	Het systeem moet de verrichte wijzigingen van de gebruiker in de workflow kunnen terugdraaien of opnieuw toepassen.	NF-02	W
FR-60	Het systeem moet met behulp van context-menu's toegang geven tot alle functionaliteit waarmee objecten in het systeem kunnen worden toegevoegd, gewijzigd en verwijderd.	NF-02	S
FR-61	Het systeem moet van alle opties van rechten een invulmatrix laten zien met de beschikbare gebruikersrollen tegenover de statussen en transities, waarmee deze ingevuld kunnen worden.	NF-02	C

4.4 Niet-functionele Requirements

De niet-functionele requirements geven kwalitatieve eisen over de algemene werking van het systeem. Hieruit kunnen functionele requirements volgen die helpen de kwaliteitseisen te realiseren.

ID	Beschrijving	↳ Uit	Prioriteit
NF-01	Het systeem moet valide workflows produceren die voldoen aan de structuur van bestaande workflows.	BR-02	M
NF-02	Het systeem moet makkelijk en intuïtief zijn om te gebruiken.	BR-03	M
NF-03	Het systeem moet volgens de "What You See Is What You Get" techniek zijn vormgegeven.	BR-03	S
NF-04	De schematische opmaak van workflows moet bewaard blijven tussen sessies.	BR-03	C
NF-05	Het systeem moet verschillende versies van een workflow kunnen beheren.	BR-08	S
NF-06	De workflows getekend in het systeem moeten voor documentatie doeleinden zijn te gebruiken.	BR-04	S
NF-07	Het systeem moet goed en snel presteren.		S

4.5 Constraints

De constraints geven harde grenzen aan waar het systeem aan moet voldoen op business niveau. Hieruit volgen Systeem Requirements die de constraints afdwingen en Niet-functionele Requirements die de kwaliteitseisen waarborgen.

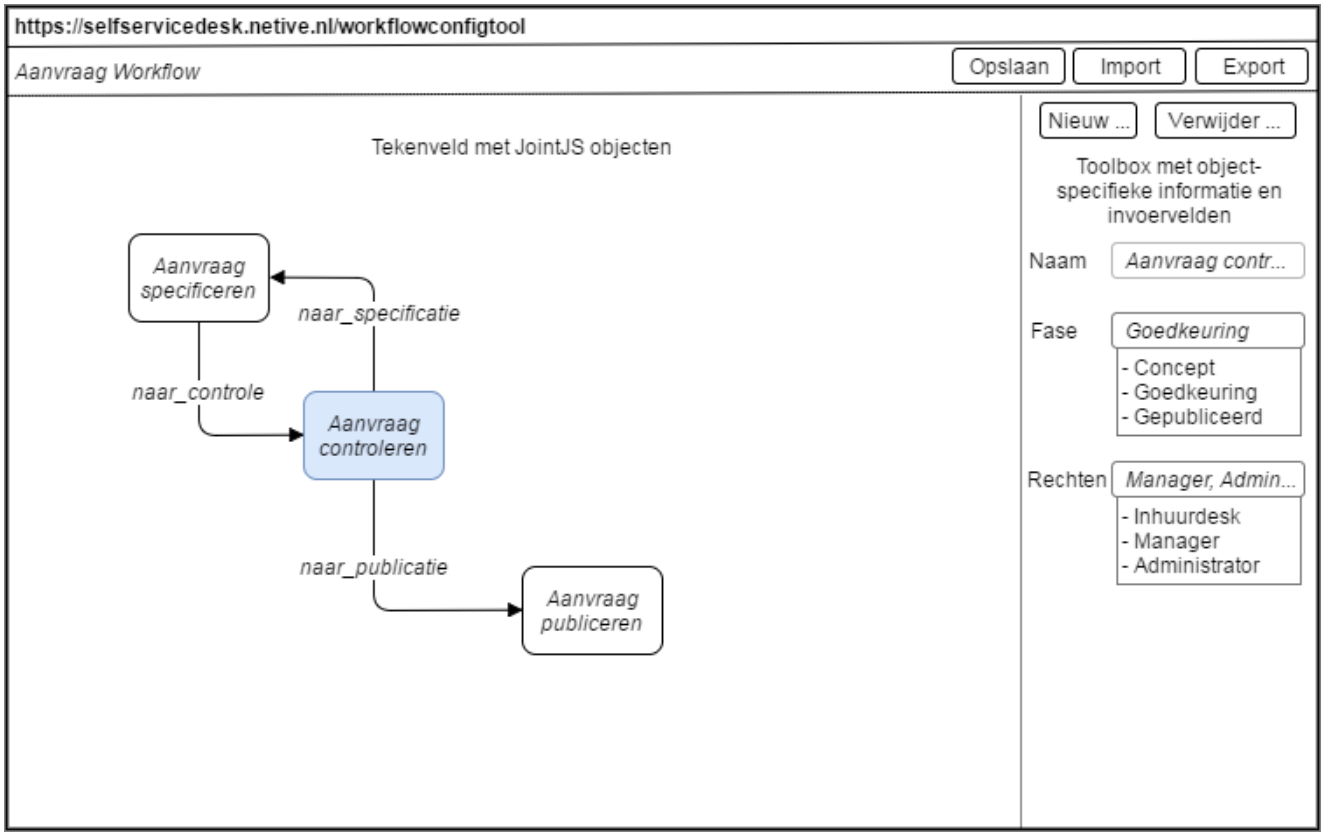
ID	Beschrijving	↳ Uit
C-01	Het systeem moet workflows in XML-formaat distribueren overeenkomstig met de structuur van bestaande workflows.	BR-02
C-02	Het systeem moet door mensen zonder technische achtergrond te gebruiken zijn.	BR-03
C-03	Het systeem moet in het Engels weergegeven zijn.	
C-04	Het systeem moet op de meest recente versies van Chrome, Firefox, Safari en Edge kunnen draaien.	
C-05	Het systeem moet met het Salesforce Lightning Design System zijn vormgegeven.	

5. Ontwerp

Op basis van de requirements zijn ontwerpen gemaakt van de layout van- en de informatiestructuur in de workflow-configuratietool. Hieronder zullen deze worden weergegeven in combinatie met een uitleg over de inhoud daarvan.

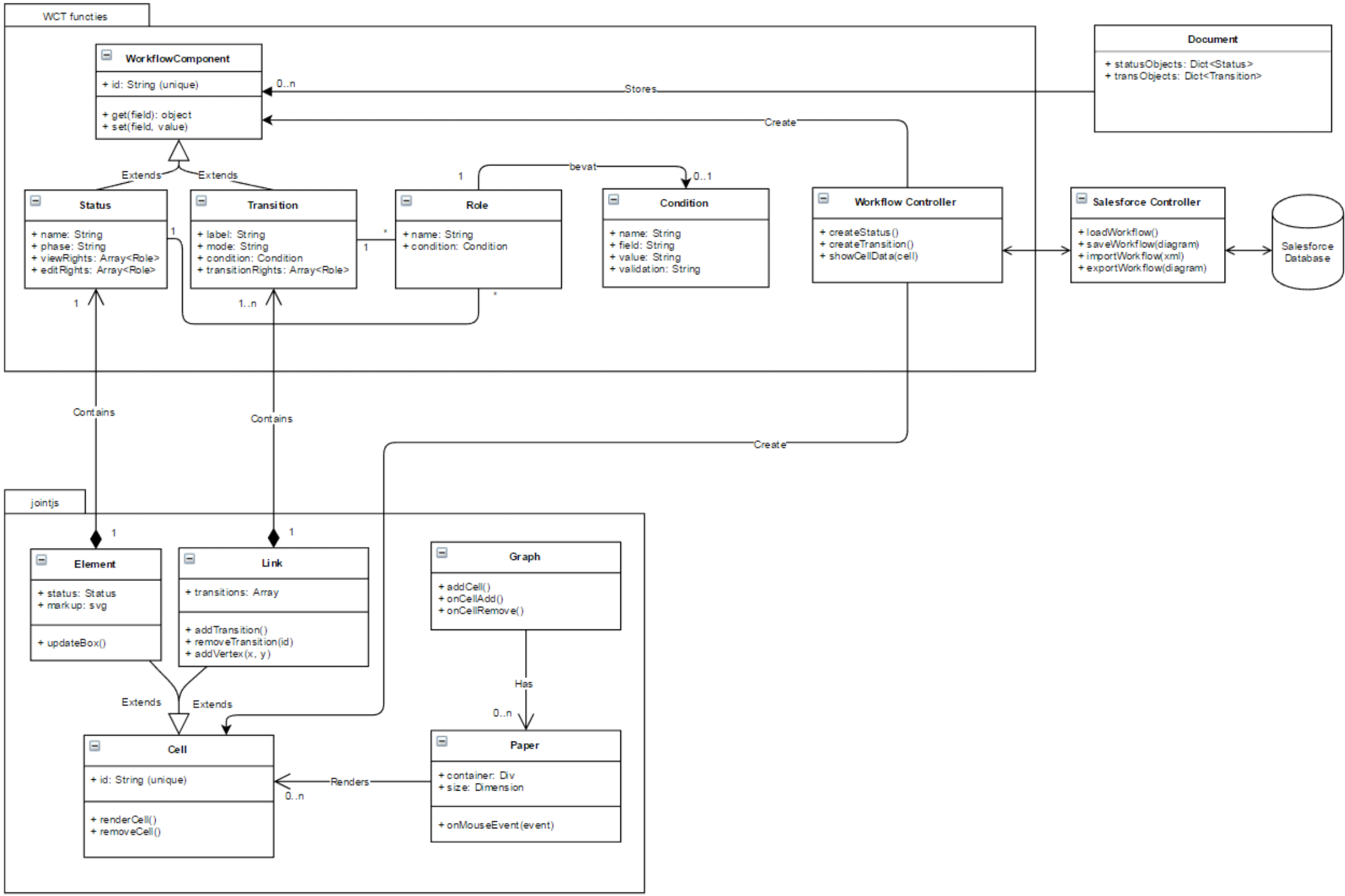
Tool-layout

Onderstaand diagram geeft aan hoe de configuratietool eruit moest gaan zien. Bovenin de header te zien met knoppen voor algemene functies, zoals het opslaan en exporteren van workflows. Aan de linker kant is het tekenveld te zien, waarin enkele workflow-objecten in zijn getekend. In dit voorbeeld heeft de gebruiker op de status “Aanvraag controleren” geklikt, waardoor informatie over het object aan de rechterkant van de pagina in de toolbox wordt weergegeven.



Klassendiagram

Op de volgende pagina staat het klassendiagram weergegeven van de workflow-configuratietool. Hierin is terug te zien hoe Statussen en Transities in respectievelijk Elementen en Links worden opgeslagen en hoe deze via de Paper en Graph klassen op de pagina worden getekend.


WCT functies
WorkflowComponent
+ id: String (unique)
+ get(field): object
+ set(field, value)
Document
+ statusObjects: Dict<Status>
+ transObjects: Dict<Transition>
0..n
Stores
Create
Extends
Extends
Status
+ name: String
+ phase: String
+ viewRights: Array<Role>
+ editRights: Array<Role>
Transition
+ label: String
+ mode: String
+ condition: Condition
+ transitionRights: Array<Role>
1
bevat
0..1
Role
+ name: String
+ condition: Condition
Condition
+ name: String
+ field: String
+ value: String
+ validation: String
Workflow Controller
+ createStatus()
+ createTransition()
+ showCellData(cell)
Salesforce Controller
+ loadWorkflow()
+ saveWorkflow(diagram)
+ importWorkflow(xml)
+ exportWorkflow(diagram)
Salesforce Database
1
1..n
Contains
Contains
Create
jointjs
Element
+ status: Status
+ markup: svg
+ updateBox()
Link
+ transitions: Array
+ addTransition()
+ removeTransition(id)
+ addVertex(x, y)
Graph
+ addCell()
+ onCellAdd()
+ onCellRemove()
Has
0..n
Extends
Extends
Cell
+ id: String (unique)
+ renderCell()
+ removeCell()
Renders
0..n
Paper
+ container: Div
+ size: Dimension
+ onMouseEvent(event)

Bijlage D - Onderzoeksrapport

1. Inleiding

In dit document zal worden beschreven welke diagram-tekentool is gekozen als basis voor de workflow-configuratietool. De opzet van workflows is vergelijkbaar met die van flowcharts, maar bevatten veel meer informatie dan gewoonlijk in een flowchart voor zou komen. De workflow moet niet alleen getekend worden, maar moet ook worden ingevuld met informatie uniek aan de software van Nétive. Een tekentool zoals Microsoft Visio zal daarom op zichzelf nooit voldoende zijn om een volledige workflow op te stellen, maar zou op een of andere manier moeten worden uitgebreid om de complexiteit van deze workflows te faciliteren.

2. Context

Workflows leggen het proces vast dat een stuk informatie moet doorlopen binnen een systeem. Deze informatie kan bijvoorbeeld bestaan uit een opstelling van een vacature. De bijbehorende workflow legt dan vast welke stappen moeten worden ondernomen om de vacature te kunnen publiceren, welke worden afgedwongen in het systeem van de klant. Het doel van de workflow-configuratietool is om het mogelijk te maken deze workflows te configureren zonder tussenkomst van een programmeur.
De opzet van workflows is vergelijkbaar met die van flowcharts, maar zij bevatten ook een hoeveelheid informatie die gewoonlijk niet in flowcharts voor zou komen. Deze informatie heeft onder andere betrekking op hoe de workflow doorlopen moet worden en wie welke stappen hierin mag uitvoeren. Voorbeelden hiervan zijn de gebruikersrollen die bepaalde permissies moeten hebben om een deel van het proces in te kunnen zien of om tussen stadia van de workflow te kunnen transitioneren. Ook moeten bijvoorbeeld taken aan een transitie worden gekoppeld, zoals e-mails of wijzigingen in data.

3. Software-eisen

Om in de workflow-configuratietool een workflow te kunnen representeren, moet er bepaalde software aanwezig zijn die een grafische weergave kan bieden van deze workflow in een vorm vergelijkbaar met een flowchart. Ook moeten de objecten in de flowchart zodanig kunnen worden ingevuld dat hierin alle nodige informatie kan worden opgeslagen die vereist is om een volledige workflow op te stellen. Het is dus vereist dat de diagram-tekentool kan worden geconfigureerd of codematig kan worden aangepast om deze eisen te kunnen vervullen. Mocht het nodig zijn om de tekentool codematig aan te passen, moet deze ook open-source zijn.
De workflow-configuratietool moet worden geïntegreerd in de bestaande software-omgeving van Nétive. Deze omgeving bevindt zich op een Cloud-platform van Salesforce, welke is opgebouwd volgens het Platform-as-a-Service-model. Het systeem van Nétive is in een web-omgeving opgebouwd en daarom zal de diagram-tekentool ook in een webapplicatie moeten kunnen worden geïmplementeerd.
De workflow-configuratietool gaat worden gebruikt door mensen zonder technische achtergrond. Daarom is het van belang dat de tekentool gebruiksvriendelijk en eenvoudig te gebruiken is. Zo niet, is het vereist dat deze kan worden aangepast zodat deze niet meer of minder functionaliteit biedt dan dat de gebruikers nodig hebben en kunnen begrijpen.
Verder moet worden gekeken hoeveel werk er nodig zal zijn om de tekentool aan alle functionele eisen te laten voldoen. Een tekentool die out-of-the-box veel functionaliteit biedt is daarom zeer gewenst, zolang deze zijn te configureren aan de hand van de opgestelde requirements.
Ook wil Nétive voor dit project nog geen software-licenties aanschaffen, wat betekent dat er voor een gratis tool of een tool met een proefperiode moet worden gekozen. In het geval van een proefperiode moet worden vastgesteld dat deze niet verloopt voor het eind van het project.

4. Selectiecriteria

Uit de opgestelde eisen voor de diagram-tekentool zijn een aantal selectiecriteria opgesteld. De gevonden diagram-tekentools zijn aan de hand van deze criteria beoordeeld. Voor elk criterium zal een inhoudelijke beschrijving worden gegeven.

Implementatie: De software moet in een webpagina kunnen worden geïmplementeerd in de applicatieomgeving van Nétive. Hiervoor is het voldoende wanneer de software in een afgesloten deel van een webpagina kan worden geplaatst, zolang deze kan communiceren met de Salesforce omgeving waar het zich in bevindt. Om deze reden is ook de taal waarin de tekentool is opgezet niet relevant als criteria. Als een bepaalde tekentool niet in een web-omgeving kan worden ingebouwd, zal de gebruiker waarschijnlijk extra software moeten installeren alvorens zij van de workflow-configuratietool gebruik kunnen maken en dat is geen gewenst scenario.

Aanpasbaarheid: De software moet zodanig kunnen worden aangepast dat het aan de opgestelde eisen voldoet betreffende de gebruiksvriendelijkheid en informatievoorziening. Het is hierbij niet noodzakelijk dat codematig wijzigingen plaats moeten vinden, zolang er aan de requirements kan worden voldaan. Met deze aanpassingen moet bijvoorbeeld de in te vullen informatie gekoppeld aan een statusobject in de workflow worden uitgebreid. Hiervoor is echter toegang nodig tot de instantie van het object binnen de applicatie. Als een tekentool niet goed genoeg kan worden aangepast, kan worden besloten om bepaalde functionaliteit buiten de tekentool om te bouwen. Dit zou echter betekenen dat de gebruiker meer werk moet verrichten en dat is niet de ideale oplossing.

Tekenfuncties: De software moet out-of-the-box zo veel mogelijk functionaliteit bieden. Hoe minder tekenfunctionaliteit handmatig hoeft te worden toegevoegd, hoe meer tijd er kan worden besteed aan het optimaliseren van de tool voor de gebruiker. Zo is het bijvoorbeeld gewenst dat de gebruiker met de tool zelf objecten aan het diagram kan toevoegen, in plaats van dat deze enkel uit een vooraf opgestelde lijst komen. Als er teveel handmatige ontwikkeling plaats moet vinden zal er minder tijd beschikbaar zijn voor andere onderdelen van de tool en zal de kwaliteit van het product achteruitgaan.

Uitleesmogelijkheden: De software moet niet alleen workflows kunnen opstellen, maar moet deze ook kunnen opslaan en distribueren naar verschillende omgevingen in het Vendor Management Systeem van Nétive. Hiervoor is het noodzakelijk dat een bestand met het workflow-model uit de tool kan worden onttrokken, zodat deze kan worden verwerkt in het systeem. Als een bepaalde tekentool van zichzelf geen uitleesmogelijkheden biedt, zal moeten worden gekeken of deze niet op een andere manier uitgelezen kan worden. Zo niet, zal de tool simpelweg niet gebruikt kunnen worden.

Prijsstelling: Het is niet gewenst dat een licentie voor de software moet worden aangeschaft binnen de termijn van het project. Nétive wil eerst een idee hebben van hoe de workflow-configuratietool er uit gaat zien, voor zij bereid zijn een licentie aan te schaffen. Daarom zal moeten worden gekozen voor een diagram-tekentool met een proefperiode of een die volledig gratis is om te gebruiken. Ook moet worden gekeken of de tool commercieel gebruikt mag worden, aangezien Nétive wenst de configuratietool aan klanten te verkopen.

Licentietype: Nétive wil uiteindelijk de workflow-configuratietool aan hun klanten gaan verkopen. Het is daarom niet gewenst dat de licentie van de gekozen diagram-tekentool weerhoudt deze de software voor commerciële doeleinden mag worden gebruikt.

5. Selectie software

5.1 Longlist

Door te zoeken op het internet en te informeren bij collega's is een lijst van tools opgebouwd die potentieel als diagram-tekentool zouden kunnen worden gebruikt. De software in de hieronder opgestelde longlist is geëvalueerd aan de hand van enkele criteria waaruit direct duidelijk werd of ze geschikt zouden zijn voor de configuratietool. Van tools die in deze fase zijn afgevallen wordt beschreven aan welke criteria zij niet voldeden.

Afgevallen software	
Software	**Criteria waarop zij zijn afgevallen**
Lucidchart	**Aanpasbaarheid:** Onvoldoende uitbreidingsmogelijkheden; **Implementatie:** Is een standalone webapplicatie; **Prijs:** 5 of 9 dollar per maand, proefversie van 7 dagen.
Draw.IO	**Aanpasbaarheid:** Onvoldoende uitbreidingsmogelijkheden; **Implementatie:** Is een standalone webapplicatie.
Visio	**Aanpasbaarheid:** Browser-versie van Visio is niet uit te breiden; **Implementatie:** Desktop- en browser-versie beide standalone applicatie.
Activiti BPMN	**Aanpasbaarheid:** Onvoldoende uitbreidingsmogelijkheden; **Implementatie:** Is een standalone webapplicatie.
JsPlumb	**Prijs:** Eenmalig 3500 AUD, gratis versie is beschikbaar; **Aanpasbaarheid:** Onvoldoende uitbreidingsmogelijkheden.
GoJS	**Aanpasbaarheid:** Onvoldoende uitbreidingsmogelijkheden.
Rappid	**Prijs:** 800 euro per jaar, proefversie van 30 dagen.
Draw2D	**Prijs:** Eenmalig 4,99 euro

In de longlist bevonden zich nog andere diagram-tekentools, namelijk Raphael, FlowchartJS, Mermaid, D3JS, MxGraph, JSPlumb CE en JointJS. Deze tools waren bij deze beoordelingsronde nog niet afgevallen en zijn verder bestudeerd.

5.2 Shortlist

Van de overgebleven tekentools uit de longlist is onderstaande shortlist opgebouwd. Deze tools zijn verder bestudeerd en getest in een webapplicatie. Via deze tests kon worden bepaald welke functionaliteit deze software in de praktijk kon bieden en hoeveel werk er nodig zou zijn om deze aan te passen aan de doelen van het project.
Van alle tools in de shortlist worden hieronder de relevante selectiecriteria beschreven. Van deze tools is reeds vastgesteld dat zij in een webpagina kunnen worden geïmplementeerd en dat zij gratis kunnen worden gebruikt binnen het project. Uiteindelijk wordt er een besluit genomen welke tool het beste in de scope van het project past.

Raphael:
Aanpasbaarheid: Raphael is software gebouwd op laag niveau en is daarom fundamenteel goed uit te breiden. Er is echter niet veel om aan te passen aangezien de tool niet over veel functionaliteit beschikt.
Tekenfuncties: Raphael kan statische diagrammen tekenen, maar deze kunnen niet dynamisch worden aangepast of uitgebreid. Gezien de aanpasbaarheid van de software zou deze functionaliteit handmatig geprogrammeerd kunnen worden.
Uitleesmogelijkheden: Raphael biedt van zichzelf geen uitleesmogelijkheden.
Licentietype: Raphael is beschikbaar onder de Open Source MIT License en mag voor commerciële doeleinden worden gebruikt.

FlowchartJS:
Aanpasbaarheid: FlowchartJS kan lastig worden aangepast omdat deze gebruik maakt van markdown language om diagrammen vast te leggen, in plaats van een object-georiënteerde aanpak.
Tekenfuncties: FlowchartJS kan statische diagrammen tekenen maar beschikt niet over functies om dynamisch diagrammen te tekenen. Het zou veel werk kosten om deze functionaliteit handmatig toe te voegen.
Uitleesmogelijkheden: FlowchartJS biedt van zichzelf geen uitleesmogelijkheden.
Licentietype: Raphael is beschikbaar onder de Open Source MIT License en mag voor commerciële doeleinden worden gebruikt.

Mermaid:
Aanpasbaarheid: Mermaid kan lastig worden aangepast omdat deze gebruik maakt van markdown language om diagrammen vast te leggen, in plaats van een object-georiënteerde aanpak.
Tekenfuncties: Mermaid kan statische diagrammen tekenen maar beschikt niet over functies om dynamisch diagrammen te tekenen. Het zou veel werk kosten om deze functionaliteit handmatig te programmeren.
Uitleesmogelijkheden: Mermaid-diagrammen kunnen met SVG-bestanden worden geëxporteerd.
Licentietype: Mermaid is beschikbaar onder de Open Source MIT License en mag voor commerciële doeleinden worden gebruikt.

D3JS:
Aanpasbaarheid: D3JS kan lastig worden aangepast omdat deze een zeer hoog-niveau tekentool is. Het biedt niet de mogelijkheid om objecten in het diagram functioneel aan te passen of uit te breiden.
Tekenfuncties: D3JS beschikt enkel over functionaliteit om statische diagrammen te maken en niet om dynamisch objecten toe te voegen. Het zou te veel werk kosten om deze functionaliteit handmatig toe te voegen.
Uitleesmogelijkheden: D3JS biedt van zichzelf geen uitleesmogelijkheden.

Licentietype: D3JS is beschikbaar onder de Open Source BSD License en mag voor commerciële doeleinden worden gebruikt.

P5JS:
Aanpasbaarheid: P5JS is goed uit te breiden doordat er toegang wordt gegeven tot de html-objecten in het diagram.
Tekenfuncties: P5JS kan statische diagrammen tekenen maar beschikt niet over functionaliteit om dynamisch diagrammen te maken. Gezien de aanpasbaarheid van de software zou deze functionaliteit handmatig geprogrammeerd kunnen worden.
Uitleesmogelijkheden: P5JS kan diagrammen als JSON-bestand of als afbeelding opslaan.
Licentietype: P5JS is beschikbaar onder de Open Source GNU LGP License en mag voor commerciële doeleinden worden gebruikt.

MxGraph
Aanpasbaarheid: MxGraph is een laag-niveau tekentool en biedt volledige toegang tot de objecten in het diagram. Deze software is daarom zeer aanpasbaar.
Tekenfuncties: MxGraph bevat een groot aantal functies om dynamische diagrammen te tekenen, samen met functies voor event-afhandeling en keyboard-input.
Uitleesmogelijkheden: MxGraph biedt de mogelijkheid om diagrammen in XML-formaat uit te lezen.
Licentietype: MxGraph is beschikbaar onder de Open Source Apache 2.0 License en mag voor commerciële doeleinden worden gebruikt.

JointJS
Aanpasbaarheid: JointJS is een tool die volledig toegang biedt tot de onderliggende architectuur van het diagram en die het mogelijk maakt om zelfgemaakte Javascript objecten aan de diagram-onderdelen toe te voegen. Deze software is daarom zeer aanpasbaar.
Tekenfuncties: JointJS komt geleverd met vele functies om dynamische diagrammen te tekenen en bevat ingebouwde event-afhandeling.
Uitleesmogelijkheden: Diagrammen in JointJS kunnen in zowel XML- als JSON-formaat worden uitgelezen.
Licentietype: JointJS is beschikbaar onder de Open Source Mozilla Public License en mag voor commerciële doeleinde worden gebruikt.

5.3 Geselecteerde software

JointJS is de diagram-tekentool die middels dit onderzoek is gekozen als basis voor de workflow-configuratietool. Alle tekentools in de shortlist zijn in de praktijk getest om te zien welke functionaliteit en aanpassingsmogelijkheden zij boden en JointJS kwam hierbij naar voren als de tool met het meeste potentieel.

Verschillende tools boden enkel een manier om figuren in een diagram te tekenen, maar bevatten geen functionaliteit om deze te verslepen, toe te voegen, of anderzijds aan te passen. In theorie was het mogelijk om deze functionaliteit zelf te programmeren, maar het voordeel van JointJS was dat het al over deze functionaliteit beschikte.

Andere tools beschikte ook over functionaliteit om dynamisch diagrammen te tekenen en te bewerken, maar hadden weer als nadeel dat zij niet genoeg mogelijkheden hadden om aangepast te worden. JointJS daarentegen is compleet open source en beschikt over verschillende opties om zelfgemaakte elementen aan het diagram toe te voegen die nog steeds van de beschikbare JointJS functionaliteit gebruik kunnen maken.

Van alle geëvalueerde tools die in een webapplicatie konden draaien en open source waren, had JointJS de beste balans tussen de beschikbare functionaliteit en de aanpassingsmogelijkheden voor verdere ontwikkelingen. Dit is ideaal voor dit project aangezien er maar beperkte tijd is om de workflow-configuratietool te ontwikkelen. De documentatie van JointJS is goed en er is een actieve community rondom deze software die op verschillende fora vragen van andere ontwikkelaars beantwoord.

Een ander interessant aspect van JointJS is dat er ook een betaalde variant van bestaat, genaamd Rappid. Rappid is in wezen een functioneel uitbreidingspakket op de gratis JointJS, welke functionaliteit biedt voor onder andere wijzigingsmanagement en keyboard- afhandeling. Mocht ooit de workflow-configuratietool in gebruik worden genomen, kan worden besloten om de Rappid software aan te schaffen zodat minder tijd hoeft worden besteed aan verdere ontwikkelingen van de tool.

6. Bronnen

Lucidchart (Lucid Software)	https://www.lucidchart.com/
Draw.IO (JGraph Ltd.)	https://www.draw.io/
Visio (Microsoft)	https://products.office.com/nl-nl/visio/flowchart-software
Activiti BPMN (Alfresco Software)	https://www.activiti.org/
D3JS (Mike Bostock)	https://d3js.org/
P5JS (Lauren McCarthy)	http://p5js.org/
Rappid (Client.IO)	https://www.jointjs.com/
JsPlumb (jsPlumb Inc.)	https://jsplumbtoolkit.com/
Draw2D (Andreas Herz)	http://www.draw2d.org/draw2d/
Raphael (Dmitry Baranovskiy)	http://dmitrybaranovskiy.github.io/raphael/
FlowchartJS (Adriano Raiano)	http://flowchart.js.org/
Mermaid (Knut Sveidqvist)	https://knsv.github.io/mermaid/
MxGraph (JGraph Ltd.)	https://jgraph.github.io/mxgraph/
GoJS (Northwoods Software)	https://gojs.net
JointJS (Client.IO)	https://www.jointjs.com/opensource

Bijlage E – Testrapport

1. Testopzet

Elke sprint zullen de opgenomen User Stories worden getest. Dit gebeurt voornamelijk aan de hand van functionele tests.

1.1 Acceptatietests

Bij elke sprint planning worden bij alle opgenomen User Stories acceptatiecriteria toegevoegd. Het doel van deze criteria is om voor het programmeren van start gaat al vast te leggen aan welke functionele eisen het betreffende systeemonderdeel moet voldoen. Aan de hand van deze acceptatiecriteria wordt vervolgens een testcase opgebouwd. Hierin wordt beschreven welke stappen moeten worden ondernomen om de functie te testen en welke resultaten er uit deze stappen horen te komen.

1.2 Unit-tests

Waar toepasselijk zijn functies ontwikkeld in de sprints ook getest met unit-tests. Hiervoor is het Javascript test-framework QUnit gebruikt (https://qunitjs.com/). Voor elke unit-test is een verzameling testdata opgezet om in de functie in te voeren en voor elk stuk invoerdata zijn gewenste resultaten opgezet.

2. Testcases

2.1 Acceptatietests

Elke user story is getest op basis van een acceptatietest. Bij elke user story zijn acceptatiecriteria opgesteld in overleg met de scrum-master. Voor elke user story wordt met behulp van een of meer acceptatietests geëvalueerd of deze juist en volledig is gerealiseerd.
De identificatiecode “WCT” geeft de code aan van de User Story in de Backlog. De code “UR” geeft de User Requirements aan in de requirements specificatie en “FR” de Functionele Requirements.

[WCT-16; UR-01; FR-03] Als consultant wil ik beschikbare workflows in de configuratietool kunnen zien. **[WCT-102; UR-02; FR-04, 15] Als consultant wil ik een nieuwe workflow-configuratie kunnen maken.**
Acceptatiecriteria: 1. In een nieuwe workflow moeten een naam, klantnaam en type kunnen worden ingevuld. 2. Een nieuwe workflow-configuratie moet direct te zien zijn in het overzicht. 3. Er moet een lijstje te zien zijn met alle beschikbare workflows.
Acceptatietest #1 voor criteria 1 Precondities: De configuratietool is geopend. Te doorlopen stappen: 1. Klik op het tabblad "Workflow Configurations." 2. Klik op de knop "New." 3. Er verschijnt een venster met invoervelden. Voer hier de naam, workflow-type en bedrijfsnaam in. 4. Klik op de knop "Save." Verwacht resultaat: Op het scherm is de nieuw gemaakte workflow configuratie te zien met de ingevoerde gegevens.
Acceptatietest #2 voor criteria 2 en 3 Precondities: In de configuratietool is een workflow-configuratie aangemaakt. Te doorlopen stappen: 1. Navigeer naar het tabblad "Workflow Configurations." Verwacht resultaat: De workflow-configuratie aangemaakt in test #1 is te zien in het overzicht.

[WCT-5; UR-01; FR-01] Als consultant wil ik workflow-configuraties kunnen openen in de configuratietool.
Acceptatiecriteria: 1. De workflow-configuratie moet via een overzicht in de applicatie kunnen worden ingeladen. 2. Wanneer een configuratie is ingeladen moet deze worden getoond in het tekenveld.
Acceptatietest #1 voor criteria 1 en 2 Precondities: De applicatie is op de beginpagina geopend en er is een workflow-configuratie beschikbaar. Te doorlopen stappen: 1. Klik op het tabblad “Workflow Configurations.” 2. Klik op de naam van een workflow in het overzicht. 3. Klik op de knop “Open Workflow Configurations” Verwacht resultaat: De workflow-configuratietool wordt geopend en de gekozen workflow wordt ingeladen.

[WCT-6; UR-06; FR-04] Als consultant wil ik gewijzigde workflow-configuraties kunnen opslaan in mijn omgeving.

Acceptatiecriteria:
1. Door op de opslaan-knop te drukken moet de huidige workflow-configuratie in het systeem worden opgeslagen.
2. Alle informatie in de workflow moet worden opgeslagen en weer kunnen worden ingeladen.

Acceptatietest #1 voor criteria 1
Precondities: In de configuratietool is een workflow geopend.
Te doorlopen stappen:
1. Maak een aantal wijzigingen in de workflow-tekening.
2. Klik op de knop "Save" in het menu.
Verwacht resultaat: Er wordt een bericht getoond dat aangeeft dat de workflow-configuratie is opgeslagen.

Acceptatietest #2 voor criteria 2
Precondities: In test #1 is een workflow-configuratie succesvol opgeslagen.
Te doorlopen stappen:
1. Navigeer naar het tabblad “Workflow Configurations” in de applicatie.
2. Open de workflow-configuratie die in test #1 is aangepast.
Verwacht resultaat: De wijzigingen die in test #1 zijn gemaakt zijn terug te zien in de configuratietool.

[WCT-39; UR-04; FR-05, 07] Als consultant wil ik workflows die niet met de workflow-configuratietool zijn gemaakt kunnen inlezen.

Acceptatiecriteria:
1. De workflow-configuratietool moet het oude formaat kunnen vertalen naar een workflow van het nieuwe formaat.
2. De nieuwe workflow-configuratie moet grafisch worden weergegeven in de workflow-configuratietool.
3. Als de workflow is geopend moet deze direct zijn aan te passen.

Acceptatietest #1 voor criteria 1, 2 en 3
Precondities: Een bestand met een workflow-configuratie in XML-formaat is beschikbaar.
Te doorlopen stappen:
1. Maak een nieuwe lege workflow-configuratie aan in de applicatie.
2. Open de workflow in de configuratietool.
3. Klik op de knop “Import.”
4. Open in het dialoogvenster het XML-bestand.
Verwacht resultaat: De workflow van XML-formaat is ingeladen en vertaald naar een grafische workflow, met alle bijbehorende gegevens ingevuld in de workflow.

[WCT-101; UR-05; FR-07, 08] Als consultant wil ik een workflow kunnen exporteren naar XML

Acceptatiecriteria:
1. De gebruiker moet op een knop kunnen klikken om de workflow te exporteren.
2. Het formaat van de workflow moet overeenkomen met bestaande workflows van XML-formaat.

Acceptatietest #1 voor criteria 1
Precondities: In de configuratietool is een workflow geopend.
Te doorlopen stappen:
1. Klik in de workflow-configuratietool op de knop “Export.”
Verwacht resultaat: Er wordt een bericht getoond dat aangeeft dat de workflow-configuratie is geëxporteerd.

Acceptatietest #2 voor criteria 2
Precondities: In test #1 is een workflow-configuratie succesvol geëxporteerd.
Te doorlopen stappen:
1. Navigeer naar het tabblad “Workflow Configurations” in de applicatie.
2. Open de workflow-configuratie die in test #1 is geëxporteerd en download de bijbehorende attachment met de XML-code.
Verwacht resultaat: De inhoud van de XML-code komt overeen met de workflow in de configuratietool.

[WCT-3; UR-07; FR-18, 27] Als consultant wil ik statussen kunnen toevoegen aan een workflow.

Acceptatiecriteria:
1. De status moet via de toolbox kunnen worden toegevoegd.
2. De status moet direct in het tekenveld te zien zijn.

Acceptatietest #1 voor criteria 1 en 2
Precondities: In de configuratietool is een workflow geopend.
Te doorlopen stappen:
1. Lokaliseer de toolbox in het tekenveld.
2. Klik op de knop voor het toevoegen van een nieuwe status.
Verwacht resultaat: Er verschijnt een nieuwe, lege status in het tekenveld.

[WCT-7; UR-07; FR-19] Als consultant wil ik statussen kunnen bewerken.
[WCT-1; UR-08; FR-23] Als consultant wil ik rechten kunnen toekennen aan statussen.

Acceptatiecriteria:
1. Statussen moet kunnen worden aangepast met behulp van input-velden die automatisch worden weergegeven en ingevuld wanneer op een status wordt geklikt.
2. De waardes van de status moet kunnen worden aangepast.
3. Aanpassingen in de inputvelden moeten automatisch in het statusobject worden opgeslagen.

Acceptatietest #1 voor criteria 1
Precondities: In het tekenveld is een status beschikbaar.
Te doorlopen stappen:
1. Klik op de status in het tekenveld.
Verwacht resultaat: Er verschijnt informatie over de geselecteerde status in de toolbox.

Acceptatietest #2 voor criteria 2 en 3
Precondities: In de configuratietool is een workflow geopend.
Te doorlopen stappen:
1. Voeg een nieuwe status toe in het tekenveld.
2. Klik op de status.
3. Pas de nieuw verschenen waardes in de toolbox aan.
4. Deselecteer de status door op een leeg gebied in het tekenveld te klikken en selecteer vervolgens dezelfde status opnieuw.
Verwacht resultaat: De waardes van de status konden worden aangepast en zijn behouden toen deze werd gedeselecteerd. Ook is de ingevoerde naam terug te zien in de status in het tekenveld.

[WCT-4; UR-09; FR-30, 33] Als consultant wil ik transities kunnen toevoegen aan een workflow.

Acceptatiecriteria:
1. Een transitie moet kunnen worden aangemaakt door een lijn te trekken tussen twee statussen.
2. De transitie moet direct te zien zijn in het tekenveld.
3. Een status moet met meerdere andere statussen en met zichzelf kunnen worden verbonden.

Acceptatietest #1 voor criteria 1 en 2
Precondities: In het tekenveld zijn twee statussen beschikbaar.
Te doorlopen stappen:
1. Klik met de muis op een van de contactpunten aan een zijde van de status en houdt het ingedrukt.
2. Beweeg de muis nu naar een andere status tot de lijn verbinding maakt met de andere status.
3. Laat de muis los.
4. Sleep een van de twee statussen met de muis door het tekenveld.
Verwacht resultaat: Er verschijnt een nieuwe transitie in het tekenveld. Deze blijft aan de statussen gekoppeld wanneer een van de statussen wordt bewogen.

Acceptatietest #2 voor criteria 3
Precondities: In het tekenveld zijn minstens drie statussen beschikbaar.
Te doorlopen stappen:
1. Trek vanuit één status in het tekenveld verschillende lijnen naar elke andere status.
2. Trek een lijn van een status naar zichzelf.
Verwacht resultaat: Er bevinden zich meerdere lijnen die vanuit één status met verschillende andere statussen zijn verbonden. Ook is er een lijn die een status met zichzelf verbindt.

[WCT-8; UR-09; FR-35, 37, 39] Als consultant wil ik transities kunnen bewerken.
[WCT-92; UR-10; FR-35] Als consultant wil ik invoervelden hebben voor labels, modi en regels in transities.
[WCT-1; UR-10; FR-37, 39] Als consultant wil ik rechten kunnen toekennen aan transities.

Acceptatiecriteria:
1. Transities moeten kunnen worden aangepast met behulp van input-velden die automatisch worden weergegeven en ingevuld wanneer op een transitie wordt geklikt.
2. De waardes van de transitie moet kunnen worden aangepast.
3. Aanpassingen in de inputvelden moeten automatisch in het transitie-object worden opgeslagen.

Acceptatietest #1 voor criteria 1
Precondities: In het tekenveld is een lijn met een transitie beschikbaar.
Te doorlopen stappen:
1. Klik op de lijn in het tekenveld.
Verwacht resultaat: Er verschijnt informatie over de geselecteerde transitie in de toolbox, ingevuld in inputvelden.

Acceptatietest #2 voor criteria 2 en 3
Precondities: In het tekenveld is een lijn met een transitie beschikbaar.
Te doorlopen stappen:
1. Klik op de lijn in het tekenveld.
2. Pas de nieuw verschenen waardes in de toolbox aan.
3. Klik op een leeg gebied in het tekenveld en vervolgens weer op de transitie.
Verwacht resultaat: De waardes van de transitie konden worden aangepast en zijn behouden toen deze werd gedeselecteerd.

[WCT-58; UR-09; FR-34] Als consultant wil ik de richting van een transitie kunnen aangeven.

Acceptatiecriteria:
1. Als op een transitie wordt geklikt, moet een optie te zien zijn om de richting ervan te wijzigen in de toolbox.
2. Transities moeten heen en terug kunnen wijzen.
3. De naam van de beschikbare statussen moet te zien zijn in de opties.

Acceptatietest #1 voor criteria 1
Precondities: In het tekenveld is een lijn met minstens één transitie beschikbaar.
Te doorlopen stappen:
1. Klik op de lijn in het diagram.
Verwacht resultaat: In de toolbox is een selectiemenu te zien waarmee het eindpunt van een transitie kan worden aangegeven.

Acceptatietest #2 voor criteria 2
Precondities: In het tekenveld is een lijn met minstens één transitie beschikbaar.
Te doorlopen stappen:
1. Klik op de lijn in het tekenveld.
2. Wijzig de richting van de transitie in het betreffende selectiemenu.
3. Deselecteer de lijn door op een leeg gebied in het tekenveld te klikken en selecteer dezelfde lijn opnieuw.
Verwacht resultaat: De waarde van de richting is aangepast en in het tekenveld is een pijltje toegevoegd of veranderd.

Acceptatietest #3 voor criteria 3
Precondities: In het tekenveld zijn twee statussen beschikbaar die met een lijn aan elkaar zijn verbonden. De lijn bevat minstens een transitie.
Te doorlopen stappen:
1. Klik op de lijn in het tekenveld.
2. Klik op het selectiemenu om de richting van de transitie aan te passen.
Verwacht resultaat: De namen van de statussen in het selectiemenu komen overeen met de namen van de statussen in het diagram.

[WCT-59; UR-09; FR-31, 32] Als consultant wil ik de informatie van alle transities behorende bij een lijn kunnen zien en bewerken in de toolbox.

Acceptatiecriteria:
1. De gebruiker moet op een knop kunnen klikken om een transitie aan een link toe te voegen.
2. De transitie moet kunnen worden aangepast en verwijderd.
3. Een link met transities in beide richtingen moet een pijltje naar beide kanten krijgen.
4. Er moet automatisch één transitie worden toegevoegd aan een link wanneer deze nieuw wordt aangemaakt.

Acceptatietest #1 voor criteria 1
Precondities: In het tekenveld is een lijn beschikbaar.
Te doorlopen stappen:
1. Klik op de lijn in het tekenveld.
Verwacht resultaat: In de toolbox verschijnt onderaan een knop om een transitie toe te voegen.

Acceptatietest #2 voor criteria 2
Precondities: In het tekenveld zijn twee statussen beschikbaar die met een lijn aan elkaar zijn verbonden. De lijn bevat minstens één transitie.
Te doorlopen stappen:
1. Klik op een lijn in het tekenveld.
2. Wijzig de informatie die zich in een van de transities bevindt.
3. Deselecteer de link door op het tekenveld te klikken en selecteer dezelfde lijn vervolgens opnieuw.
4. Klik nu op de knop om de transitie te verwijderen.
Verwacht resultaat: De ingevuld gegevens zijn bewaren gebleven nadat de lijn was gedeselecteerd en vervolgens is de transitie verwijderd uit de lijn.

Acceptatietest #3 voor criteria 3
Precondities: In het tekenveld zijn twee statussen beschikbaar die met een lijn aan elkaar zijn verbonden. De lijn bevat minstens twee transities.
Te doorlopen stappen:
1. Klik op de lijn in het tekenveld.
2. Verander het eindpunt van een transitie zodat de transities van de betreffende lijn beide kant op navigeren.
Verwacht resultaat: In het tekenveld bevinden zich pijltjes aan beide kanten van de lijn.

Acceptatietest #4 voor criteria 4
Precondities: In het tekenveld zijn twee statussen beschikbaar.
Te doorlopen stappen:
1. Trek een nieuwe link tussen de twee statussen.
2. Klik op de nieuw gemaakte link.
Verwacht resultaat: In de toolbox is een lege transitie te zien.

[WCT-60; UR-09; FR-40] Als consultant wil ik per status meerdere contactpunten hebben waar ik de transities aan kan verbinden.

Acceptatiecriteria:
1. Drie contactpunten per zijde van een status.
2. De lijn tussen statussen moet aan de gekozen contactpunten verbonden blijven.
3. De contactpunten moeten volledig zichtbaar worden wanneer een lijn wordt versleept.

Acceptatietest #1 voor criteria 1 en 2
Precondities: In het tekenveld zijn twee statussen beschikbaar.
Te doorlopen stappen:
1. Trek een lijn van een van de connectiepunten (cirkels) van een status naar die van een andere status.
2. Beweeg een van de statussen door het tekenveld.
Verwacht resultaat: Vanuit elk van de contactpunten op de status kan een lijn worden getrokken. De lijn blijft verbonden aan de gekozen contactpunten.

Acceptatietest #2 voor criteria 3
Precondities: In het tekenveld zijn twee statussen beschikbaar.
Te doorlopen stappen:
1. Trek vanuit een contactpunt van een status een lijn naar een contactpunt van een andere status.
Verwacht resultaat: Terwijl de lijn wordt getrokken, worden de beschikbare contactpunten op de statussen volledig zichtbaar.

[WCT-77; NF-02] Als consultant wil ik kunnen blijven zien welk object in het diagram geselecteerd is.

Acceptatiecriteria:
1. Als een object in het diagram wordt aangeklikt, moet deze worden gemarkeerd met een kleur.
2. Als een object wordt aangeklikt, moet deze voor alle andere objecten komen te staan.

Acceptatietest #1 voor criteria 1
Precondities: In het tekenveld zijn een status en een lijn beschikbaar.
Te doorlopen stappen:
1. Klik op de status.
2. Klik op een leeg gebied in het tekenveld om de status te deselecteren.
3. Klik op de lijn.
4. Klik wederom op een leeg gebied in het tekenveld.
Verwacht resultaat: De status en lijn krijgen een gekleurde markering wanneer erop wordt geklikt en verliezen deze markering wanneer ze worden gedeselecteerd.

Acceptatietest #2 voor criteria 2
Precondities: In het tekenveld zijn twee statussen beschikbaar.
Te doorlopen stappen:
1. Klik op een status.
2. Versleep de status zodat deze zich boven de andere status bevindt.
Verwacht resultaat: De eerst geselecteerde status bevindt zich boven de andere status en niet erachter.

Acceptatietest #3 voor criteria 2
Precondities: In het tekenveld is een status beschikbaar met meerdere lijnen die naar hetzelfde contactpunt verwijzen.
Te doorlopen stappen:
1. Klik op een van de lijnen die naar de status verwijst.
2. Versleep het uiteinde van de lijn aan de kant van de status naar een andere plek in het diagram.
Verwacht resultaat: De geselecteerde lijn wordt door het diagram bewogen en niet een andere lijn die ook naar de status verwees.

2.2 Unit-tests

Voor elke unit-test zijn verschillende testobjecten gemaakt. Deze worden hier weergegeven en uitgelegd.

Nieuw Element

Met deze functie worden nieuwe elementen gemaakt in het tekenveld, met bepaalde standaardwaarden reeds ingevoerd. Er is echter maar één soort nieuw element dat kan worden gemaakt, en dat is een leeg element.

```
Testobject 1:
{
	name: 'base element model',
	from: {},
	make: {
		objtype: 'element'
	}
}

Testobject 2:
{
	make: 'no base element',
	from: null,
	make: {
		objtype: 'element'
	}
}
```

Nieuwe Link

Met deze functie worden nieuwe links gemaakt in het tekenveld. Belangrijk hier is dat bij het maken van een link de bijbehorende identificatie van de statussen wordt meegenomen. Als deze niet wordt meegegeven, moet deze simpelweg leeg blijven.

Testobject 1:

```
{
	name: 'base link model',
	from: {
		ids: {
			s1: 'status_1',
			s2: 'status_2'
		},
		stat1: 's1',
		stat2: 's2'
	},
	make: {
		objtype: 'link',
		source: {
			id: 'status_1'
		},
		target: {
			id: 'status_2'
		}
	}
}
```

Testobject 2:

```
{
	name: 'faulty transition',
	from: {
		ids: {
			s1: 'status_1',
			s2: 'status_2'
		},
		stat1: 'not_s1',
		stat2: 's2'
	},
	 make: {
		objtype: 'link',
		source: {
			id: ''
		},
		target: {
			id: 'status_2'
		}
	}
}
```

Calculeer hoek

Met deze functie wordt de hoek berekend van het link object (met de tangens functie). Het is hierbij van belang dat altijd een getal wordt teruggegeven of een “not-a-number.”

```
Testobjecten met een valide resultaat:
{ height: 0, width: 20, angle: 0 }
{ height: 20, width: 20, angle: 45 }
{ height: 40, width: 20, angle: 63.43 }
{ height: 20, width: 0, angle: 90 }
{ height: 0, width: 0, angle: 0 }

Testobjecten die NaN horen te retourneren:
{ height: 'A', width: 'B', angle: 0 }
{ height: '', width: '', angle: 0 }
{ height: undefined, width: undefined, angle: 0 }
{ height: null, width: null, angle: 0 }
```

String achter String

Met deze functie kan de string worden gevonden achter een andere opgegeven string. Deze functie moet altijd de (subsectie van de) originele string teruggeven.

```
Testobjecten:
{ string: 'aaabbb', after: 'a', becomes: 'aabbb' }
{ string: 'aaabbb', after: 'aaa', becomes: 'bbb' }
{ string: 'ababab', after: 'b', becomes: 'abab' }
{ string: 'aaabbb', after: 'c', becomes: 'aaabbb' }
{ string: 'aaabbb', after: 'AAA', becomes: 'aaabbb' }
{ string: 'aaabbb', after: '', becomes: 'aaabbb' }
{ string: 'aaabbb', after: undefined, becomes: 'aaabbb' }
{ string: 'aaabbb', after: null, becomes: 'aaabbb' }
{ string: '!@#$%', after: '#', becomes: '$%' }
{ string: '', after: 'a', becomes: '' }
```

Verleng Array met Array

Met deze functie wordt een Array verlengd met een andere. Hierbij moet altijd de oorspronkelijke Array worden geretourneerd.

```
Testobjecten:
{ array: [1, 2], ext: [3, 4], becomes: [1, 2, 3, 4] }
{ array: [4, 3], ext: [2, 1], becomes: [4, 3, 2, 1] }
{ array: [1, 2], ext: [1, 2], becomes: [1, 2, 1, 2] }
{ array: [1, 2], ext: [3, ['4']], becomes: [1, 2, 3, ['4']] }
{ array: [1, 2], ext: [], becomes: [1, 2] }
{ array: [1, 2], ext: 'abc', becomes: [1, 2] }
{ array: [1, 2], ext: null, becomes: [1, 2] }
{ array: [1, 2], ext: undefined, becomes: [1, 2] }
{ array: [], ext: [3, 4], becomes: [3, 4] }
```

Bijlage F – Gevolgen voor SSD-Project

Inleiding

In dit document zal worden uitgelegd welke keuzes zijn gemaakt tijdens de ontwikkeling van de workflow-configuratietool en welke gevolgen dit heeft voor de verdere ontwikkeling van het bovenliggende Self Service Desk project.

Wat zit er nú in de Workflows

Deze objecten en variabelen zitten in de bestaande workflows:

- **Statussen** (+Naam en Fase)
  - **Permissies** voor lees- en schrijfrechten op statussen
- **Transities** tussen statussen
  - **Permissies** voor transitierechten op transities
- **Acties** op transities
  - Mail- en export-acties worden met een **Anchor** vastgelegd
  - Taken en veld-update-acties worden in de code vastgelegd
- **Condities** op transities en rechten

Overwegingen

"de gebruiker" zijn S&D-medewerkers en consultants die de WCT gebruiken.
"de klant" zijn gebruikers van het VMS die de workflows in de praktijk gebruiken.

Permissies zijn nu een verzameling van gebruikersrollen (een permissie-set)**.** Wij vinden dat het te veel en overbodig werk is voor de gebruiker om eerst die sets vast te leggen en dan deze sets te koppelen aan rechten. Daarom willen we **afzien van Permissiesets** en de gebruiker enkel uit gebruikersrollen laten kiezen bij het vastleggen van de lees/schrijf/transitierechten.

Anchors worden nu gebruikt om vast te leggen waar de klant in de VMS een actie kan koppelen aan een transitie. Wij vinden dat anchors het systeem onnodig complex maken en willen de klant zelf de macht geven om op elke transitie (afgezien van transities in de backend) acties te kunnen toevoegen. Wij willen dus ook **afzien van Anchors.** De klant kan dan dus zelf kiezen waar hij Mail- en Export-acties en Taken koppelt. Veld-update-acties blijven een meer technische actie en zullen door de gebruiker zelf in de tool worden gemaakt.

Condities worden nu gebruikt om bepaalde rechten of transities alleen onder bepaalde voorwaarden toe te kennen. Wij vinden dat condities te technisch van aard zijn om door de gebruikers op te laten zetten. Daarom houden wij het erbij om de gebruiker uit **een lijst met condities** te laten kiezen die door de developers zelf worden opgezet.

Gevolgen voor de workflow-configuratietool

Permissies zullen simpelweg bekend komen te staan als **Rechten** en enkel bestaan uit een lijst met gebruikersrollen. In de praktijk betekent dit dat de gebruiker bij elke status en transitie alle gebruikersrollen moeten aanvinken die daar rechten toe heeft. Om te voorkomen dat een gebruiker een gigantische hoeveelheid checkboxjes aan moet vinken, moet een systeem worden opgezet om dit proces overzichtelijk, snel, en gestroomlijnd te maken. Om te beginnen willen wij hiervoor het toekennen van deze rechten in een apart scherm, los van de tekentool, opzetten. Er zal een soort matrix te zien zijn waarin, van een bepaalde rechtscategorie als "leesrechten op statussen," alle statussen op een rijtje staan tegenover alle gebruikersrollen waaruit gekozen kan worden. Dit biedt de gebruiker een overzichtelijke weergave van welke gebruikersrollen welke stappen kunnen ondernemen in de workflow.
Er wordt verwacht dat deze matrix veel tijd gaat kosten om te maken. Daarom zal eerst een eenvoudig drop-down-lijst worden gemaakt waarin per status en transitie de rechten aan kunnen worden gevinkt. Dit is minder overzichtelijk voor de gebruiker maar het is dan in ieder geval mogelijk om rechten toe te kennen.

Anchors zullen in de workflow-configuratietool compleet afwezig zijn. In het VMS zal echter een kleine aanpassing vereist zijn om niet meer afhankelijk te zien van anchors voor het koppelen van acties (zie volgend hoofdstuk).

Condities zullen wel in de tool worden opgenomen. De gebruiker zal op een bepaalde rechtscategorie of transitie kunnen klikken om daar vervolgens een transitie aan te kunnen koppelen. Deze condities zullen echter niet door de gebruiker zelf kunnen worden gemaakt, omdat deze naar onze mening te technisch zijn van aard. Daarom zullen deze condities door de developers zelf worden gemaakt en zullen ze door de gebruiker uit een lijstje te selecteren zijn.

Gevolgen voor de Self Service Desk

Permissies in het huidige systeem zijn opgeslagen als verzamelingen van gebruikersrollen. Als wij willen overstappen naar lijsten met gebruikersrollen zullen in bestaande workflows de permissiesets ook moeten worden omgezet naar een lijst van gebruikersrollen. Ook moet het systeem worden omgebouwd om niet meer uit te gaan naar permissiesets maar naar gebruikersrollen. De opzet van gebruikersrollen blijft verder echter hetzelfde.

Anchors zullen in het bestaande systeem worden geschrapt. De klant zal daarentegen in het VMS de optie hebben om een bepaalde transitie aan te klikken om hier vervolgens een actie aan te koppelen. Dit is vrijwel identiek aan het huidige systeem, aangezien de klant daar ook zelf zijn acties moest invullen, alleen heeft de klant nu meer keuze in de transities waar hij acties op kan koppelen, in tegenstelling tot alleen de transities waar een anchor aan is gekoppeld. Wel moet worden voorkomen dat de klant acties kan koppelen aan transities waar hij eigenlijk niks mee te maken heeft, zoals transities van en naar de backend van het systeem en andere geautomatiseerde transities.

Condities zullen door de developers moeten worden opgezet en opgeslagen in het systeem. Deze functionaliteit is daar reeds beschikbaar. Het is enkel vereist dat de gebruiker toegang heeft tot deze lijst en ze kan koppelen aan rechten en transities.

Tot slot

Verder moet er de mogelijkheid zijn voor mensen (waarschijnlijk developers) om templates op te zetten in het systeem, waar de gebruiker vervolgens uit kan kiezen om sneller volledige workflows op te zetten. Dit kan op een wijze vergelijkbaar met hoe nu de XML-bestanden worden opgezet, alleen zijn deze templates natuurlijk volledig door de gebruiker zelf in te vullen.
Uiteindelijk willen we ook bestaande workflows om kunnen zetten naar de standaard die de nieuwe workflows aan gaan houden, maar eerst willen we zorgen dat de nieuwe workflows volledig werken. Dan kunnen we namelijk pas met zekerheid zeggen hoe we de bestaande workflows om moeten bouwen.