Afstudeerverslag Michiel Post

– Het realiseren van een applicatie om meldingen over fouten en wijzigingsverzoeken gestructureerd af te handelen. –

M.R. Post
20011937
IVIT

Qurius ETX
Lange Kleiweg 56b
2288 GK Rijswijk

Qurius ETX

Referaat

Dit is het afstudeerverslag van de afstudeerstage die is uitgevoerd van 7 februari 2005 tot en met 10 juni 2005 door Michiel Post bij Qurius ETX, te Rijswijk. De afstudeerstage is uitgevoerd in het kader van de opleiding Informatica en Informatiekunde aan de Haagse Hogeschool.

De opdracht was het ontwikkelen van de applicatie TrackMonkey.NET. Met deze applicatie is het mogelijk om meldingen over fouten of wijzigingen in een applicatie in aanbouw of in onderhoud, gestructureerd af te handelen.

- ASP.NET
- C#
- Sharepoint
- Webpart
- Smartpart
- Microsoft SQL Server
- UML
- Stored Procedures
- Workflow
- Autorisatie

Voorwoord

In de periode van 7 februari tot en met 10 juni heb ik bij Qurius ETX gewerkt aan mijn afstudeerproject. Dit verslag is bedoeld ter beoordeling van de afstudeerperiode door de examinatoren en de gecommitteerde.

Graag wil ik Qurius ETX bedanken voor de afstudeerplek die ze mij aanboden. Binnen het bedrijf functioneerde Michiel Bruins als mijn opdrachtgever. Ik wil hem bedanken voor de tijd en energie die hij in het project heeft gestoken.
Verder bedank ik Karel van der Lelij voor de goede begeleiding vanuit het bedrijf en de nuttige tips en opmerkingen over mijn eindverslag.

Tot slot wil ik mijn examinatoren Arno Nederend en Theo van Gerwen bedanken voor de begeleiding vanuit school.

Michiel Post

Rijswijk, 9 juni 2005

Inhoudsopgave

Bijlagen

INTERNE BIJLAGEN

1. PLAN VAN AANPAK
2. OPDRACHTOMSCHRIJVING

EXTERNE BIJLAGEN

1. SYSTEEMEISEN
2. SYSTEEM CONCEPT
3. ONDERZOEK MAATWERK OF BESTAAND PAKKET
4. PILOTPLAN
5. PILOTONTWIKKELPLAN
6. FUNCTIONELE STRUCTUUR
7. TECHNISCHE STRUCTUUR
8. GEBRUIKERS HANDLEIDING
9. INSTALLATIE HANDLEIDING
10. BEHEER HANDLEIDING
11. TESTPLAN
12. WORKFLOW

1 Inleiding

Dit afstudeerverslag vormt de afsluiting van het afstudeerproject dat ik heb uitgevoerd bij Qurius ETX. De afstudeeropdracht was het ontwikkelen van de applicatie TrackMonkey.NET. Met deze applicatie is het mogelijk om meldingen, over fouten of wijzigingen in een applicatie in aanbouw of in onderhoud, gestructureerd af te handelen.

Het doel van dit verslag is de examinatoren en de gecommitteerde inzicht te geven in mijn werkzaamheden.

Dit verslag begint met een korte introductie van het afstudeerbedrijf, Qurius ETX, in hoofdstuk twee. In dit hoofdstuk wordt ook verteld wat de aanleiding tot de opdracht was en wat de uiteindelijke doelstellingen van de opdracht waren.

In hoofdstuk drie bespreek ik hoe mijn plan van aanpak tot stand is gekomen, wat de methode en technieken waren en hoe de planning er uit zag.
Vervolgens bespreek ik in hoofdstuk vier het opstellen van de systeemeisen, het systeemconcept en het pilotplan. In het pilotplan wordt vastgesteld welke pilots ik ga uitvoeren. In hoofdstuk vijf bespreek ik vervolgens de ontwikkeling van de eerste pilot.

Na het beoordelen en testen van de eerste pilot, is er een tweede pilot uitgevoerd. De uitvoering van de definitie studie voor de tweede pilot staat beschreven in hoofdstuk zes, daar worden onder andere de systeemeisen en het systeemconcept geüpdate met opmerkingen naar aanleiding van de eerste pilot.

Hoe de tweede pilot tot stand is gekomen staat beschreven in hoofdstuk zeven. De invoering in hoofdstuk acht.

Er wordt afgesloten met een evaluatie van zowel het proces als het product. Tot slot is er nog een begrippenlijst en een literatuurlijst te vinden.

2 Qurius ETX + Opdracht

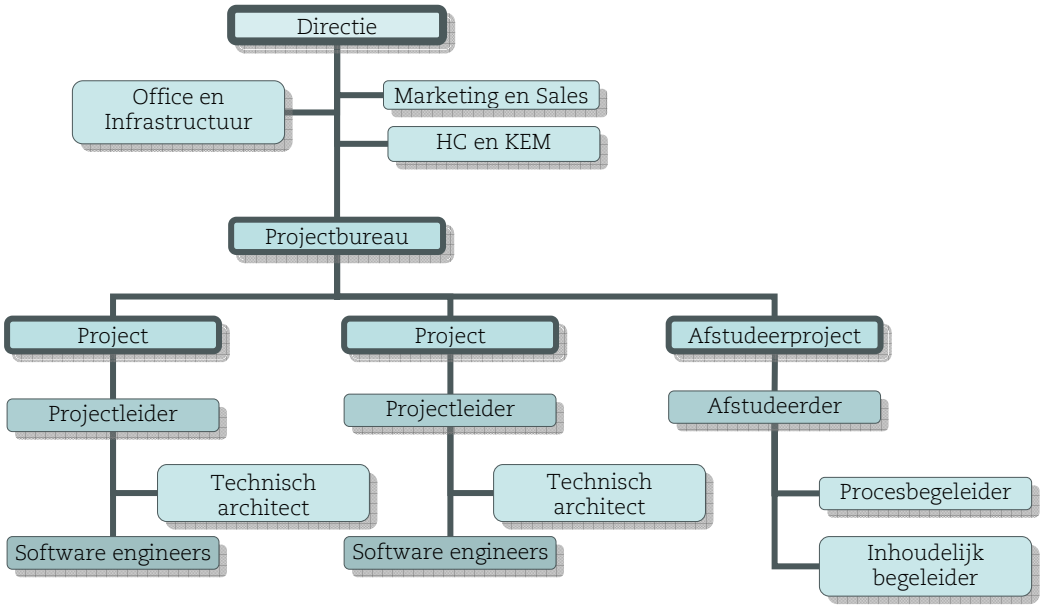
Qurius ETX voert softwareontwikkelingsprojecten uit op het Microsoft platform. Zij hebben een platform ontwikkeld (SoftwareStraat.Net) op basis waarvan zij al hun software ontwikkelen. Ze richten zich met name op de integratie van informatiesystemen, portals en andere e-commerce applicaties.

Qurius ETX heeft vooral middelgrote tot grote klanten. Eén van deze klanten is bijvoorbeeld Wegener. Dit is de grootste uitgever van regionale dagbladen en huis-aan-huiskranten in Nederland. Qurius ETX heeft voor Wegener de website kleintjesmarkt.nl gemaakt. Op deze website kunnen advertenties geplaatst worden. Er is een koppeling ontwikkeld tussen de website kleintjesmarkt.nl en de regionale dagbladen. Door de koppeling is het mogelijk dat ingevoerde advertenties zowel op internet als in de krant verschijnen.

2.1 De organisatie

Qurius ETX wordt geleid door de directie. De directie wordt ondersteund door verschillende afdelingen. Office regelt de algemene zaken binnen Qurius ETX. HC staat voor Human Capital en regelt werving en personeelszaken. De afdeling KEM is verantwoordelijk voor het kennis & ervaringsmanagement. De projecten worden binnengehaald door het marketing/sales bureau. Voor het sturen van de planning en resources van de projecten bestaat het projectbureau. Het projectbureau is ook verantwoordelijk voor de kwaliteitsborging.

Er werken bij Qurius ETX 31 mensen in verschillende projectgroepen. Elke projectgroep heeft een projectleider die het contact met de klant verzorgt. Voor het technisch ontwerp is er voor elke projectgroep een technisch architect. In het organogram staat dit grafisch weergegeven. De technisch architect is verantwoordelijk voor de kwaliteit op project niveau.
Hoewel er relatief veel ondersteunende afdelingen zijn in de organisatie, is het geen bureaucratische organisatie. De reden hiervoor is het feit dat er binnen de ondersteunende afdelingen niet veel mensen werkzaam zijn. Bijvoorbeeld KEM, projectbureau en HC worden allen door één persoon uitgevoerd.



Organogram Qurius ETX

Afstudeerders worden beschouwd als gewone werknemers. Ze werken echter aan een interne afstudeeropdracht en niet aan commerciële projecten.

2.1.1 Workflow

Tijdens de loop van een project heeft de klant continu toegang tot de applicatie die ontwikkeld wordt. Zo kan hij controleren of zijn eisen en wensen juist zijn geïnterpreteerd. Over afwijkingen en gewijzigde inzichten communiceert hij met een projectleider bij Qurius ETX.
Wanneer er een melding, over een fout of wijziging, van een klant of ontwikkelaar binnen komt, besluit de projectleider wat er mee moet gebeuren. Hij wijst bijvoorbeeld een ontwikkelaar aan die deze melding moet afhandelen. Wanneer de ontwikkelaar klaar is met de afhandeling, wordt er een controle uitgevoerd. Na goedkeuring wordt de melding afgesloten. Dit is de workflow die een melding doorloopt.

2.1.2 Trackrecord

Om de workflow geautomatiseerd te laten verlopen, werd het programma TrackRecord gebruikt. Dit programma voldeed echter niet aan de eisen van de opdrachtgever. Het was te groot en te complex. De meeste functionaliteit werd nooit gebruikt. Ook was het voor klanten niet mogelijk zelf meldingen aan te melden. Het programma had maar een beperkte functionaliteit via de web interface.

2.1.3 Sharepoint

Bij de opdrachtgever heeft elke klant zijn eigen Sharepoint portal, waar alle gegevens over projecten van de klant beschikbaar zijn. Een Sharepoint portal is een website vanwaar uit verschillende applicaties te benaderen zijn. Een webpart is een applicatie die binnen de Sharepoint portal weergegeven kan worden. Gebruikers die inloggen op de Sharepoint portal hebben een persoonlijk overzicht van de door hen ingestelde applicaties.
Sharepoint draait op het .NET framework van Microsoft. Dit framework geeft de mogelijkheid om (web)applicaties te programmeren in verschillende talen, zoals C# en VB.NET. Webapplicaties die geschreven zijn in een .NET programmeertaal, kunnen op een Sharepoint server draaien.

2.2 Opdrachtomschrijving definiëren

Het probleem was dat er onvoldoende de mogelijkheid was om meldingen, van klanten of ontwikkelaars, over fouten of wijzigingen in een door Qurius ETX ontwikkelde applicatie vast te leggen. Daarnaast was er geen inzicht in de status van deze meldingen en kon de opdrachtgever de klant er ook geen inzicht in geven.

TrackRecord probeerde dit probleem op te lossen, maar deed dat niet doordat het te groot en te complex was. Ook is er al eens eerder een afstudeerder geweest die heeft geprobeerd dit probleem op te lossen. Dit heeft geen werkend eindproduct opgeleverd.

De opdrachtgever gaf aan de mogelijkheid te willen hebben om meldingen over fouten en wijzigingsverzoeken vast te leggen en de status van zo'n melding bij te houden, om zo klantgerichter te kunnen werken. Wanneer een klant opbelt om te vragen hoe het zit met zijn melding over een fout, kan hem meteen verteld worden wat de status van de melding is. Nog mooier zou het natuurlijk zijn als de klant, via internet, zelf de status van zijn melding kan opvragen.

2.2.1 Doelstelling

De doelstelling van de opdracht was het ontwikkelen van TrackMonkey.NET. Deze applicatie geeft klanten en ontwikkelaars de mogelijkheid om meldingen te doen over fouten en wijzigingsverzoeken in een applicatie in aanbouw of in onderhoud. Deze meldingen worden geregistreerd en gestructureerd afgehandeld door het ontwikkelteam. De workflow die gevolgd dient te worden, moet aanpasbaar zijn. De applicatie moet beschikbaar zijn als 'webpart' binnen de Sharepoint Portal.

Van een foutmelding en wijzigingverzoek moesten verschillende gegevens worden opgeslagen. Er moest een rechtenstructuur aanwezig zijn, die regelde wie welke gegevens van een fout of wijzigingsverzoek mocht zien, wie het mag aanpassen etc.

Ook was het erg belangrijk dat er inzicht kwam in de status van een melding tijdens de afhandeling. Deze status moest door zowel ontwikkelaars als door klanten opgevraagd kunnen worden.

2.2.2 Workflow

Een belangrijk onderdeel van de opdracht was de workflow die meldingen moeten kunnen volgen. Bij de opdrachtgever werd het programma TrackRecord gebruikt om de workflow geautomatiseerd te laten verlopen. De workflow die dit programma kon volgen, moest ook gevolgd kunnen worden door TrackMonkey.NET.

Met behulp van de workflow worden de interne procedures gestructureerd doorlopen. Een melding van een klant over een fout moet bijvoorbeeld door een projectleider worden toegewezen aan een ontwikkelaar. Wanneer de workflow zo geconfigureerd wordt, zal het systeem afdwingen dat de projectleider de ingevoerde melding toewijst aan een ontwikkelaar. Het zal dus niet mogelijk zijn dat bijvoorbeeld een ontwikkelaar de melding aan zichzelf toewijst.
Nadat de melding is toegewezen aan een ontwikkelaar, kan deze de melding oplossen of terugsturen met het verzoek om meer informatie.

3 Plan van aanpak opstellen

Naar aanleiding van de oriëntatie op de opdracht tijdens de opdrachtomschrijving, is het plan van aanpak opgesteld. In het plan van aanpak stond onder andere omschreven welke methode gevolgd ging worden, wat de risico's waren en de planning.

3.1 Methode

Voor dit project heb ik de methode IAD gebruikt. Qurius ETX heeft zijn eigen ontwikkelmethode ontwikkeld, genaamd Softwarestraat.Net. Ik heb er voor gekozen om hier niet mee te werken, omdat ik het belangrijk vond genoeg ervaring te hebben met een ontwikkelmethode om daar goed gebruik van te kunnen maken.

Het project had de eigenschap dat het sterk tijdgebonden was, door de beperkte afstudeertijd. Tevens was veel interactie met de opdrachtgever vereist, zodat alle eisen en wensen in kaart gebracht konden worden. IAD heeft hier uitgebreide mogelijkheden voor.
Tijdens dit project werd er gewerkt met nieuwe technieken zoals ASP.NET. De opdrachtgever gaf aan het begin van het project aan, het belangrijk te vinden om uiteindelijk een bruikbaar eindproduct te hebben en niet slechts een aantal probeersels. Het werken met nieuwe technieken bracht echter het gevaar met zich mee, dat de kans groter werd dat de gemaakte planning niet uitkwam. Door gebruik te maken van de timeboxing mogelijkheden en het incrementeel en iteratief ontwikkelen van IAD, is dit risico beperkt.

3.1.1 Iteratie strategie

IAD werkt met het itereren van een drietal fases; de definitie studie, pilotontwikkeling en invoering. De volgorde van de iteraties bepaalt de iteratie strategie. Er is gekozen voor evolutionair ontwikkelen. Dit houdt in dat alle drie de fases elke iteratie worden uitgevoerd.
Het voordeel van deze strategie is dat tijdens de definitie studie de eisen niet tot in alle detail uitgewerkt hoeven te worden. Er kan relatief snel aan de eerste pilot begonnen worden. Er zullen snel concrete resultaten geboekt worden en dat is wat de opdrachtgever heeft aangegeven graag te willen.
De opdrachtgever bepaalt uiteindelijk of de pilot ingevoerd wordt of dat er nog een pilot nodig is om de functionaliteit uit te breiden voor de pilot ingevoerd gaat worden.

3.2 Communicatie

Vanuit het bedrijf waren er twee begeleiders. Met Karel van der Lelij besprak ik de procesgang en Michiel Bruins functioneerde als mijn opdrachtgever. Met hem communiceerde ik over de opdracht. Hij was aangesteld om de eisen en wensen, die het bedrijf had met betrekking tot de applicatie, aan mij duidelijk te maken. Het was een bewuste keuze van het bedrijf om mij één contactpersoon met betrekking tot de opdracht te geven.

3.3 Risico

In het plan van aanpak heb ik beschreven wat de risico's waren voor dit project. Voordat ik aan dit project begon, had ik nog geen kennis van ASP.NET. Ik wist wat het was, maar had nog geen idee hoe ik het kon gebruiken. Omdat de webpart op Sharepoint kwam te draaien en dus in ASP.NET gemaakt zou moeten worden, lag het voor de hand om dit als risico aan te merken.
Dit risico diende zo veel mogelijk beperkt te worden. Dat is onder andere gedaan door extra tijd vrij te houden tijdens de definitie studie, waarin ik kon onderzoeken of ik de gewenste functionaliteit het beste kon implementeren via maatwerk of een bestaand pakket. Aan het einde van dit onderzoek werd er een keuze gemaakt of ik de gewenste functionaliteit zelf ging implementeren of een bestaand pakket ging inzetten.
Mocht de keuze uiteindelijk uitkomen op maatwerk, dan bleef het risico bestaan dat ik te weinig kennis had van ASP.NET. Ik zou daarom gaan werken met pilots, zodat er aan het einde van de afstudeerperiode altijd een werkend deel van het eindproduct opgeleverd kon worden.

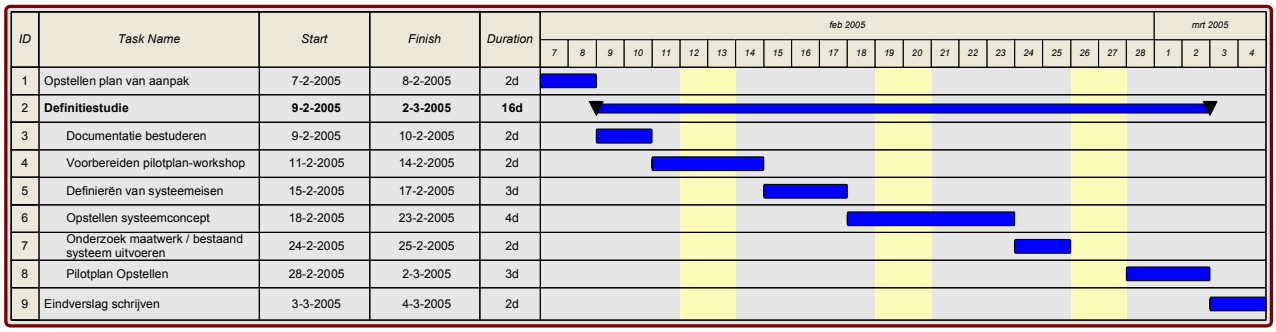
Tijdens het globaal functioneel en technisch ontwerp zou ik me inhoudelijk op ASP.NET concentreren. Tijdens de werkelijke bouw zal er uiteraard ook continu geleerd worden.

3.4 Planning

Tijdens het project is de planning na elke fase vernieuwd met de laatste inzichten. De oude planning, waarmee ik de fase was ingegaan, heb ik wel bewaard. Enkel de toekomstige planningen zijn aangepast.
De oorspronkelijke planning waarmee ik mijn afstuderen begon, is hieronder te vinden. Tijdens het maken van deze planning was er nog geen duidelijkheid over de verschillende pilots.

Activiteit	Dagen
Opstellen plan van aanpak	2
Definitiestudie:	
Documentatie bestuderen	2
Voorbereiden pilotplan-workshop	3
Definiëren van systeemeisen	4
Opstellen Systeemconcept	6
Onderzoek maatwerk / bestaand systeem uitvoeren	2
Pilotplan opstellen	4
Pilotontwikkeling:	
Voorbereiden pilotontwerp-workshop	2
Specificeren van globaal-functionele structuur pilot	4
Specificeren van globaal-technische structuur pilot	4
Pilotontwikkelplan opstellen	2
Documentatie bestuderen	4
Ontwerp software-bouweenheden	7
Bouw software-bouweenheden	15
Opstellen handleidingen pilot	3
Opstellen invoeringsprocedures	1
Beoordelen en testen pilot	5
Invoering:	
Invoeren pilot	3

Voordat ik de eerste fase ben ingegaan, heb ik hiervan een gedetailleerdere planning gemaakt.



ID	Task Name	Start	Finish	Duration
1	Opstellen plan van aanpak	7-2-2005	8-2-2005	2d
2	**Definitiestudie**	**9-2-2005**	**2-3-2005**	**16d**
3	Documentatie bestuderen	9-2-2005	10-2-2005	2d
4	Voorbereiden pilotplan-workshop	11-2-2005	14-2-2005	2d
5	Definierën van systeemeisen	15-2-2005	17-2-2005	3d
6	Opstellen systeemconcept	18-2-2005	23-2-2005	4d
7	Onderzoek maatwerk / bestaand systeem uitvoeren	24-2-2005	25-2-2005	2d
8	Pilotplan Opstellen	28-2-2005	2-3-2005	3d
9	Eindverslag schrijven	3-3-2005	4-3-2005	2d

Planning definitie studie

4 Definitie Studie

De eerste definitie studie werd uitgevoerd na het opstellen van het plan van aanpak. De gevolgde methode tijdens het afstuderen was IAD. IAD begint altijd met de fase definitie studie. In deze fase wordt duidelijkheid geschapen over het doel van het project en de eisen en wensen van de opdrachtgever.
Uiteindelijk zal een pilotplan worden opgesteld, waarin staat beschreven wanneer welke pilot gerealiseerd zal worden en wat de inhoud van de pilot is.

4.1 Systeemeisen opstellen

De eerste activiteit, die ik heb uitgevoerd tijdens de definitie studie, was het onderzoeken van bestaande documentatie met betrekking tot de opdracht. Er was een interne memo beschikbaar, waarin een evaluatie van de huidige workflow in TrackRecord stond beschreven.
Ook was er een timebox rapport van Idde Zijlstra, een afstudeerder die al eerder aan deze opdracht heeft gewerkt. Dit rapport heeft me geholpen om een algemeen beeld te krijgen van de opdracht. Ik heb enkel in de beginfase gebruik gemaakt van dit rapport, omdat al snel bleek dat er te veel verschillen waren tussen de opdracht van Idde en die van mij.

Door het opstellen van de opdrachtomschrijving en het doornemen van de bovengenoemde documentatie heb ik mijn eigen ideeën over het systeem gekregen. Deze heb ik opgenomen in de eerste versie van de systeemeisen. De gedachte hierachter was, dat tijdens de bespreking van de systeemeisen met de opdrachtgever, meteen gereageerd kon worden op mijn ideeën. De opdrachtgever hoeft dan minder nieuwe eisen te vertellen, de meeste stonden al beschreven.

Uit de opdrachtomschrijving was al duidelijk geworden dat de applicatie foutmeldingen en wijzigingsverzoeken moest kunnen afhandelen. Tevens moest er de mogelijkheid zijn om inzicht te krijgen in de status van een melding.

4.1.1 Onderscheid meldingen vaststellen

In de applicatie moesten zowel foutmeldingen als wijzigingsverzoeken kunnen worden ingevoerd. Dit moest mogelijk zijn binnen één applicatie. Ik heb de opdrachtgever gevraagd hoe hij dit onderscheid voor ogen zag binnen de applicatie. Hij gaf aan, dat hij graag zou willen dat het invoerscherm altijd dat van een foutmelding is. Een melding komt dus altijd als foutmelding in het systeem binnen. Later kan er beslist worden dat de melding een wijzigingsverzoek is. Een melding moet dus van soort kunnen wisselen.

4.1.2 Workflow vastleggen

Een melding die ingevoerd is, moet stappen kunnen doorlopen in het systeem. Dit is de workflow. De opdrachtgever legde uit dat een melding steeds een status heeft, plus een aantal mogelijke acties om naar een andere status te gaan. Om de workflow te kunnen doorlopen, was er de eis dat de status van een melding aan te passen is, mits je de juiste rechten in het systeem hebt. De nieuwe status van een melding mag alleen een volgende stap in de workflow zijn. Het mag dus niet mogelijk zijn, om zomaar vanuit een status naar een willekeurige andere status te gaan.
In de interne memo stond duidelijk weergegeven welke workflow werd doorlopen met het huidige TrackRecord systeem. Er stond schematisch weergegeven welke statussen er bestonden en welke keuze uit welke status beschikbaar was. Dit heb ik als uitgangspunt genomen. De toekomstige applicatie moest minstens deze workflow kunnen volgen.

Naar aanleiding van de gesprekken met de opdrachtgever, heb ik het volgende over de workflow geschreven in de systeemeisen:

Een melding doorloopt verschillende statussen. In elke status zijn er één of meerdere rollen die de workflow status mogen wijzigen. Een foutmelding en een wijzigingsverzoek hebben een verschillende workflow. De nieuwe status van een melding mag alleen een volgende stap in de workflow zijn. De workflow zoals te zien is in externe bijlage 12 moet mogelijk zijn om te doorlopen.

4.1.3 Rollen definiëren

In het workflow schema, bij de interne memo, stonden de volgende rollen aangegeven:

- Everyone
- Project Admin
- Development
- Customer
- Test Engineer

Deze rollen kwam uit het oude Trackrecord systeem. In het Timebox rapport van Idde Zijlstra stonden deze rollen aangegeven:

- Projectcoördinator (intern/extern)
- Projectmanager (intern/extern)
- Task manager
- End user

Het opstellen van de systeemeisen heeft onder andere als doel, de juiste actoren te definiëren die gebruik zullen maken van het systeem. Omdat uit de documentatie niet duidelijk werd welke rollen er moesten komen, heb ik dit als “open vraag” in de systeemeisen opgeschreven.
Vervolgens heb ik de systeemeisen, met de open vragen die ik daarbij heb opgesteld, besproken met Michiel Bruins, mijn opdrachtgever. Er is in het gesprek besproken welke rollen er in het systeem beschikbaar moesten komen en wat globaal hun taken waren.
Hierbij is afgeweken van de rollen die in de documentatie beschreven waren. Bij het opstellen van de nieuwe rollen, zijn de rollen die er in het echt bestaan als uitgangspunt genomen. Zo zijn er bij Qurius ETX projectleiders en ontwikkelaars, dit zijn twee rollen geworden. Er zijn geen echte test engineers, daarom is deze rol komen te vervallen. Ook is er geen onderscheid tussen intern en extern, dit onderscheid hoeft er dus in de applicatie ook niet te zijn.
Wel moet het mogelijk zijn voor de klant, om meldingen in het systeem in te voeren en te bekijken. In uitzonderlijke gevallen kan een klant de mogelijkheid krijgen bepaalde stappen in de workflow goed te keuren. Er zijn daarom twee rollen met betrekking tot de klant, namelijk “medewerker klant” en “projectleider klant”.

Over de meldingen in het systeem worden gegevens bijgehouden. De opdrachtgever wilde voor deze gegevens aan elke rol drie soorten rechten kunnen toekennen, dit waren:

- Niet zien
- Zien
- Zien en wijzigen

Deze rechten gelden per veld, zoals naam en prijs. Het is dus mogelijk dat een gebruiker wel de naam van een melding mag zien, maar niet de prijs.

Uit de gesprekken met de opdrachtgever bleek, dat het belangrijk was dat het systeem makkelijk uitbreidbaar en aanpasbaar zou worden. De rechten op velden moesten aanpasbaar zijn. Tevens moest het makkelijk zijn een extra stap in de workflow toe te voegen. Ook de gegevens die over een melding worden opgeslagen moesten makkelijk uitbreidbaar zijn.
Dit heb ik aangemerkt als een belangrijk punt. Wanneer hier niet meteen vanaf het begin rekening mee gehouden zou worden, zou dit later voor problemen kunnen zorgen. Het risico bestond dat de gehele applicatie dan verbouwd zou moeten worden. In onderstaand schema staan deze eisen bij de niet functionele eisen.

Functionele eisen:

	Prioriteit	Naam	Systeem Concept
1	Hoog	Foutmeldingen en wijzigingsverzoeken invoeren	Ja
2	Hoog	Soort wisselen	Ja
3	Hoog	Rollen toekennen aan gebruikers	Ja
4	Hoog	Gegevens melding opvragen	Ja
5	Hoog	Gegevens melding wijzigen	Ja
6	Hoog	Meldingen kunnen verwijderen	Ja
7	Hoog	Melding status wijzigen	Ja
8	Hoog	Overzichten opvragen	Ja
9	Hoog	Zoeken	Ja
10	Middel	Overzichten printen	Ja
11	Middel	Sorteren	Ja
12	Middel	Automatisch invullen velden (datum / naam)	Ja
13	Middel	Controle op invoer bij nieuwe melding of wijziging	Ja
14	Middel	Historie van stappen in de workflow opslaan.	Ja
15	Laag	Wijzigingsgeschiedenis bijhouden van gegevens	Nee
16	Laag	Extra velden toevoegen (GUI)	Nee
17	Laag	Veldrechten aanpassen (GUI)	Nee
18	Laag	Aanpasbare workflow (GUI)	Nee
19	Laag	URL / bijlage toevoegen bij een melding	Nee

Niet functionele eisen

	Prioriteit	Naam	Systeem Concept
1	Hoog	Platform: Sharepoint Portal	Ja
2	Hoog	Configureerbaar webpart (welk overzicht standaard getoond wordt)	Ja
3	Hoog	Andere gegevens opslaan over een wijzigingsverzoek dan over een foutmelding	Ja
4	Hoog	Aanpasbare workflow (via database)	Ja
5	Hoog	Veldrechten aanpassen (via database)	Ja
6	Hoog	Extra velden toevoegen (via database)	Ja
7	Middel	Kolommen in overzicht kunnen aanpassen (via database)	Ja
8	Middel	Sharepoint look and feel	Ja
9	Laag	Aanpasbaar overzicht per klant	Nee

Ik heb er voor gekozen de eisen in een schema te zetten, omdat dit makkelijk te overzien was door zowel mij als de opdrachtgever. Het diende als een samenvatting van de systeemeisen. In de kolom “Systeem Concept” staat of de eis is verwerkt in het systeemconcept.
Elke eis in het schema heeft een uitgebreide beschrijving in de systeemeisen. De systeemeisen zijn te vinden in externe bijlage 1.

4.1.4 Prioriteit vaststellen

In de gesprekken met de opdrachtgever over de systeemeisen, is gelet op de prioriteit van de eisen. De systeemeisen met bijbehorende prioriteit heb ik zichtbaar gemaakt in het schema op de vorige bladzijde. In overleg met de opdrachtgever is bepaald welke eisen, welke prioriteiten kregen.

Eisen met een hoge prioriteit zijn absoluut noodzakelijk om het systeem te kunnen laten draaien. De eisen met de prioriteit “middel” daarentegen, maken het systeem makkelijker in gebruik, maar zijn niet absoluut noodzakelijk. Eisen met een lage prioriteit zijn handig om te hebben, maar zeker niet noodzakelijk.

Zoals weergegeven in het schema, is er onderscheid gemaakt tussen het aanpassen van de workflow via het datamodel en via een user interface in het programma. De opdrachtgever gaf aan het heel erg belangrijk te vinden om de workflow aan te kunnen passen. Ik heb aangegeven bij de opdrachtgever, dat het ontwikkelen van een GUI voor dit onderdeel veel tijd in beslag zou nemen en gevolgen zou hebben voor de andere eisen. Het zou dan niet mogelijk zijn om één van de andere eisen met een hoge prioriteit te realiseren.
Vervolgens heb ik het alternatief aangeboden om rekening te houden met deze eis bij het ontwerp van het datamodel. De workflow is dan wel aanpasbaar via het datamodel, maar niet via een GUI. Een GUI zou er in een later stadium nog aan toegevoegd kunnen worden. De opdrachtgever gaf aan dit plan te ondersteunen, het was namelijk niet wenselijk dat andere eisen een lagere prioriteit zouden krijgen. Het gevolg was, dat er met deze eisen rekening gehouden diende te worden, bij het ontwerp van het datamodel.

4.2 Systeemconcept

Het doel van het systeemconcept is het beschrijven van een oplossingsrichting. De functionele architectuur wordt besproken, die het systeem weergeeft vanuit het gezichtspunt van de gebruiker.

Als eerste heb ik de verschillende actoren omschreven die gebruik maken van het systeem. Deze actoren kunnen acties uitvoeren, use-cases genaamd. De actoren die ik heb omschreven, zijn de rollen die ik eerder heb vastgesteld in de systeemeisen.

Ik heb er voor gekozen om niet voor een gelaagd actorenmodel te gaan. In een gelaagd model kan een child-actor alles wat zijn parent-actor ook kan, plus meer. Dit beperkt de mogelijkheden, het is bij een gelaagd systeem niet mogelijk om de child-actor alles te laten erven behalve één use-case.
De projectleider mag bijvoorbeeld alles wat een ontwikkelaar ook kan. Elke stap in de workflow die een ontwikkelaar mag doen, mag een projectleider ook doen. Wanneer je echter een stap in de workflow wilt maken die alleen de ontwikkelaar mag doen, loop je vast bij een gelaagd model. Dit is niet mogelijk. De opdrachtgever mag die stap ook automatisch maken.
Om dus voor meer flexibiliteit te gaan, staat elke actor op zichzelf.

4.2.1 Use-cases opstellen

Na het definiëren van de actoren ben ik gaan onderzoeken welke use-cases er waren. Dit heb ik gedaan door de functionele eisen uit de systeemeisen te nemen. De functionele eisen zijn de eisen over bepaalde acties. Zo is er bijvoorbeeld de eis dat meldingen ingevoerd en gewijzigd moeten kunnen worden, dit zijn acties. Dit resulteerde uiteindelijk in een negental use-cases.

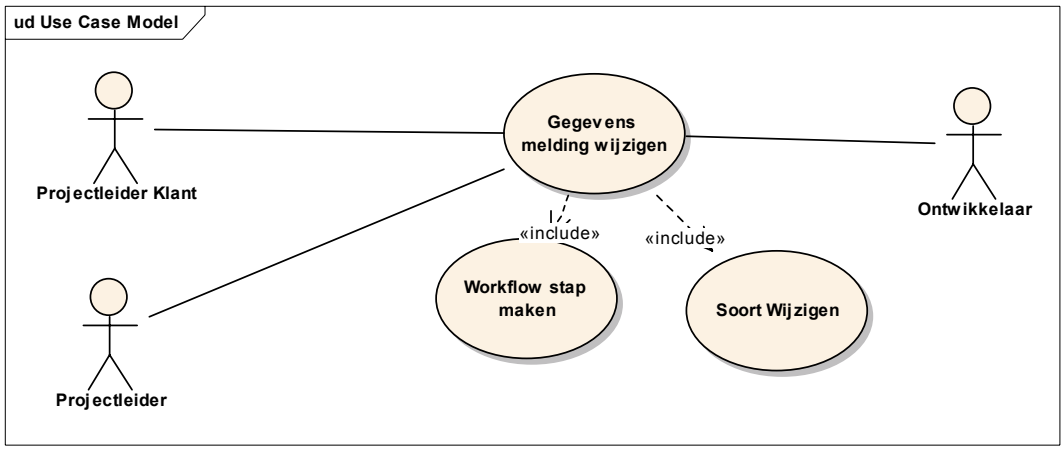
1. Koppelen gebruiker - rol
2. Invoeren melding
3. Gegevens melding opvragen
4. Gegevens melding wijzigen
5. Soort wijzigen
6. Workflow stap maken
7. Overzichten opvragen
8. Zoeken
9. Melding verwijderen

Deze use-cases zijn uitgewerkt in het systeemconcept in externe bijlage 2.

De functionele eisen één tot en met negen zijn use-cases geworden. De eisen tien tot en met veertien zijn onderdeel geworden van de verschillende use-cases.

In het use-case diagram is door middel van een “include” relatie aangegeven dat de use-case “workflow stap maken” en “soort wijzigen” onderdeel is van de use-case “gegevens melding wijzigen”.

Na het vaststellen van de use-cases, ben ik de use-cases met de actoren gaan koppelen. Ook hierbij heb ik de systeemeisen als uitgangspunt genomen. In de systeemeisen stond beschreven wat een actor globaal mocht kunnen. Niet voor elke use-case was meteen duidelijk welke actor deze mocht uitvoeren. Na overleg met de opdrachtgever is het uiteindelijke use-case diagram opgesteld. Het volledige use-case diagram is te vinden in externe bijlage 2. Hieronder is het relevante gedeelte van het use-case diagram te vinden voor de use-case “gegevens melding wijzigen”.



Het use-case diagram van “Gegevens melding wijzigen” met de bijbehorende actoren.

De eerste versie van het systeemconcept was gebaseerd op de eisen met de prioriteit hoog en middel. Eisen met een lagere prioriteit zouden later aan bod komen in een volgende versie van het systeemconcept. Hier heb ik voor gekozen, omdat er anders te veel tijd verloren zou gaan met het omschrijven van functionaliteit die hoogst waarschijnlijk niet geïmplementeerd ging worden tijdens dit project. De opdrachtgever heeft namelijk aangegeven erg graag concreet resultaat te willen zien aan het einde van de afstudeerperiode. Hij zou niet tevreden zijn als ik al mijn tijd zou besteden aan het opstellen van use-cases en aan het einde van mijn afstuderen zou stranden bij het bouwen van de use-cases.
Door het eerste systeemconcept beperkt te houden, hield ik dus meer tijd over voor de andere fases, zoals de pilotontwikkelfase.

De use-cases uit het use-case diagram heb ik uitgewerkt in het systeemconcept. Van elke use-case heb ik kort het doel genoemd, een samenvatting van de functionaliteit, een scenario met mogelijke alternatieven en enkele open vragen, wanneer deze er waren.
Ik neem de use-case “Gegevens melding wijzigen” als voorbeeld, daar heb ik in een scenario de stappen beschreven die de actor en het systeem uitvoeren om tot deze actie te komen. Dit scenario is op de volgende pagina weergegeven.

Bij het uitwerken van de use-cases ben ik dieper gaan nadenken over de functionaliteit van het systeem en hoe deze aangesproken kon worden. Daardoor ontstonden er vragen, die ik geformuleerd heb als open vragen.
Een open vraag bij deze use-case was, of gegevens meteen gewijzigd konden worden bij het bekijken van de detailgegevens of dat dit in een apart scherm moest gebeuren.
Het systeemconcept is vervolgens besproken met de opdrachtgever. Hij gaf als antwoord op de bovenstaande open vraag, dat het wijzigen van de gegevens in een ander scherm moet gebeuren dan het bekijken van de gegevens. Alle antwoorden op de open vragen heb ik verwerkt in een tweede versie van het systeemconcept.

Stap	Actor	Beschrijving
1	Projectleider	In het scherm met detailgegevens over een melding wordt er op de link “Gegevens wijzigen” geklikt.
2	Systeem	Bepaalt aan de hand van de rechten welke velden getoond en gewijzigd mogen worden.
3	Systeem	Bepaalt de status in de workflow en de mogelijke keuzes.
4	Systeem	Toont de mogelijke keuzes voor de workflow status in een drop down box.
5	Projectleider	Wijzigt de gegevens.
6	Systeem	Slaat de nieuwe waarden op van de gegevens.

Use-case scenario voor “gegevens melding wijzigen”

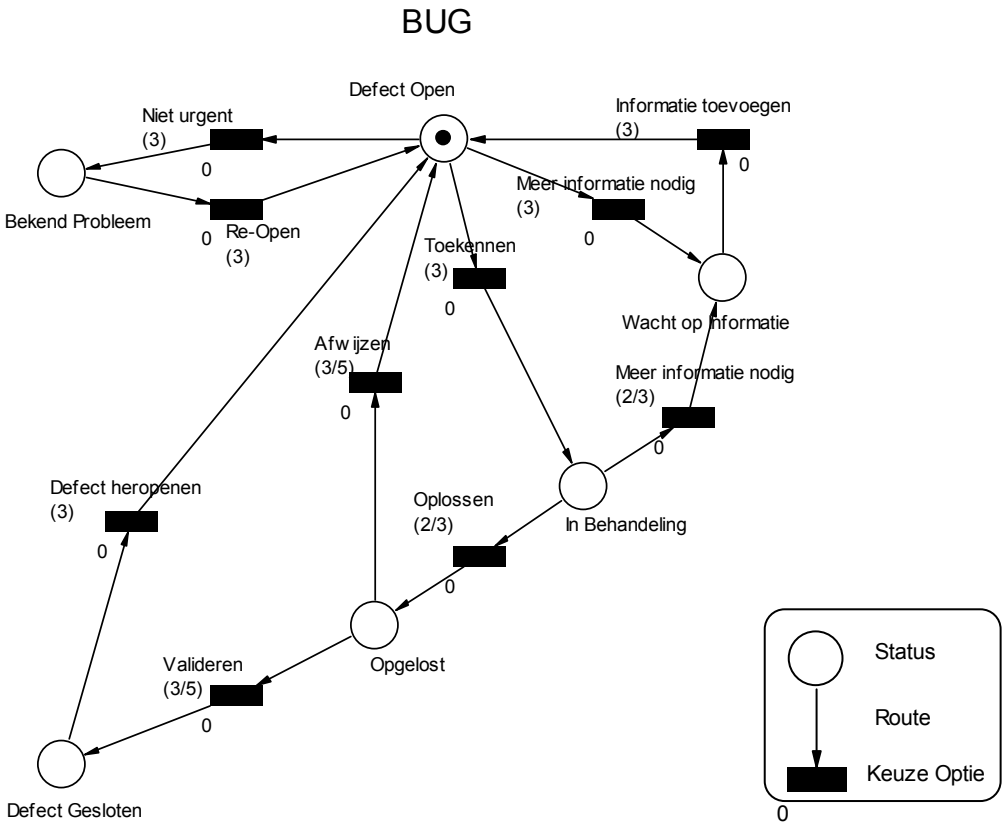
De applicatie moet uiteindelijk alle meldingen in een lijst kunnen weergeven. In het overzicht komen zowel foutmeldingen als wijzigingsverzoeken te staan. Vanuit dat overzicht kan er nog meer informatie over een melding worden opgevraagd door op een melding te klikken. Om een deel van de gegevens te kunnen wijzigen dient er vervolgens op een “wijzig” knop gedrukt te worden.
In hetzelfde scherm waar je de gegevens kan wijzigen, kan er ook een stap in de workflow gemaakt worden. Tevens kan het soort van de melding, een foutmelding of wijzigingsverzoek, gewijzigd worden.

Het volledige systeemconcept is te vinden in externe bijlage 2.

4.2.2 Workflow

In het systeemconcept is een voorstel gedaan met betrekking tot het afhandelen van de workflow. Het doel hiervan was om duidelijk vast te leggen hoe de workflow doorlopen kon worden en wat voor een soort workflow doorlopen kon worden.

Een ingevoerde melding zal altijd in een bepaalde status zijn. Vanuit een status zijn er één of meerdere keuzes beschikbaar om de melding in een andere status te krijgen.



Deel uit de workflow

In het scherm waar de detailgegevens van een melding te zien zijn, zal er een drop down box komen met daarin de mogelijke keuzes. Het is mogelijk om één van de keuzes te selecteren.

Status: Defect Open
Keuze:

- Niet Urgent
- Meer informatie nodig
- Toekennen

Na de keuze voor '*Toekennen*' zal de nieuwe status '*In Behandeling*' zijn. Vervolgens zal het weer mogelijk zijn een keuze te maken en naar een nieuwe status te gaan.

Op deze manier is het mogelijk om alle stappen van de workflow te doorlopen.

Het uiteindelijke systeemconcept vormde een goede basis voor de te realiseren pilots.

4.3 Onderzoek maatwerk of bestaand pakket uitvoeren

Nadat de eisen en wensen aan het systeem duidelijk waren en de functionaliteit beschreven stond in het systeemconcept, ben ik gaan onderzoeken of het systeem het beste gerealiseerd kon worden met een maatwerkoplossing of met een bestaand pakket.
Dit was uiteraard niet het eerste project waarbij er een workflow geïmplementeerd diende te worden. Het was dus mogelijk geweest dat de doelstellingen van het project bereikt konden worden met een bestaand pakket.

Het onderzoek heb ik uitgevoerd aan de hand van vier criteria:

- Kan het de gewenste workflow volgen?
- Prijs
- Complexiteit
- Kan het integreren met Sharepoint?

Aan de hand van het systeemconcept heb ik geanalyseerd wat voor een type workflow het systeem moest kunnen volgen. Het bleek dat de opdrachtgever geen complexe workflow van het systeem vereiste. Het te implementeren systeem moet kunnen werken met statussen en keuzes. Vanuit een status zijn er verschillende keuzes te maken, die leiden tot een nieuwe status.

Dit is anders dan de meer voorkomende workflow, die draait om activiteiten. Bij zo'n workflow wordt er een activiteit uitgevoerd. Het resultaat van die activiteit bepaalt welke volgende activiteiten uitgevoerd mogen worden. De activiteit die op een bepaald moment uitgevoerd wordt, bepaalt de status van een melding. In het systeem is er dan altijd één persoon bezig met een activiteit. Deze activiteit heeft een start en eind tijd.

Op internet heb ik gezocht naar workflow pakketten. De pakketten die er waren, voldeden echter niet voor de opdrachtgever, ze waren te duur en te complex. De meeste functionaliteit zou niet worden gebruikt.
In het onderzoek naar de eisen en wensen van het op te leveren systeem, kwamen een aantal specialistische eisen naar voren. Zoals bijvoorbeeld dat het systeem moet werken als webpart binnen de Sharepoint server en dat er verschillende workflows per klant aangegeven moeten kunnen worden. Dit zou eventueel opgelost kunnen worden door een interface te schrijven tussen een bestaand pakket en de Sharepoint server.

Er was bij de opdrachtgever een licentie voor Biztalk server aanwezig. Met dit pakket zou een workflow geïmplementeerd kunnen worden binnen Sharepoint, maar ook dit pakket was te complex voor wat de opdrachtgever wilde. Het is niet mogelijk met Biztalk slechts een simpele workflow te volgen met statussen en keuzes.

De integratie met Sharepoint van het pakket Biztalk, is waarschijnlijk makkelijker en sneller dan maatwerk software te ontwikkelen. Echter, door de grote complexiteit van Biztalk is dit pakket toch niet geschikt. De opdrachtgever wilde dit namelijk niet.

Tijdens de afweging tussen maatwerk of een bestaand pakket heeft de voorkeur van de opdrachtgever een belangrijke rol gehad. Hierdoor is de keuze uiteindelijk op maatwerk gevallen.

Het volledige onderzoek is te vinden in externe bijlage 3.

4.4 Pilotplan Opstellen

Het pilotplan geeft de pilots weer die ontwikkeld en ingevoerd gaan worden. In deze paragraaf beschrijf ik hoe ik ben gekomen tot mijn pilots en wat de inhoud van de pilots werd.

Bij het opstellen van het pilotplan, heb ik de functionaliteit uit het systeemconcept als basis genomen. Grofweg kon elke use-case in twee delen worden gesplitst, de werkelijke functionaliteit en de verschillende actoren die de use-case mochten uitvoeren.
Het systeem moet namelijk bij elke use-case bepalen wie deze mag uitvoeren. Dit wordt vooral complex bij het opvragen en wijzigen van de gegevens over een melding. Bij die use-cases zal namelijk voor elke actor bepaald worden welke gegevens over een melding hij mag zien en/of wijzigen.

Ik heb ervoor gekozen om in de eerste pilot de belangrijkste functionaliteit met betrekking tot de workflow te realiseren. Er moeten meldingen ingevoerd kunnen worden en deze moeten een workflow kunnen doorlopen. Dit is waar het systeem namelijk om draait. Sommige use-cases zouden dus maar deels geïmplementeerd worden, omdat ik in de eerste pilot nog geen rekening hield met de gedefinieerde actoren en hun rechten in het systeem.

In de tweede pilot zouden er pas actoren komen die verschillende rechten hebben. Dit leek mij een goede volgorde, omdat ik nog geen kennis had van ASP.NET zou het al niet makkelijk zijn om de basis workflow functionaliteit te realiseren. Als ik dan ook meteen rekening had moeten houden met de verschillende rechten van de actoren, werd het te complex.

Het gevolg van deze opdeling in pilots was, dat de eerste pilot niet ingevoerd ging worden. Dit kwam doordat niet alle eisen met een hoge prioriteit werden geïmplementeerd. Ik had er voor kunnen kiezen om mijn pilots anders te verdelen en eerst bijvoorbeeld drie use-cases volledig te implementeren, met rechtenstructuur. Het resultaat van die pilot zou echter ook niet ingevoerd kunnen worden, omdat ook dan niet alle use-cases om het systeem bruikbaar te maken geïmplementeerd zouden zijn.

Welke functionaliteit ik precies ging realiseren, heb ik aan de hand van de use-cases beschreven. Eerst ben ik gaan bepalen welke use-cases ik wilde implementeren. Om dit te doen, ben ik uitgegaan van de samenhang tussen de use-cases, zodat de op te leveren pilot een bruikbare applicatie zou worden.

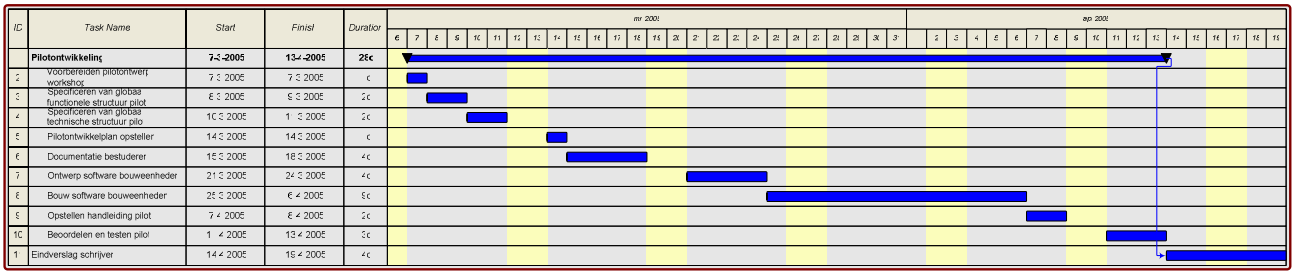
Het pilotplan is te vinden in externe bijlage 4.

4.4.1 Planning

Voor de eerste pilot had ik een maximumtijd van 6 weken vastgesteld. Tijdens de eerste pilot zou er nog veel nieuwe kennis moeten worden opgedaan over ASP.NET, de Sharepoint Portal en Webparts. Dit was tevens het risico van de eerste pilot. De maximumtijd van 6 weken heb ik gebaseerd op een schatting die ik heb gemaakt per use-case. Bij de schatting per use-case bekeek ik op basis van de moeilijkheidsgraad en mijn programmeerervaring, hoe lang ik er over zou doen. De benodigde tijd voor het ontwerp en de bouw van de use-case “gegevens melding wijzigen” schatte ik bijvoorbeeld op 3 dagen.

Het totaal aantal dagen voor de ontwikkeling van alle use-cases was 11 dagen, hierbij kwamen nog twee dagen voor het maken van de webpart bij. Daarnaast heb ik tijd gereserveerd voor de andere activiteiten die uitgevoerd gingen worden, zoals het opstellen van het pilotplan en testplan. Het totaal aantal dagen kwam toen uit op 24 dagen. Om tijdens de eerste pilot genoeg tijd te hebben om kennis van ASP.NET op te doen, heb ik 4 dagen gereserveerd voor het bestuderen van documentatie.

Ik heb er voor gekozen om niet per pilotdeel timeboxing toe te passen, omdat de gehele programmeertaal nog nieuw voor mij was en ik me niet wilde binden aan een te strak schema. Alle functionaliteit uit de eerste pilot had een hoge prioriteit, het moest allemaal gerealiseerd worden. Daarom kon ik bij mijn eerste pilot geen volledige timeboxing toepassen op de gehele pilot. Wel heb ik de grens van maximaal 6 weken als harde limiet gesteld.



Planning eerste pilot

Voor de tweede pilot had ik 4 weken gereserveerd. Mijn verwachting was dat ik dan al veel ervaring opgedaan zou hebben met ASP.NET en tevens de basis van de applicatie gerealiseerd zou hebben. Ik dacht dat ik dan een stuk minder tijd nodig zou hebben om het systeem uit te breiden met een rechtenstructuur. Na de eerste pilot zou ik de planning voor de tweede pilot eventueel nog kunnen aanpassen.

ID	Task Name	Start	Finish	Duration
1	**Pilotontwikkeling**	**29-4-2005**	**25-5-2005**	**19d**
2	Voorbereiden pilotontwerp-workshop	29-4-2005	29-4-2005	1d
3	Specificeren van globaa-functionele structuur pilo	2-5-2005	3-5-2005	2d
4	Specificeren van globaa-technische structuur pilot	4-5-2005	5-5-2005	2d
5	Pilotontwikkelplan opsteller	6-5-2005	6-5-2005	1d
6	Ontwerp software bouweenheden	9-5-2005	11-5-2005	3d
7	Bouw software bouweenheder	12-5-2005	18-5-2005	5d
8	Opstellen handleiding pilot	19-5-2005	19-5-2005	1d
9	Opstellen invoeringsprocedures	20-5-2005	20-5-2005	1d
10	Beoordelen en testen pilot	23-5-2005	25-5-2005	3d

Planning van de tweede pilot

4.4.2 Overzicht

In onderstaand schema is een overzicht te zien van alle functionaliteit en of deze geïmplementeerd zou gaan worden in de eerste of tweede pilot.

Use-Cases:

	Naam	Pilot 1	Pilot 2
1	Koppelen gebruiker - rol		Volledig
2	Invoeren melding	Beperkt	Volledig
3	Gegevens melding opvragen	Beperkt	Volledig
4	Gegevens melding wijzigen	Beperkt	Volledig
5	Soort wijzigen	Beperkt	Volledig
6	Workflow stap maken	Beperkt	Volledig
7	Overzichten opvragen	Beperkt	Volledig
8	Zoeken		Volledig
9	Melding verwijderen	Beperkt	Volledig

Systeem eisen:

	Naam	Pilot 1	Pilot 2
1	Andere gegevens opslaan over een wijzigingsverzoek dan over een foutmelding	Volledig	
2	Aanpassen workflow	Volledig	
3	Veldrechten aanpassen		Volledig
4	Extra velden toevoegen	Volledig	
5	Mogelijkheid om de webpart te koppelen aan één specifiek soort / klant / (zodat enkel meldingen van die klant / soort / zichtbaar zijn).	Volledig	
6	Draait op Sharepoint	Volledig	
7	De layout van het systeem zal overeenkomen met de Sharepoint layout.	Volledig	

- Beperkt betekent dat de functionaliteit beperkt gerealiseerd zal worden, bij de eerste pilot zal dit vooral betekenen dat er nog geen rekening wordt gehouden met de rechtenstructuur.
- Volledig betekent dat de functionaliteit volledig gerealiseerd zal worden zoals deze is omschreven in het systeemconcept in externe bijlage 2.

Dit schema is afgeleid van het eerdere schema met daarin de eisen en wensen, zie paragraaf 4.1.3. Zo zijn alle functionele eisen met een hoge prioriteit use-cases geworden. Een aantal niet functionele eisen, zoals dat de historie van stappen wordt bijgehouden, is verwerkt in de use-cases. De rest van de niet functionele eisen staat in dit schema als systeemeis.

5 Pilot 1: Workflow pilot

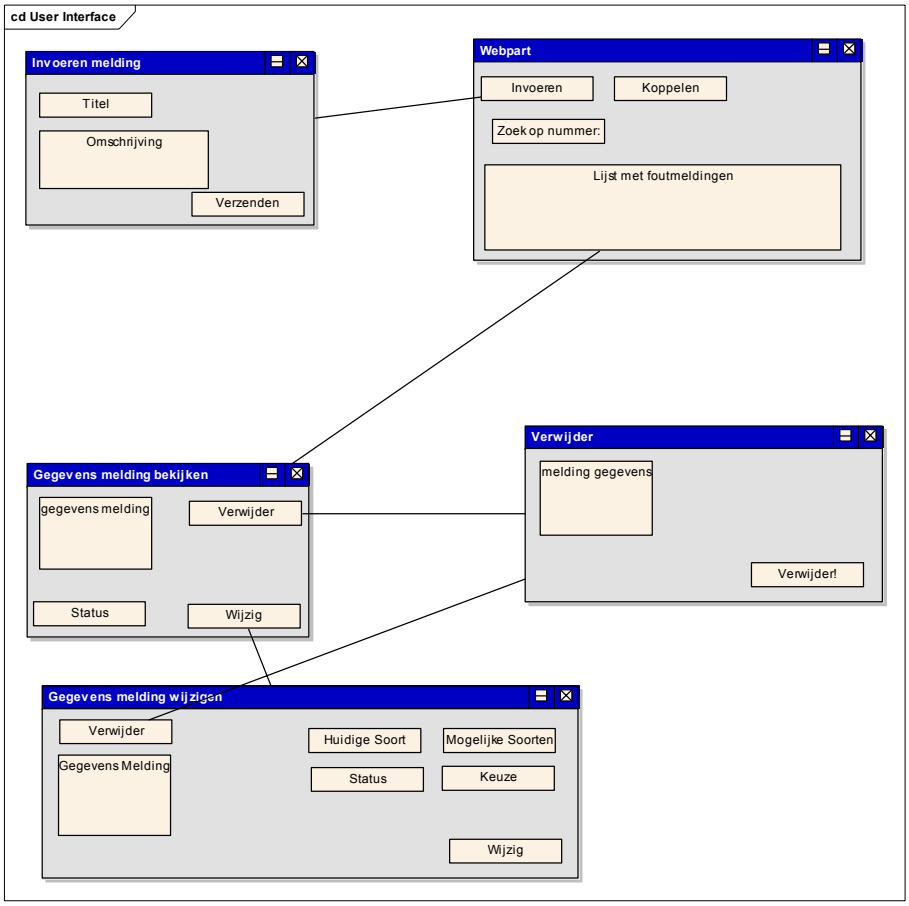
De eerste pilot was bedoeld voor het realiseren van de belangrijkste functionaliteit met betrekking tot de workflow. De te realiseren functionaliteit stond beschreven in het pilotplan dat was opgesteld aan het einde van de definitie studie.
Ik zal beginnen met beschrijven hoe ik de globale en technische structuur heb opgesteld. Vervolgens bespreek ik het opstellen van het pilotontwikkelplan. Hierna wordt er overgegaan op het ontwerp en de bouw van de softwarebouwstenen, die zijn vastgesteld in het pilotontwikkelplan.

5.1 Globaal Functionele structuur opstellen

Het doel van het opstellen van de globaal functionele structuur, is de functionaliteit van de eerste pilot te beschrijven op een zodanige manier dat er een degelijke basis voor de bouw van de pilot is.

In de globaal functionele structuur heb ik onder andere een schema gemaakt van de schermen in het systeem. Dit schema is afgeleid van de use-cases in het systeemconcept. Voor elke use-case staat in een scenario beschreven hoe deze bereikt kan worden. Aan de hand van het scenario is de relatie tussen de schermen opgesteld in onderstaand schema. In dit schema staan enkel de schermen weergegeven die betrekking hebben op de use-cases van de eerste pilot.

Een lijn van een knop naar een scherm betekent dat dit scherm wordt aangeroepen nadat er op de knop is geklikt.



Gebruikersinterface concept

De gehele functionele structuur is te vinden in externe bijlage 6.

5.2 Globaal technische structuur opstellen

In de globaal technische structuur heb ik onder andere het datamodel beschreven en zijn er keuzes gemaakt over de implementatie op de Sharepoint server.

5.2.1 Datamodel vastleggen

De eerste versie van het datamodel was gebaseerd op het systeemconcept en de systeemeisen. Ik ben gaan analyseren welke gegevens opgeslagen zouden moeten worden, om de functionaliteit uit het systeemconcept te kunnen realiseren.
Dit heb ik gemodelleerd in een klassendiagram, om een duidelijk overzicht te krijgen van de klassen en de relaties.

Het klassendiagram kon grofweg worden opgedeeld in drie delen:

1) Workflow
2) Meldingen
3) Gebruikers / Rechten

Ondanks dat ik er voor gekozen had om de gebruikers en hun rechten pas te implementeren in de tweede pilot, hield ik er wel rekening mee tijdens het ontwerp van het datamodel. Wanneer ik dat niet had gedaan, was het mogelijk geweest dat er in de tweede pilot naar voren kwam dat mijn datamodel sterk moest veranderen. Dit zou weer gevolgen hebben voor de opgeleverde eerste pilot. Deze zou dan waarschijnlijk veel aanpassingen moeten ondergaan voor hij uitgebreid kon worden met de nieuwe functionaliteit. Dit zou zonde zijn van de tijd.

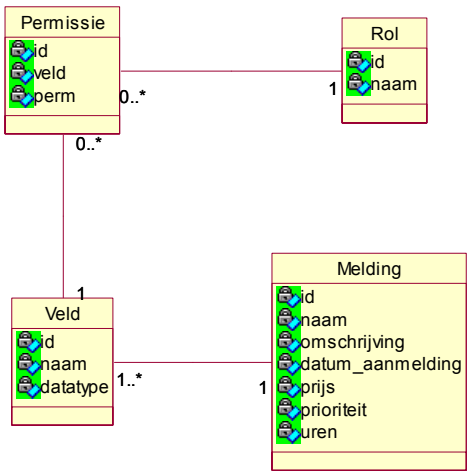
Aan de hand van voorbeeld gegevens bespreek ik twee problemen die ik tegenkwam bij het opstellen van het datamodel.

Probleem: melding met bijbehorende gegevens
Alle drie de delen van het datamodel hadden andere eisen die soms lastig te combineren waren.
Eisen aan het datamodel, vanuit de systeemeisen, waren bijvoorbeeld:

- Welke gegevens (velden) over een melding bijgehouden worden, moet makkelijk uitbreidbaar zijn.
- Gebruikers met een rol moeten bepaalde rechten kunnen krijgen op de velden die bijgehouden worden over een melding.

Als de velden bij een melding dus uitbreidbaar moeten zijn, dan moeten de rechten van de rollen op de velden dit ook zijn. Anders kunnen bepaalde rollen op de nieuw aangemaakte velden geen rechten krijgen.

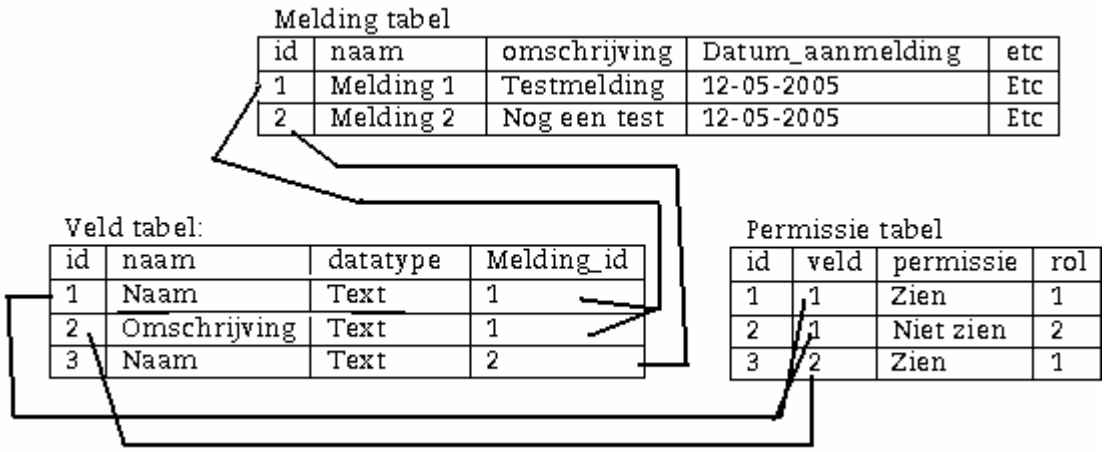
In de eerste versie van het datamodel is een poging gedaan dit te realiseren. Dit is op de volgende bladzijde weergegeven.



Datamodel: poging om het eerste probleem op te lossen

In dit model heeft een Melding meerdere Velden. Een Veld hoort altijd bij één melding. Op een Veld kunnen meerdere permissies gelden, zoals zien en niet zien. Een permissie hoort altijd bij een bepaalde rol.

In dit model zouden de verschillende gegevens van een melding worden opgeslagen bij de klasse Melding, zoals de prijs en de prioriteit.
De klasse Veld houdt bij welke velden er bijgehouden worden van een melding.
De permissie van een bepaald veld wordt bijgehouden bij de klasse Permissie.



Melding tabel

id	naam	omschrijving	Datum_aanmelding	etc
1	Melding 1	Testmelding	12-05-2005	Etc
2	Melding 2	Nog een test	12-05-2005	Etc

Veld tabel:

id	naam	datatype	Melding_id
1	Naam	Text	1
2	Omschrijving	Text	1
3	Naam	Text	2

Permissie tabel

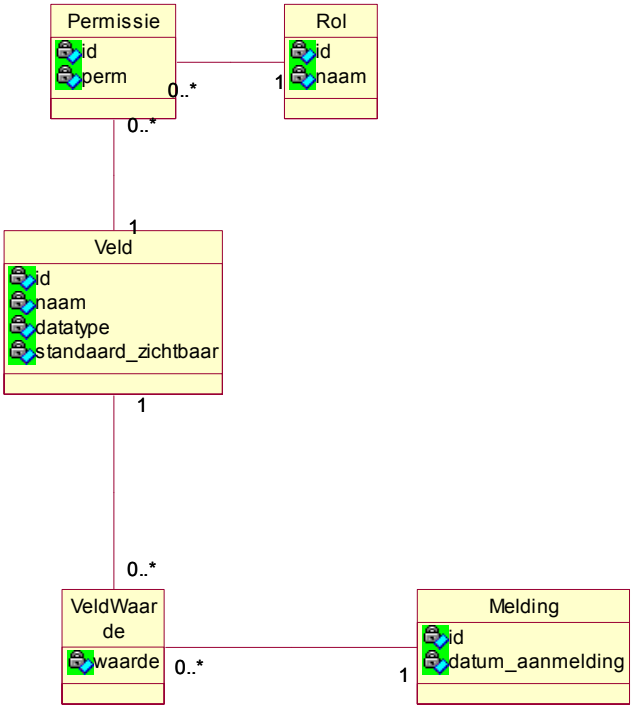
id	veld	permissie	rol
1	1	Zien	1
2	1	Niet zien	2
3	2	Zien	1

Tabellenstructuur van de implementatie van het eerste datamodel

In dit model is er een permissie per veld van elke melding, dit kan veel algemener. Niet elk veld van elke afzonderlijke melding hoeft een andere permissie te hebben. Alle identieke velden van verschillende meldingen mogen dezelfde permissie hebben.
Ook de metadata over een gegeven wordt dubbel opgeslagen. In de tabel melding is er een kolom “naam” en in de tabel Veld wordt, in een tupel, opgeslagen dat het over de kolom “naam” gaat.

De volgende versie had als doel het probleem op te lossen dat er geen echte relatie was tussen Melding en Veld. Ook moeten er gegevens van een Melding bijgehouden kunnen worden. Van deze gegevens moet herleid kunnen worden bij welk Veld ze horen. Welke gegevens er worden opgeslagen voer een Melding, moet makkelijk uitbreidbaar zijn.

Om dit te realiseren is er een extra klasse gekomen die de waarde van elk veld bijhoudt, de klasse VeldWaarde. Deze waarde staat dus niet meer in de klasse Melding. Het model ziet er dan als volgt uit:



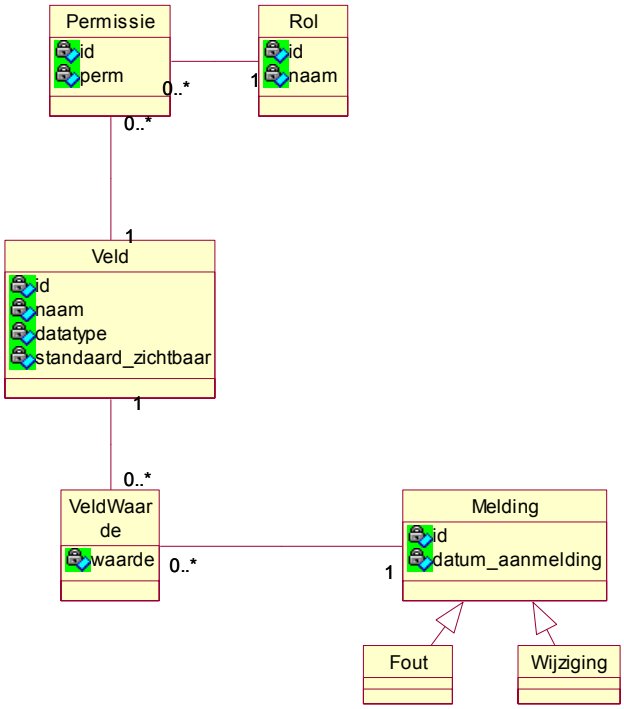
Datamode: oplossing eerste probleem

Een melding kan nu meerdere waarden (VeldWaarde) hebben. Een waarde hoort altijd bij één bepaald Veld (bijvoorbeeld het veld prijs). Op dat Veld kunnen permissies gelden. Het ID en de datum van aanmelding horen nog steeds bij de klasse Melding, omdat deze nooit gewijzigd zullen worden en hier geen permissies op gelden.

In dit model is het wel mogelijk om de velden die over een melding worden bijgehouden uit te breiden. De velden hoeven tevens maar één keer aangegeven te worden bij de klasse Veld. De waarden van elk veld bij een bepaalde melding staat vervolgens bij VeldWaarde. De permissies hoeven nu ook slechts éénmaal aangegeven te worden per Veld. Deze gelden vervolgens voor elke waarde die bij dit veld hoort.

De waarde van een veld wordt opgeslagen als “waarde” in VeldWaarde. Echter niet elke waarde is van hetzelfde datatype. Er kunnen zowel strings (tekst) als integers (getallen) in worden opgeslagen. Het attribuut “waarde” kan niet steeds van datatype veranderen. Om de meeste datatypes te kunnen opslaan is er gekozen om het attribuut “waarde” van het type string te laten zijn. Een integer kan namelijk wel worden gerepresenteerd als string, maar andersom gaat dat niet.
Ik heb er voor gekozen om het werkelijke datatype wel bij te houden. Dit kan later van pas komen bij bijvoorbeeld het valideren van invoer, zodat er geen tekst kan worden ingevoerd als het veld van het type integer is. Ook al zou het theoretisch wel opgeslagen kunnen worden, omdat de waarde altijd als string wordt opgeslagen.
Het werkelijke datatype van een veld wordt opgeslagen bij “datatype” in Veld.

Probleem: foutmelding of wijzigingsverzoek?
Een ander probleem waar ik tegenaan liep, was dat een melding van soort kon wisselen. In het oude TrackRecord systeem, werd dit gezien als een aftakking in de workflow. Dit heb ik daarom ook als uitgangsidee genomen voor het nieuwe systeem. Een melding kon van twee soorten zijn en de wisseling in soort werd gedaan door een bepaalde stap in de workflow. Het datamodel hierbij zag er als volgt uit:



Datamodel: poging om het tweede probleem op te lossen

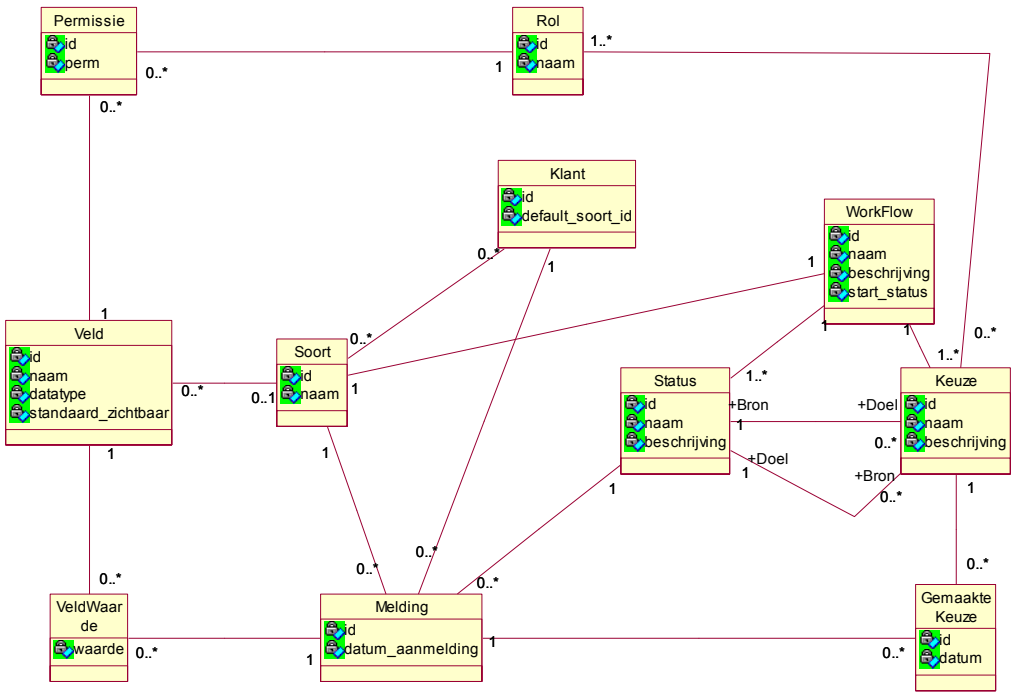
In dit model zijn er twee soorten (Fout en Wijziging) die erven van de klasse Melding.

Het moest mogelijk zijn om van een fout andere gegevens op te slaan dan van een wijziging. Dit kon in dit model niet makkelijk gedaan worden, omdat de klasse Veld geen weet heeft van de verschillende soorten meldingen. Je kan dan dus niet bijhouden welke velden een fout heeft en welke velden een wijziging heeft.

Het wisselen van het soort van een melding was een aftakking in de workflow. Dit was een nadeel, omdat er bijgehouden moest worden welke stap in de workflow de soort wijziging in actie zet. Dit zou als gevolg hebben dat de applicatie minder flexibel zou zijn. Bij uitbreiding met meerdere soorten zou dit erg rommelig worden.

Ik heb er voor gekozen om van het oude TrackRecord idee af te stappen en elke soort melding zijn eigen workflow te geven. Er is een extra klasse Soort aangemaakt, deze klasse heeft een relatie met de klasse Melding, een melding is namelijk van een bepaald soort. Ook heeft Soort een relatie met Veld, zo kan de opdrachtgever aangeven dat een Veld enkel bij een foutmelding hoort en niet bij een wijzigingsverzoek. Soort heeft ook een koppeling met de klasse Workflow, omdat elk soort zijn eigen workflow kan hebben.

Het model kwam er als volgt uit te zien:



Relevante delen uit het datamodel voor de eerste pilot

De workflow werkt met statussen en keuzes. Vanuit een status zijn er keuzes beschikbaar en elke keuze leidt terug naar een nieuwe status. De gemaakte keuzes van een melding worden bijgehouden bij GemaakteKeuze.

Een voordeel van het hebben van de klasse Soort is dat het systeem nu uit te breiden is met meerdere soorten en je niet beperkt bent tot de soorten foutmelding en wijzigingsverzoek.
Daarom heeft Soort ook een relatie met Klant. Een klant kan een aantal mogelijke soorten hebben, zo'n soort kan bij meerdere klanten horen. Door deze relatie kan er voor een klant een nieuw soort worden aangemaakt met bijvoorbeeld een andere workflow. Andere klanten hoeven dit soort niet te gebruiken.

Wat moet er echter gebeuren met de waarde van de velden als een melding verandert van soort? Als een foutmelding een wijzigingsverzoek wordt, is het niet wenselijk dat alle gegevens over deze melding verloren gaan. Maar alles automatisch overzetten gaat ook niet, want er zullen specifieke velden zijn bij een foutmelding, die een wijzigingsverzoek niet heeft.
Om dit op te lossen kon ik gaan bijhouden welke velden, van de verschillende soorten, met elkaar overeenkwamen en bij het wisselen van soort, de waarden van deze velden kopiëren of laten verwijzen naar een ander veld. Het nadeel zou zijn dat je dan moet bijhouden welke velden een overeenkomstig veld hebben bij een ander soort, dit zou erg complex worden bij veel soorten.
Ik heb voor een flexibelere en simpelere oplossing gekozen. De meeste soorten zullen namelijk veel overeenkomstige velden hebben. Al deze velden zullen niet gekoppeld worden aan een specifiek soort. Ze horen als het ware bij elke soort. Bij het wisselen van soort zullen dus dezelfde velden zichtbaar blijven. Alleen de specifieke velden, die slechts bij één soort horen, veranderen.

De keuze om elk soort zijn eigen workflow te kunnen geven en om een veld zonder soort te kunnen aanmaken, maakt het systeem heel generiek en aanpasbaar. Er kunnen op een makkelijke manier extra soorten worden aangemaakt, die een zelfde workflow hebben als een al bestaand soort. Zonder veel moeite heb je een nieuw soort, omdat je niet een geheel nieuwe workflow hoeft in te voeren. Ook de algemene velden, die elk soort heeft, zijn meteen beschikbaar bij het nieuwe soort.

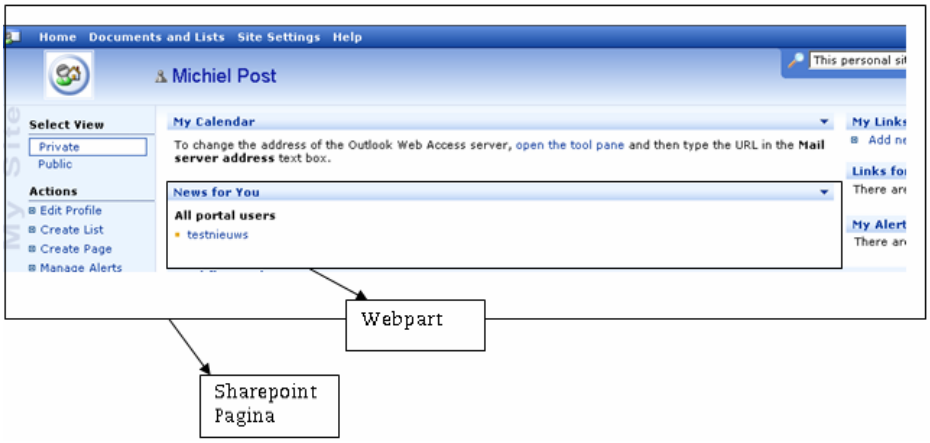
De verschillende modellen zijn elke keer besproken met de opdrachtgever. Ik vertelde hem de mogelijkheden en onmogelijkheden van elk model, waarop hij reageerde met de veranderingen die hij wilde. Zo is uiteindelijk besloten om op basis van dit laatste model verder te gaan.

Van het uiteindelijke datamodel is een relationeel representatiemodel gemaakt, welke te vinden is in externe bijlage 7. Dit is overgezet in een relationeel implementatiemodel, met behulp van Microsoft SQL 2000.

5.2.2 Platform

De applicatie moest als webpart beschikbaar worden binnen een Sharepoint portal op de Sharepoint server. Ik heb onderzocht hoe ik dit het beste kon realiseren. Door te zoeken op internet en de Sharepoint documentatie te bestuderen kwam ik op een aantal mogelijkheden:

1) Webpart
   Een webpart dat in Sharepoint geladen kan worden.
2) Smartpart
   Een smartpart is een webpart waar een Web User Control ingeladen kan worden.
3) Page Viewer Webpart
   Dit is een webpart die een normale pagina kan weergeven. De applicatie zou dan als web applicatie ontwikkeld kunnen worden.
4) Combinatie van bovenstaande
   Bijvoorbeeld een deel als Smartpart en een deel als normale webpagina (Web Applicatie).



Voor de ontwikkeling van de applicatie zou ik gebruik gaan maken van VisualStudio.NET. Dit is een code-editor en heeft de mogelijkheid GUI design makkelijker te maken. Elementen van een pagina, zoals tekstvakken of tabellen, kunnen op de pagina worden gesleept. Het programma zorgt er voor dat de juiste code wordt opgebouwd om de elementen daar te plaatsen.
Optie één heeft echter als nadeel dat webparts anders worden opgebouwd, wat Visual Studio niet ondersteunt. Dit zou bij de eerste optie dus handmatig gebeuren. Als elk scherm handmatig moet worden gemaakt, zal dit veel tijd kosten.

Om toch gebruik te kunnen maken van VisualStudio.NET, bij het maken van een webpart, ben ik gaan zoeken naar andere mogelijkheden. Zo kwam ik op de smartpart terecht. Een smartpart is een webpart waarin een “web user control” te laden is. Een “web user control” is een soort ASP.NET pagina, welke wel te ontwikkelen is met VisualStudio.NET.
De derde optie was om de gehele applicatie als normale webapplicatie te maken en deze vervolgens te laten tonen door het “Page Viewer Webpart”. Dit is een webpart die een normale webpagina kan tonen als webpart. Het nadeel van deze optie was, dat er geen gebruik gemaakt kon worden van webpart specifieke functionaliteit, zoals het meegeven van “custom properties” aan de webpart. Met custom properties is bijvoorbeeld in te stellen voor welke klant de webpart zijn gegevens moet ophalen.

Na het overwegen van bovenstaande opties, ben ik uiteindelijk voor een combinatie gegaan. Om het beginscherm met het overzicht van alle meldingen te tonen, heb ik gekozen voor de smartpart, omdat deze tegenover de Webpart veel tijd bespaarde. Ik heb er ook rekening mee gehouden dat het onderzoeken hoe de smartpart geïntegreerd kon worden met Sharepoint tijd zou kosten. Maar dit zou naar mijn idee minder of gelijk zijn dan de installatie van een normaal webpart met Sharepoint. Bij de keuze voor een normaal webpart zou er nog tijd bijkomen om de GUI te ontwikkelen.
De andere functionaliteit, zoals het wijzigen van gegevens en het invoeren van een melding, kwam buiten de webpart als gewone Web Applicatie. Dan is het gehele scherm beschikbaar om gegevens te tonen en ben je niet beperkt tot het soms kleine webpart scherm.

Mijn keuze, om voor een combinatie van Smartpart en webapplicatie te gaan, heb ik besproken met de opdrachtgever. Hij vond het een goed idee om de Smartpart te gebruiken, maar wilde wel graag weten hoe het zat met de gebruiksrechten van de smartpart. Het bleek dat de smartpart vrij voor commercieel gebruik was, wat het gebruik van de smartpart bij dit project mogelijk maakte.

De gehele technische structuur is te vinden in externe bijlage 7.

5.3 Pilotontwikkelplan Opstellen

Na het opstellen van de globaal functionele en technische structuur, heb ik in het pilotontwikkelplan beschreven wat de pilotdelen en de bouwstenen waren van de eerste pilot. Door het opdelen van de pilot in pilotdelen, kon er een volgorde worden bepaald voor de ontwikkeling. De bouwstenen zijn de onderdelen van een pilotdeel.

Pilotdelen moeten afzonderlijk te testen zijn, daarom heb ik er voor gekozen elke afzonderlijke use-case een pilotdeel te laten zijn.

In het pilotontwikkelplan heb ik de volgorde van ontwikkeling vastgesteld. Hierbij heb ik gelet op de moeilijkheidsgraad van de pilotdelen. Het opbouwen van het overzicht van alle meldingen leek mij erg lastig, omdat dit overzicht wordt opgebouwd uit vier verschillende tabellen en het niet te maken is met een simpele sql-query. Dit probleem wordt verder behandeld in paragraaf 5.5.6.

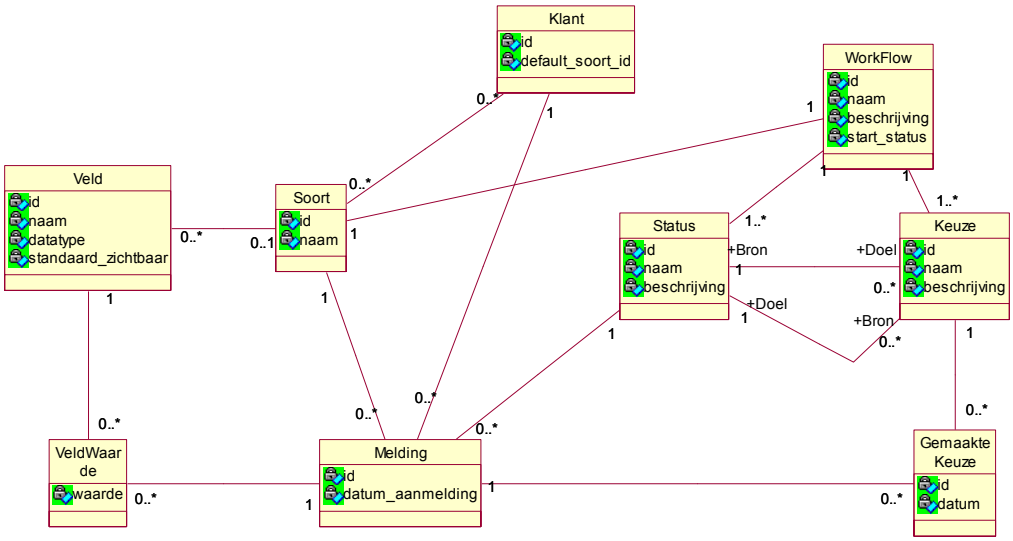
Uit ervaring weet ik dat het makkelijker is om eerst gegevens op het scherm te tonen, voor je ze gaat aanpassen of verwijderen. Daarom besloot ik om als eerste het opvragen van de gegevens over een melding te realiseren.
Daarna stond het wijzigen, invoeren en verwijderen op de planning. Het overzicht van alle meldingen bewaarde ik tot het laatst.
Voor de eerste pilot had ik zes weken beschikbaar. Om dan een essentieel onderdeel als het beginscherm tot het einde te bewaren lijkt misschien een vreemde keuze. Ik verwachtte echter dat het overzicht met alle meldingen makkelijker te maken was als ik ervaring had opgedaan met ASP.NET. Daarnaast was mijn planning voor de eerste pilot ruim opgezet. Het was dus onwaarschijnlijk dat ik in tijdnood zou komen.

Het pilotontwikkelplan is te vinden in externe bijlage 5.

5.4 Ontwerp software-bouweenheden

Het ontwerp van de software-bouweenheden is tegelijkertijd gebeurd met het bestuderen van de documentatie over ASP.NET. Nieuwe inzichten tijdens het bouwen van de software-bouweenheden zijn steeds teruggekoppeld, zodat het systeemontwerp up-to-date bleef.

Voor de eerste pilot ging ik de belangrijkste functionaliteit met betrekking tot de workflow implementeren. Alle klassen, behalve de klassen die verantwoordelijk waren voor de gebruikers en rollen, waren van belang.



Alle klassen uit het datamodel, behalve de klassen die verantwoordelijk zijn voor de gebruikers en rollen

Aan de hand van dit model ben ik de klassen, die in mijn applicatie gebruikt zouden worden, gaan modelleren. De belangrijkste eis bij het vinden van een klasse, is dat een klasse verantwoordelijkheid moet hebben.
In de applicatie staan meldingen centraal. Daarom ben ik bij het ontwerp van de software-bouweenheden als eerste de klasse Melding gaan modelleren. Deze klasse is verantwoordelijk voor een melding in de applicatie. Van een melding worden het ID en de datum van aanmelding opgeslagen. Deze attributen moet de klasse dus bezitten.
De operaties bij deze klassen zijn als eerste het opvragen van de gegevens. Zo is er GetDatumAanmelding() nodig om de datum van aanmelding op te kunnen halen. Daarnaast moet het mogelijk zijn om het status object dat bij een melding hoort te kunnen opvragen, hiervoor is de operatie GetStatus() gemaakt.

De volgende klassen werden in de eerste pilot gebruikt:

- Melding
- Klant
- Soort
- Workflow
- Status
- Keuze
- GemaakteKeuze

De klasse Veld en Veldwaarden zijn DataSets geworden. Wat dat zijn, bespreek ik in paragraaf 5.4.3.

5.4.1 Sequentiediagrammen opstellen

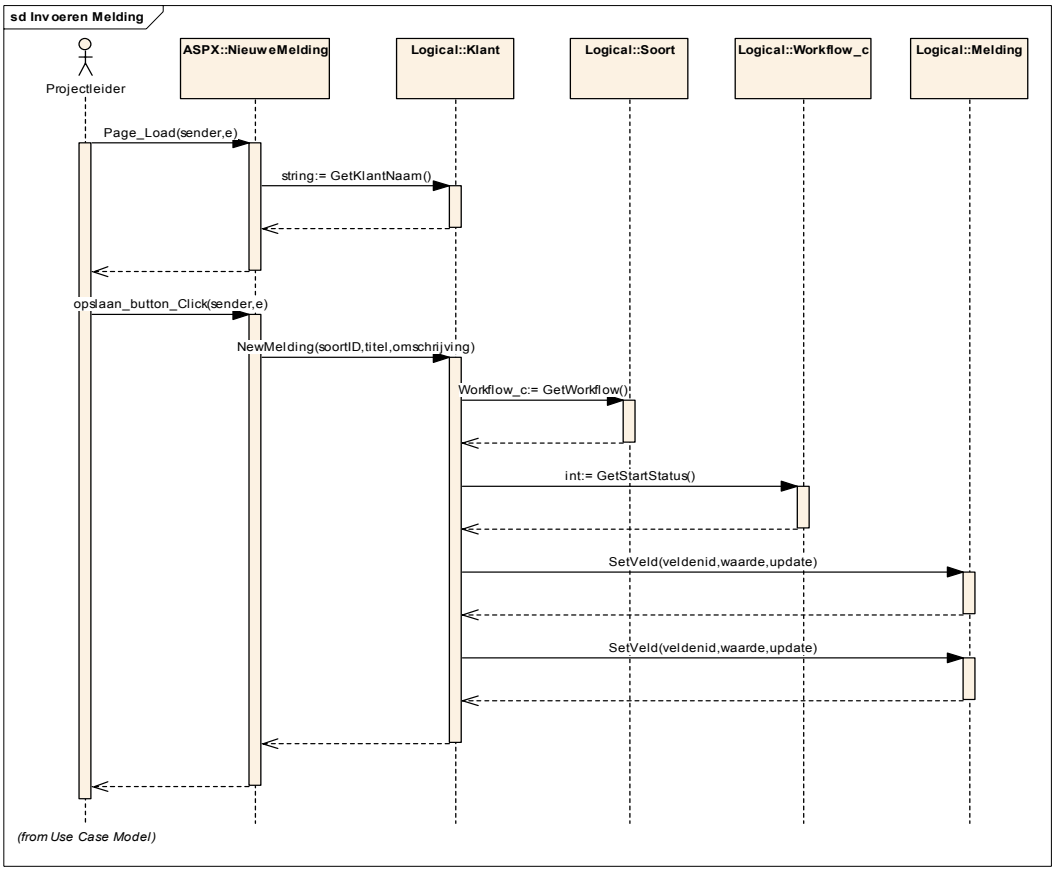
Nadat ik de klassen met bijbehorende attributen globaal duidelijk had, ben ik sequentiediagrammen gaan modelleren. Elk pilotdeel (use-case) heeft zijn eigen sequentiediagram.
Het doel van het modelleren van sequentiediagrammen is duidelijkheid krijgen over de afhandeling van een use-case. Dit kan zeer nuttig zijn bij de latere implementatie, omdat de stappen om tot het einddoel te komen al duidelijk weergegeven zijn.

Ik heb bijvoorbeeld het sequentiediagram “Invoeren melding” gemaakt, welke te vinden is op de volgende bladzijde. Dit sequentiediagram heb ik opgesteld door eerst in een stappenplan te beschrijven wat de benodigde acties zijn voor het invoeren van een melding. De stappen die nodig zijn, zal ik hieronder uitleggen.

Nadat de gebruiker de gegevens over een melding heeft ingevoerd, worden deze verstuurd. De operatie NewMelding() wordt dan aangeroepen.
Deze operatie wil een nieuwe melding in de database schrijven, maar heeft dan wel de volgende gegevens nodig:

- ID
- Datum van aanmelding
- Soort
- Status
- Klant

Het ID wordt automatisch aangemaakt door de database. De datum is op het systeem bekend en de klant via de Sharepoint portal. Dan blijven enkel de soort en de status over.
Elke klant heeft een default soort, dit is gemaakt zodat een melding standaard als foutmelding ingevoerd kan worden, dit was een eis van de opdrachtgever. Doordat de klant bekend is, is dus ook het soort bekend.
Vervolgens is het soort gekoppeld met een workflow. Een workflow heeft een start status. Deze start status zal de status worden van de nieuwe melding.
Dan zijn alle gegevens aanwezig om een nieuwe melding aan te maken. Tot slot moeten de titel en de omschrijving die zijn ingevoerd op de pagina bij het juiste veld in de VeldWaarde tabel worden opgeslagen.



Sequentiediagram voor het invoeren van een melding.

Alle sequentiediagrammen zijn te vinden in externe bijlage 6.

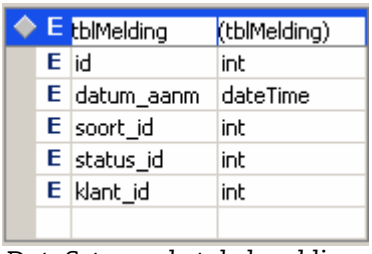
5.4.2 ETX Database Component

Voor ik begon aan deze opdracht had ik nog geen kennis van ASP.NET. Ik stond volledig open voor de ideeën over hoe ik de toegang tot de database het beste kon aanpakken.
Als eerste heb ik geëxperimenteerd met het handmatig opbouwen en sluiten van de database connectie in de applicatiecode. De code, om de connectie op te bouwen en af te breken, kon in één basisklasse staan. De verschillende queries op de tabellen konden verdeeld worden over subklassen, die afleiden van de basisklasse, zodat niet elke subklasse zelf een connectie hoeft aan te maken.

Voor deze oplossing heb ik echter niet gekozen, omdat een vergelijkbaar, maar beter alternatief beschikbaar was. Bij de opdrachtgever was al veel ervaring met databasetoegang vanuit ASP.NET. Daarom hadden zij hun eigen component ontwikkeld wat dit vereenvoudigde.
Het component handelde de connectie af met de database. Verder werkte het samen met DataAdapters en DataSets. Elke tabel in de database krijgt in principe zijn eigen DataAdapter en zijn eigen DataSet. De DataAdapter heeft als functie de query uit te voeren op de database. Elke DataAdapter erft van het database component, zodat het niet zelf de connectie met de database hoeft te implementeren.
Een DataSet is een schema met de definitie van de data. De data uit de database kan via de Data Adapter in een DataSet geladen worden.
Het gebruik van het ETX Database Component heeft tevens als voordeel dat het database onafhankelijk is en de applicatie dus makkelijk overgezet kan worden naar een andere database.

5.4.3 DataSets

Een DataSet is een vooraf gedefinieerd schema waar tabellen, die voldoen aan dat schema, in opgeslagen kunnen worden. De DataSet wordt opgeslagen in XML. Zo'n tabel in een DataSet kan gekoppeld worden aan een element (bijvoorbeeld een tekstvak of drop down box) in een ASP.NET pagina.
Ook is het mogelijk om bepaalde rijen en kolommen uit te lezen of de data in een DataSet in een andere volgorde te presenteren door middel van een DataView.



DataSet van de tabel melding.

Het voordeel van het gebruik van een DataSet is dat de opbouw van de DataSet bekend is. De data die je inleest moet voldoen aan je DataSet schema, net als de applicatiecode die de DataSet uitleest. Je hebt dus vooraf een soort validatie. Als ik in mijn applicatiecode een kolom probeer uit te lezen die niet bestaat in de DataSet, word ik hier vooraf al op gewezen door mijn ontwikkelomgeving.
Het is ook mogelijk om DataSets vanuit een query op te bouwen, dan vervalt het voordeel van vooraf valideren. Welke kolommen in je DataSet komen, hangt af van welke kolommen je opvraagt in je sql-query. Je hebt dan een zogenaamde untyped DataSet.

Een nadeel van het gebruik van DataSets is, dat het tijd kost deze aan te maken en te onderhouden. Als je namelijk aanpassingen maakt aan een tabel in je datamodel, zal je de DataSet ook moeten updaten, omdat deze anders niet meer overeenkomen.

Ik heb er voor gekozen om vooraf DataSets te definiëren. DataSets zijn een veel gebruikt onderdeel in ASP.NET waar je niet omheen kan. Ze zullen ofwel typed of untyped gebruikt moeten worden. De typed variant biedt voordelen, het zal mij zeker helpen om gestructureerd te werk te gaan, de applicatiecode zal er ook overzichtelijker door worden.
Door duidelijke naamgeving van de DataSets, zal altijd bekend zijn wat er in een bepaalde DataSet is opgeslagen. Dit voorkomt onduidelijkheid, zoals je die zou kunnen krijgen bij untyped DataSets.

Een DataSet die ik bijvoorbeeld heb aangemaakt is de MeldingDataSet. Deze kan gevuld worden door de MeldingDataAdapter. De MeldingDataSet is aangemaakt door de definitie van de Melding tabel, uit het datamodel, over te nemen in de MeldingDataSet.

5.4.4 Stored Procedures

Stored procedures zijn queries op de database die vooraf gedefinieerd en opgeslagen zijn op de database server. Bij de opdrachtgever worden deze altijd aangemaakt voor applicaties die gegevens uit een database gebruiken.

Het voordeel van het gebruik van stored procedures is:

- Portable, makkelijk te kopiëren of over te zetten.
- Scheiding code en query, geen queries in je applicatie code
- Veiligheid

De opdrachtgever verplichtte mij niet om stored procedures te gebruiken, ik stond zelf voor de keuze of ik dit zou doen. Ik had er voor kunnen kiezen geen stored procedures te gebruiken. De queries op de database zouden dan in de DataAdapters komen te staan.
Ik heb er voor gekozen om wel met stored procedures te werken. Dit levert een goede scheiding op van queries en code. Hierdoor is een query makkelijk aan te passen, zonder de applicatiecode opnieuw te hoeven compileren.

Het gevolg was, dat ik voor elke tabel stored procedures aan moest maken. Dit heb ik echter niet in één keer gedaan, maar elke keer op het moment dat ik de stored procedure nodig had.

Door de samenwerking tussen het ETX Database Component, de DataAdapter en de DataSet worden automatisch aangemaakte ID's in de database meteen teruggezet op de juiste plek in de DataSet. Hierdoor is het mogelijk om het unieke ID dat een melding na aanmaken krijgt, uit te lezen uit de DataSet.

5.4.5 Drie lagen model

Door de scheiding van DataSets, DataAdapters, Klassen en de User Interface heb ik een drie lagen model gecreëerd. Dit drie lagen model wordt gebruikt door Microsoft en ook de opdrachtgever maakt gebruik van dit model.

De drie lagen zijn:

- User Interface
  Deze laag bevat de userinterface in aspx bestanden.

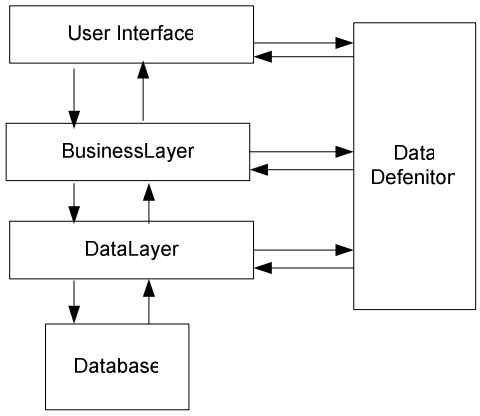
- BusinessLayer
  Hier komen de klassen met de applicatie logica.

- DataLayer
  De DataLayer laag zorgt voor de communicatie met de database. Hierin zitten de DataAdapters en Stored procedures.

Daarnaast is er nog een “ondersteunende” laag:

- DataDefinition
  In de DataDefinition zitten de DataSets.

Tijdens het verdere verloop van het project ben ik uitgegaan van deze lagen en heb ik mijn ontwikkelomgeving hier op ingericht, door mijn hoofdproject op te delen in deelprojecten.



Visuele weergave van het drie lagen model

5.5 Bouw software-bouweenheden

Na het ontwerp van de pilotdelen en bouwstenen ben ik begonnen met de bouw in de volgorde zoals die was afgesproken in het pilotontwikkelplan. Ik begin met het uitleggen van de grafische opbouwen van de schermen. Vervolgens bespreek ik de bouw van enkele pilotdelen.

5.5.1 Schermbouw

De opbouw van de schermen is volledig gebeurd in VisualStudio.Net. Dit was mogelijk, doordat ik had gekozen om het Smartpart te gebruiken. Via VisualStudio.Net is het mogelijk om elementen op een pagina te slepen en te positioneren. Ook de naam en de eigenschappen van een element kunnen aangepast worden.

Voor elke pagina die ik ging ontwikkelen, ben ik begonnen met de gewenste elementen op de pagina te zetten. Soms voegde ik nog enkele extra elementen toe, om de tussenstappen tijdens het ontwikkelen daarin zichtbaar te maken. Bij het ontwikkelen van het overzicht van alle meldingen bijvoorbeeld, heb ik elke tussenstap zichtbaar gemaakt op het scherm en na afloop weer verwijderd. Meer hierover is te vinden in paragraaf 5.5.6.

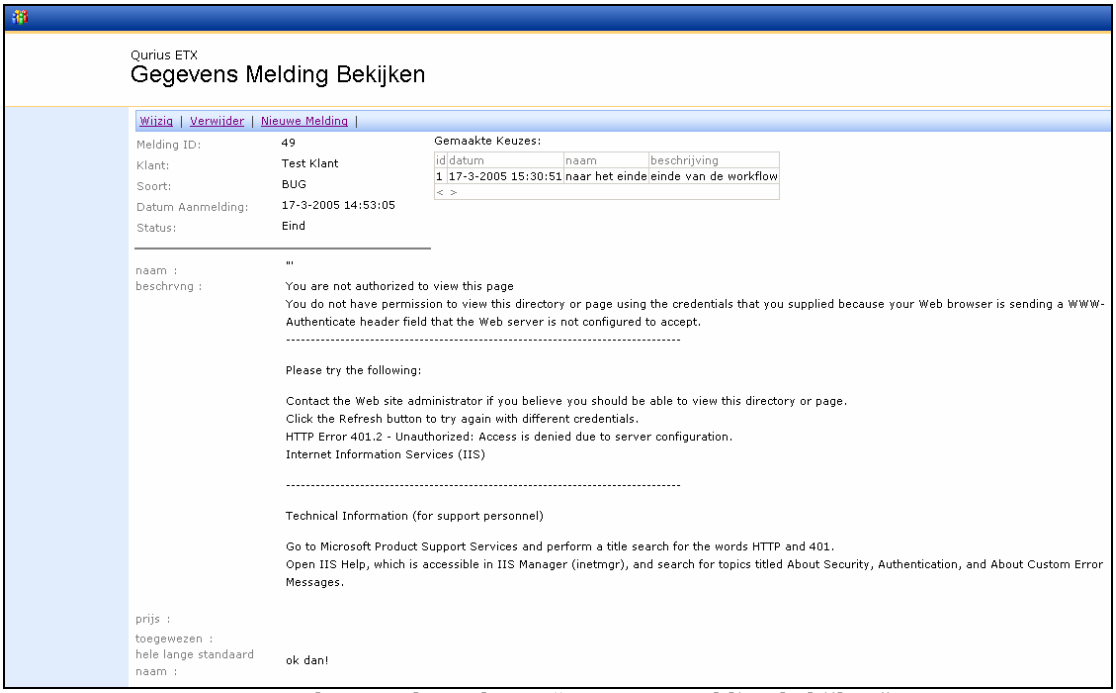
De opdrachtgever wilde dat de pagina's er uitzagen als Sharepoint pagina's. Om dit te realiseren, heb ik in een standaard Sharepoint pagina alle tekst verwijderd. Ik hield uiteindelijk een lege pagina over, met enkel het uiterlijk van een Sharepoint pagina.
Om de pagina's het uiterlijk van Sharepoint te geven, kopieerde ik de inhoud van mijn pagina naar de lege Sharepoint pagina. Om vervolgens de verschillende elementen op mijn pagina ook nog het Sharepoint uiterlijk te geven, moest ik aangeven welke stijl elk element moest hebben.
De verschillende stijlen, zoals bijvoorbeeld een klein lichtgrijs lettertype, staan gedefinieerd in een CSS bestand. Dit is bedoeld voor het opslaan van stijlen. Het CSS bestand bestond al en werd gebruikt door Sharepoint. De stijl van de applicatie zal automatisch mee veranderen als de stijl in het CSS bestand wordt aangepast.

5.5.2 Gegevens melding bekijken

Bij het opstellen van het pilotontwikkelplan heb ik een keuze gemaakt voor de volgorde van ontwikkeling van de use-cases. De use-case die verantwoordelijk is voor het opvragen van de gegevens over een melding, stond op de planning om als eerste ontwikkeld te worden.

Een probleem waar ik tegenaan liep bij deze use-case was het presenteren van de velden (zoals naam en omschrijving) met hun bijbehorende waarde. Doordat velden toegevoegd en verwijderd konden worden, kon er niet van tevoren vastgelegd worden welke velden op de pagina gepresenteerd moesten worden.
Door middel van een DataAdapter, DataSet en stored procedure kon ik de gegevens uit de database halen. Dit leverde als resultaat een DataSet met daarin de veldnamen en veldwaarden voor een specifieke melding. Deze moest ik nu nog presenteren op de pagina.
ASP.NET bood hier twee oplossingen voor, een DataGrid en een DataList. Een DataGrid kan de gegevens uit een DataSet presenteren in een tabel. Een DataList kan alle rijen in een DataSet doorlopen en voor elke rij de waarde in bijvoorbeeld een textbox zetten, zodat de waarde is aan te passen. Maar het is ook mogelijk de waarde gewoon als tekst te presenteren.

Ik heb gekozen om de DataList te gebruiken en niet de DataGrid. Dit heb ik gedaan, omdat de DataList mij meer vrijheid gaf met betrekking tot het presenteren van de gegevens. Zo is het gemakkelijk om de code te hergebruiken en aan te passen zodat de gegevens in een wijzigbare textbox komen te staan.



Screenshot van het scherm "gegevens melding bekijken"

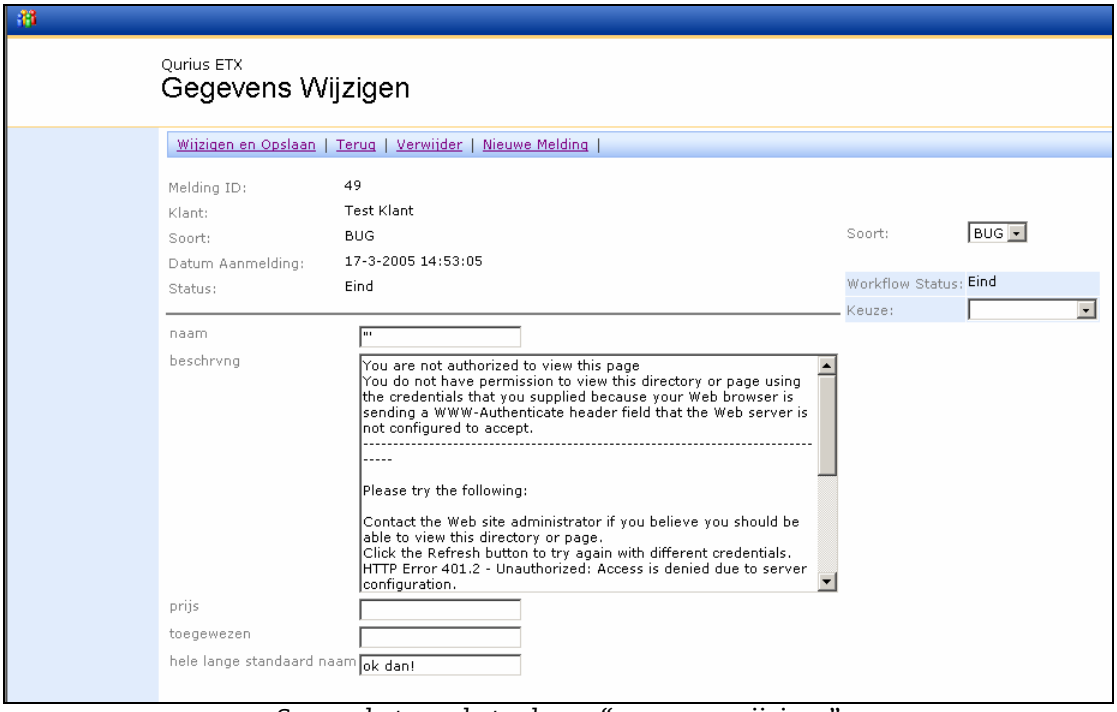
5.5.3 Gegevens melding wijzigen

Op de pagina waar de gegevens gewijzigd konden worden, kwamen vrijwel dezelfde gegevens als op de pagina waar de gegevens over een melding opgevraagd konden worden. Het enige verschil was dat nu een deel van de gegevens wijzigbaar was.

Er is ook een gedeelte dat nooit wijzigbaar is, dit zijn de algemene gegevens over een melding, zoals het unieke nummer en de datum van aanmelding. Dit gedeelte kon direct gekopieerd worden van de pagina die de gegevens over een melding opvraagt. Ik verwachtte echter dat code hergebruik netter opgelost kon worden in ASP.NET. Daarom ben ik gaan zoeken hoe dit het beste gedaan kon worden. Via het boek "Developing web applications" kwam ik er achter dat een Web User Control, zoals ik die ook ging maken voor mijn webpart, hergebruikt kon worden op meerdere pagina's.
Deze oplossing was een stuk netter. Wanneer er nu een verandering plaatsvindt bij de algemene gegevens, hoeft dit slechts aangepast te worden in één bestand. In de toekomst is het dus makkelijk om de gebruikersnaam, van de gebruiker die de melding heeft aangemeld, toe te voegen. Deze gebruikersnaam zal dan op alle pagina's die het Web User Control importeren zichtbaar zijn.

Een probleem waar ik tegenaan liep bij het wijzigscherm, was hoe ik de gegevens kon presenteren om te wijzigen. Sommige velden bevatten namelijk een integer en anderen een lange tekst, zo'n lange tekst kan niet in een klein vakje komen te staan. In mijn datamodel heb ik hier al rekening mee gehouden door het datatype van het veld bij te houden.

In mijn applicatiecode heb ik dit probleem opgelost door standaard elke waarde in een textbox te plaatsen. Vervolgens, als de gehele lijst is opgebouwd, met alle velden en hun waarde in een textbox, doorloop ik deze en vraag het datatype van het veld op. Als het datatype voldoet aan de voorwaarde “longtext” zal de textbox worden veranderd in een grotere textbox, zodat er een groter veld beschikbaar is voor het wijzigen van de gegevens. Dit is in het screenshot hieronder te zien.



Screenshot van het scherm “gegevens wijzigen”

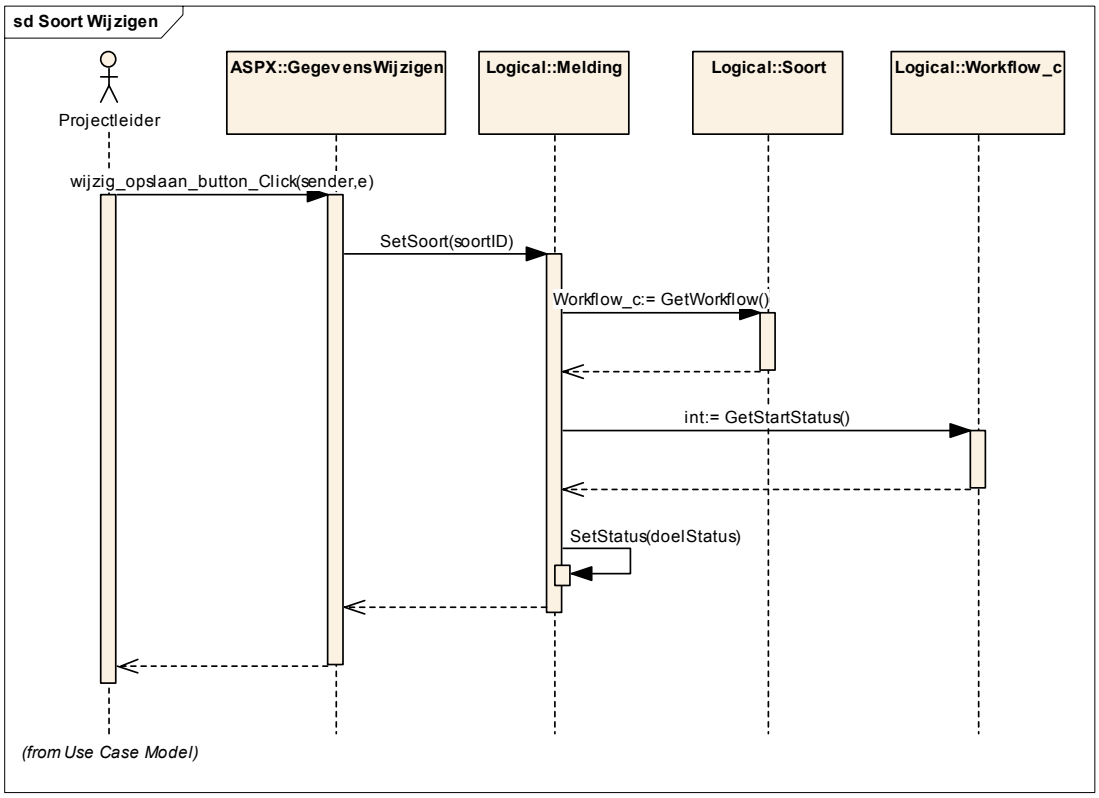
5.5.4 Workflow

Naast het wijzigen van de gegevens is het in dit scherm ook mogelijk om een stap in de workflow te nemen. In de drop down box aan de rechterkant van het scherm zijn de verschillende mogelijke keuzes te kiezen. Aan de hand van de gekozen keuze zal de status van de melding veranderen.

5.5.5 Soort Wijzigen

De use-case “soort wijzigen” is onderdeel van de use-case “Gegevens melding wijzigen”. Ze vinden plaats in hetzelfde scherm en de actie zit onder één knop. Op het scherm heb ik een drop down box gemaakt, met daarin de huidige soort en de mogelijke soorten.

Het veranderen van het soort heeft echter meer gevolgen dan alleen de soort wissel. Bij een soort kan namelijk een andere workflow horen. De oplossing voor dit probleem heb ik eerst beschreven in een sequentiediagram, dit is hieronder weergegeven. Daarin heb ik gemodelleerd welke objecten met elkaar communiceren. Wanneer het soort wordt veranderd in “wijzigingsverzoek” wordt eerst de workflow van een wijzigingsverzoek opgehaald. Bij deze workflow hoort een start status, namelijk “RFC Open”. Deze status wordt opgevraagd en zal de nieuwe status van de melding worden.
Doordat ik de communicatie tussen de klassen van tevoren had gemodelleerd, was tijdens de bouw meteen duidelijk hoe dit moest worden geïmplementeerd.

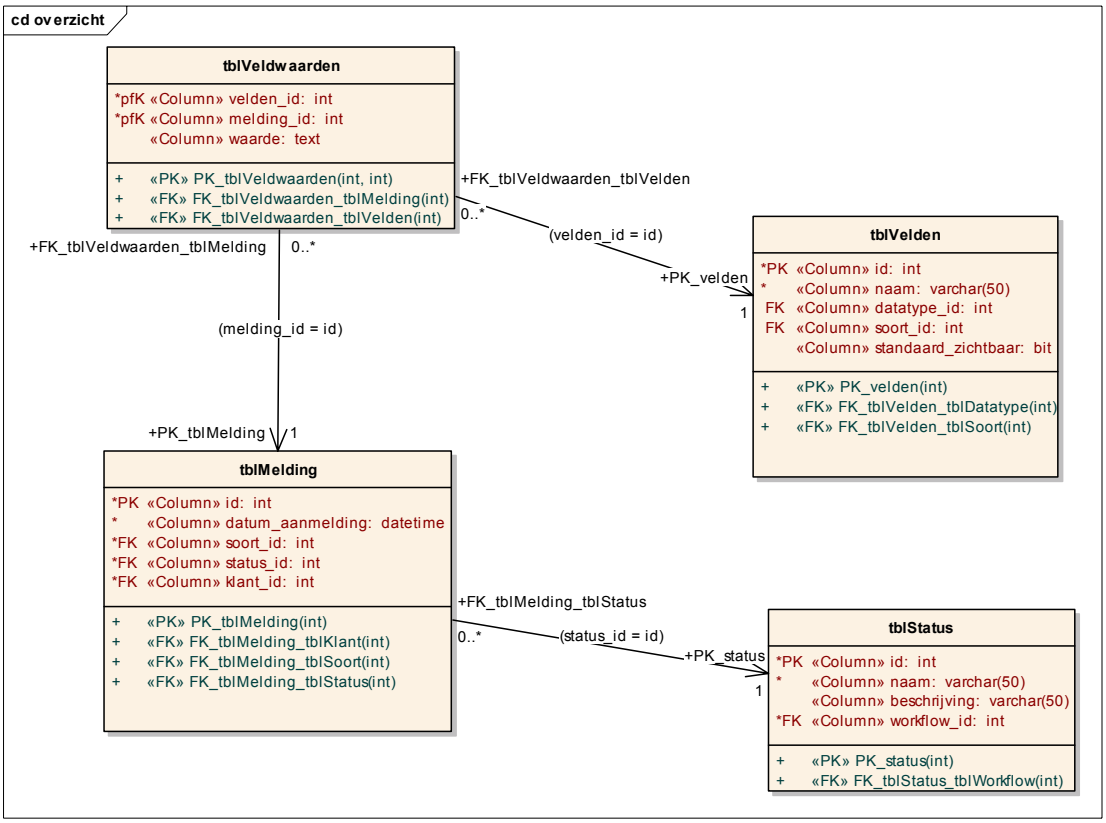


Sequentiediagram “Soort Wijzigen”

5.5.6 Overzicht meldingen opvragen

Het overzicht van meldingen opvragen was een van de meest complexe onderdelen van deze pilot. Dit had ik bewaard tot het einde van de pilot, zodat met behulp van mijn kennis die ik had opgedaan bij de andere onderdelen, dit onderdeel sneller zou gaan.
Bij de totstandkoming van het datamodel is er voor gekozen om de waarde van een veld van een melding, zoals de naam en omschrijving, niet bij de Melding zelf op te slaan, maar in de klasse VeldWaarde. Wat voor een waarde er wordt gepresenteerd, wordt bijgehouden bij de klasse Veld.

Dit datamodel leidde tot de volgende tabellen waaruit het overzicht met alle meldingen opgebouwd werd:

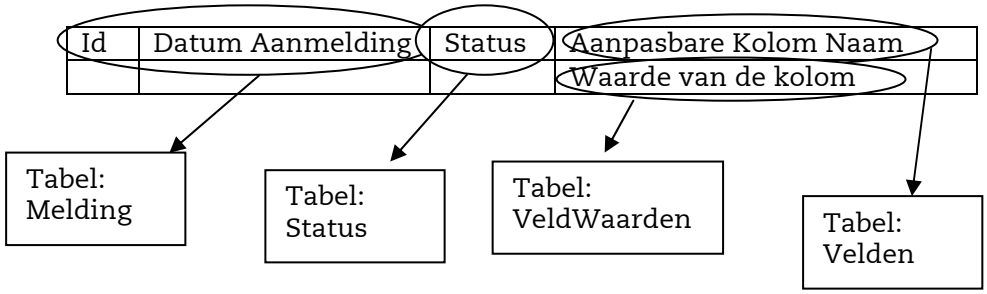


De vier tabellen die nodig zijn voor het opbouwen van het overzicht met meldingen.

Dit onderdeel was complex, omdat er rijen uit de tabel Velden moeten worden gepresenteerd als kolomnamen. De waarde uit zo'n kolom komt weer uit de tabel Veldwaarden.

Het doel was om een overzicht te krijgen met daarin voor elke melding de standaard gegevens, zoals het nummer en de datum van aanmelding. Daarnaast de naam van de status en daarnaast de waarden van de velden.

Om tot het uiteindelijke overzicht te komen, heb ik deze eerst uitgetekend en aangegeven welke gegevens waar vandaan komen. Dit is hieronder te zien.



De naam van de aanpasbare kolomnaam komt uit de tabel Velden. Alle velden in die tabel worden getoond als kolom naam ,mits ze niet gekoppeld zijn aan een specifiek veld en de waarde “standaard zichtbaar” op true (1) staat.

De gegevens die nodig zijn voor het opbouwen van dit overzicht worden opgehaald uit de database en opgeslagen in tijdelijke DataSets. Om zeker te weten dat de juiste data in de DataSets stond, heb ik ook de tijdelijke DataSets tijdens het ontwikkelen zichtbaar gemaakt op de pagina. Dit hielp goed bij het maken van elke tussenstap, omdat ik kon zien met welke data ik werkte en of de tussenstappen succesvol verliepen.

Ik zal nu kort de stappen bespreken die nodig waren voor het opbouwen van het overzicht. Dit wordt gedaan met voorbeeld gegevens.
De tabel Melding heeft een vreemde sleutel naar de tabel Status. Hierdoor is het wel mogelijk om met één query zowel de gegevens uit de tabel Melding en Status op te halen. Dit was de eerste stap voor het maken van het overzicht, en resulteerde in de volgende tabel:

OverzichtDataSet:

Melding_id	Datum_aanmelding	StatusNaam
1	01-01-2005	Aangemeld
2	01-01-2005	In behandeling

De volgende stap was het ophalen van alle velden uit de Velden tabel waarvan het soort_id null is, dit zijn de velden die alle meldingen gemeen hebben en dus getoond kunnen worden in het overzicht. Een andere voorwaarde was, dat de waarde van standaard_zichtbaar in de tabel Velden op true (1) moest staan, dit betekende dat het veld ook getoond mocht worden in het overzicht.

Deze stap had als resultaat een tabel met daarin alle Velden die getoond moesten worden in het overzicht als kolom.

VeldenDataSet:

ID	Naam	soort	Standaard_zichtbaar
1	Naam	null	1
2	Omschrijving	null	1

Om deze velden als kolom te tonen werd de tabel doorlopen en voor elk veld werd er een nieuwe kolom aangemaakt bij de uiteindelijke tabel.

Na het éénmaal doorlopen van deze loop, was de onderstaande tabel het resultaat.

OverzichtDataSet:

Melding_id	Datum_aanmelding	StatusNaam	Naam
1	01-01-2005	Aangemeld	
2	01-01-2005	In behandeling	

Tijdens deze loop werden alle waarden opgehaald uit de tabel Veldwaarden die specifiek hoorden bij het huidige veld in de loop. Deze waarden horen bij een melding en moeten in de specifieke kolom komen te staan. Dit had onderstaande tabel als resultaat.

VeldenVeldwaardenDataSet:

Velden_id	Melding_id	waarde
1	1	Testnaam
1	2	Melding 2

Vervolgens werd in een andere loop de bovenstaande tabel doorlopen. Elke waarde hoort bij een melding, deze melding werd opgezocht in de uiteindelijke tabel en de waarde kwam onder de nieuw aangemaakte kolom, op de juiste rij.
Dit had het volgende resultaat:

OverzichtDataSet:

Melding_id	Datum_aanmelding	StatusNaam	Naam
1	01-01-2005	Aangemeld	Testnaam
2	01-01-2005	In behandeling	Melding 2

Het resultaat was, dat de kolom met de juiste waarden, was toegevoegd. Dit werd net zo vaak herhaald als er kolommen toegevoegd moesten worden. Het uiteindelijke resultaat in dit voorbeeld zou dus na twee loops zijn:

OverzichtDataSet:

Melding_id	Datum_aanmelding	StatusNaam	Naam	Omschrijving
1	01-01-2005	Aangemeld	Testnaam	Test 1
2	01-01-2005	In behandeling	Melding 2	Test 2

5.5.7 Webpart bouwen

Het scherm waar het overzicht van meldingen werd getoond, moest binnen de Sharepoint portal als webpart komen te draaien. Om dit te kunnen doen, is dit scherm ontwikkeld als Web User Control en niet als normale pagina. Hierdoor kon het smartpart het web user control inladen en tonen als webpart binnen de Sharepoint portal.

De opdrachtgever wilde graag gebruik maken van een specifieke webpart mogelijkheid, namelijk “custom properties”. Dit houdt in dat het er bij het instellen van de webpart enkele variabelen meegegeven worden. Op die manier is het mogelijk dezelfde webpart te gebruiken op meerdere portals. Door het veranderen van bijvoorbeeld de variabele klant, zullen er in de webpart specifieke meldingen voor die klant worden getoond.

De opdrachtgever wilde de mogelijkheid om de webpart te installeren per klant, maar ook per soort melding. Ik heb daarom deze twee variabelen beschikbaar gemaakt als custom property in Sharepoint.

5.6 Beheer en Installatie handleiding

Omdat de webpart zelf geen beheermogelijkheden had, heb ik een handleiding geschreven gericht op het onderhoud van de webpart. In deze handleiding staat omschreven hoe nieuwe klanten aangemaakt kunnen worden en hoe velden aan een soort gekoppeld kunnen worden.
Als onderdeel van deze handleiding heb ik een stored procedure gemaakt die de workflow, zoals te vinden in externe bijlage 12, automatisch aanmaakt. Dit maakt het voor de opdrachtgever makkelijk om een kleine aanpassing te doen en een gehele nieuwe workflow opnieuw aan te maken.

De volledige beheer en installatie handleiding is te vinden in externe bijlage 9 en 10.

5.7 Beoordelen en Testen

Tijdens het ontwikkelen, is gecontroleerd of de verschillende klassen werkten door middel van unit testing. Dit is een whitebox test techniek. Whitebox testen houdt in dat de interne architectuur bekend is. Ook is er getest op grenswaarden, zoals bijvoorbeeld het verschijnen van een “>” als er meer dan tien meldingen in de overzichtlijst staan, zodat de volgende tien meldingen bekeken kunnen worden.
Defect testing is het testen van de applicatie, gericht op het veroorzaken van fouten. Dit heb ik gebruikt om fouten te vinden in de applicatie. Door middel van het debuggen van de code kon ik de fouten lokaliseren en wist ik precies op welke regel code het mis ging. Vervolgens kon ik het probleem verhelpen.

Vanuit het bedrijf zijn er standaarden en richtlijnen om software te testen. Aan de hand hiervan heb ik een testplan gemaakt om de acceptatie test te kunnen uitvoeren. Dit testplan is te vinden in externe bijlage 11.
Het doel van het testplan was, om te testen of de applicatie voldeed aan de eisen die de opdrachtgever had gesteld en of de functionaliteit overeenkwam met de omschrijving in de use-cases.
In het testplan heb ik stappen beschreven, die door een tester of de opdrachtgever uitgevoerd konden worden. Na elke stap stond het verwachte resultaat van de actie.
Het testplan is opgesteld aan de hand van de use-cases uit het systeemconcept. Elk use-case scenario had zijn eigen deel-testplan.

In het testplan werd ook rekening gehouden met het testen van de invoer van lege velden of vreemde tekens, omdat deze foutgevoelig zijn bij het invoeren naar een database.

De acceptatie test is uitgevoerd door Michiel Bruins. Hij heeft het systeem doorlopen volgens het testplan. Voor de test was de werkelijke workflow die gevolgd diende te worden ingevoerd, zodat deze ook getest kon worden.
Uit de test bleken geen fouten, alleen enkele opmerkingen en verbeterpunten. Zo wilde de opdrachtgever bijvoorbeeld “terug” buttons, zodat de applicatie makkelijker te doorlopen was.
Ook vroeg hij om meer mogelijkheden om de kolommen, die zichtbaar waren in het overzicht in de webpart, aan te passen. Deze nieuwe eisen en wensen werden meegenomen naar de volgende definitie studie.
Enkele tekstuele wijzigingen werden meteen doorgevoerd.

6 Definitie studie 2

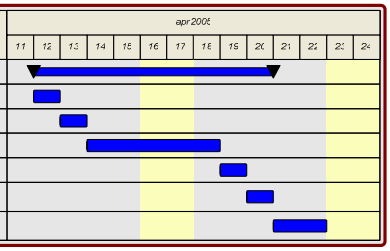
De eerste pilot was voorspoedig verlopen. Ik was ongeveer een week eerder klaar dan gepland. Desondanks had ik besloten om geen derde pilot uit te voeren, omdat er niet zoveel tijd over was om nog een gehele iteratie te doorlopen. Ik heb besloten om de tweede pilot uit te breiden.
Dit besluit heb ik aan het begin van de tweede definitie genomen en duidelijk gecommuniceerd naar de opdrachtgever. Hij kon er rekening mee houden, door aan te geven welke eisen hij nog graag gerealiseerd zou zien worden in de tweede pilot.

In de tweede pilot ging de rechtenstructuur geïmplementeerd worden. Ook zouden er enkele wijzigingen plaatsvinden die naar voren kwamen naar aanleiding van de eerste pilot. Hoe ik ben gekomen tot de systeemeisen en de ontwikkeling van het systeemconcept, wordt in dit hoofdstuk besproken.

6.1 Systeemeisen vaststellen

Om in de tweede pilot meer functionaliteit te kunnen realiseren, zou ik tijdens de definitie studie ook meer tijd nodig hebben om meer functionaliteit te beschrijven in het systeemconcept. Ik heb hier in mijn planning rekening mee gehouden, door enkele dagen extra tijd vrij te maken voor de tweede definitie studie. De tijd die nog niet ingevuld was in mijn planning, heb ik gereserveerd voor pilot twee.

ID	Task Name	Start	Finish	Duration
1	**Definitiestudie**	**12-4-2005**	**20-4-2005**	**7d**
2	Voorbereiden pilotplan-workshop	12-4-2005	12-4-2005	1d
3	Definieren van systeemeiser	13-4-2005	13-4-2005	1d
4	Opstellen systeemconcept	14-4-2005	18-4-2005	3d
5	Pilotplan Opsteller	19-4-2005	19-4-2005	1d
6	Testplan opsteller	20-4-2005	20-4-2005	1d
7	Eindverslag schrijver	21-4-2005	22-4-2005	2d



Planning definitie studie 2

De acceptatie test, die was uitgevoerd door Michiel Bruins, had als resultaat enkele opmerkingen en nieuwe systeemeisen en wensen. De opmerkingen, eisen en wensen heb ik verwerkt in het document met de systeemeisen. In dit document heb ik op een aparte pagina beschreven wat de eisen en wensen naar aanleiding van de eerste pilot waren, zodat voor de opdrachtgever duidelijk was dat deze verwerkt waren.
De opdrachtgever wilde bijvoorbeeld graag dat er bij het wijzigen van de gegevens een controle was op de invoer, sommige velden mochten niet leeg zijn en in sommige velden mocht enkel een getal ingevoerd worden. Dit heb ik verwerkt in de systeemeisen en later in het systeemconcept in de use-case “gegevens melding wijzigen”.

De systeemeisen zijn vervolgens besproken met Michiel Bruins. Tijdens het gesprek gaf hij al aan welke eisen hij erg graag in de tweede pilot gerealiseerd zou willen zien worden. Aan de hand van dit gesprek heb ik de prioriteit van de eisen opgesteld.

De systeemeisen zijn te vinden in externe bijlage 1.

6.2 Systeemconcept

Nadat de systeemeisen besproken en goedgekeurd waren door de opdrachtgever, ben ik deze gaan verwerken in het systeemconcept. Het systeemconcept is opgedeeld in de functionele eisen (use-cases) en de niet functionele eisen.
Aan de hand van de nieuwe systeemeisen zijn de bestaande use-cases aangepast. Er zijn geen nieuwe use-cases bijgekomen, dit kwam doordat ik de use-cases die ik in de tweede pilot ging realiseren al had beschreven tijdens de eerste definitie studie.

6.2.1 Overzicht

De grootste verandering vond plaats bij de use-case “Overzicht alle meldingen opvragen”. De opdrachtgever wilde namelijk graag verschillende mogelijkheden om het overzicht van alle meldingen te presenteren. Bij elk overzicht moesten andere kolommen zichtbaar zijn. In de eerste pilot werden enkel de kolommen getoond die niet bij een soort hoorden. De opdrachtgever wilde dat het ook mogelijk werd dat er kolommen getoond werden die bij een specifiek soort hoorden.

De kolommen worden opgebouwd uit de velden die bij een soort horen. Een melding is vervolgens ook van een bepaald soort. Een melding van het soort “wijzigingsverzoek” heeft o.a. de velden prioriteit en prijs. Dit kunnen kolommen worden in het overzicht van alle meldingen. Van elk veld kan worden aangegeven of deze getoond mag worden in het overzicht door de variabele “standaard zichtbaar”.

De opdrachtgever wilde de volgende overzichten:

- Een overzicht van alle meldingen (van verschillende soorten) van een klant.
  De kolommen die getoond moeten worden:
  - Alle velden waarvan aangegeven is dat ze standaard zichtbaar zijn en horen bij een soort dat bij deze klant hoort.

- Een overzicht van alle meldingen (van één soort) van een klant.
  De kolommen die getoond moeten worden:
  - Alle velden waarvan aangegeven is dat ze standaard zichtbaar zijn en horen bij het soort dat wordt weergegeven in het overzicht.

- Een overzicht van alle meldingen (van één of meerdere soorten) van een klant.
  De kolommen die getoond moeten worden:
  - Enkel de basis velden (ID, datum van aanmelding, naam, omschrijving) van iedere melding.

Voor elk overzicht geldt dat enkel de velden waar de gebruiker recht op heeft om die te mogen zien, getoond mogen worden.

Dit waren de eisen vanuit de opdrachtgever. Ik concludeerde hieruit dat de eerste twee overzichten eigenlijk hetzelfde waren. Bij het eerste overzicht worden alle kolommen getoond die bij de soorten horen in het overzicht en bij het tweede overzicht ook. Bij het tweede overzicht is er echter maar één soort in het overzicht.

De verschillende overzichten heb ik ook beschreven in het systeemconcept. Hier heb ik alle drie de overzichten beschreven en niet slechts twee. Door ze alle drie te beschrijven in het systeemconcept, is het voor de opdrachtgever duidelijker wat er gebeurt als er meerdere soorten in een overzicht zitten of slechts één soort.

6.2.2 Zoeken

De use-case “Zoeken” was al uitgewerkt in de eerste versie van het systeemconcept, maar was niet gerealiseerd in de eerste pilot. Tijdens de bespreking van het systeemconcept bleek de opdrachtgever nu veel concretere ideeën te hebben over de zoekmogelijkheden. Hij stelde voor om in het webpartscherm op trefwoord te kunnen zoeken en voor meer geavanceerde zoekopties te verwijzen naar een aparte zoekpagina.
Zelf had ik ook nieuwe ideeën over het zoeken gekregen. Omdat de velden die over een melding worden bijgehouden dynamisch zijn en verschillende rollen verschillende rechten hebben op de velden, zou het mooi zijn als de zoekpagina ook dynamisch wordt opgebouwd. Mijn idee was, om een gebruiker de mogelijkheid te geven te zoeken in alle velden in de database, waar hij de permissie op heeft om deze te zien. Op deze manier zal een gebruiker nooit velden tegen komen waar hij niet in mag zoeken.
Als er in de toekomst velden toegevoegd worden aan een soort, zal hier automatisch ook op gezocht kunnen worden. De opdrachtgever gaf aan dit een erg mooie oplossing te vinden. Daarop heb ik deze use-case zo uitgewerkt in het systeemconcept.

Het systeemconcept is te vinden in externe bijlage 2.

6.3 Pilotplan opstellen

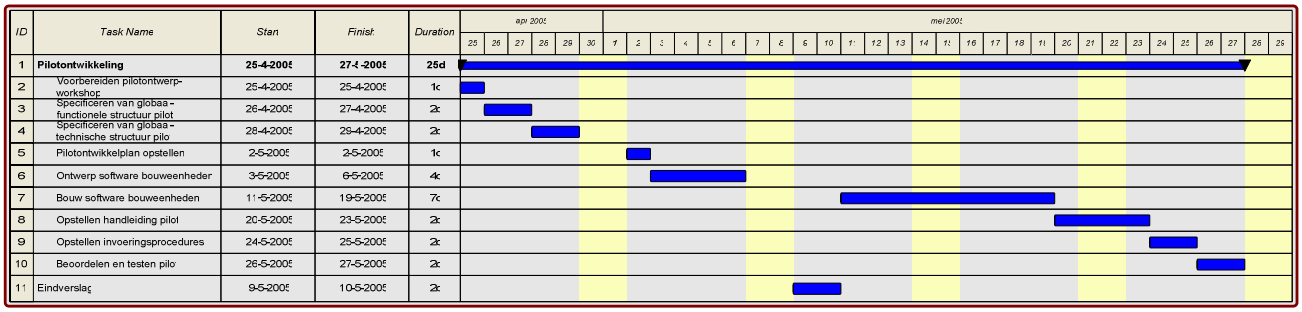
Tijdens de tweede pilot zou de gebruikers- en rechtenstructuur gerealiseerd worden. Dit was essentieel voor de applicatie. Daarom ben ik begonnen met het inschatten van de benodigde tijd om de bestaande use-cases aan te passen en de nieuwe use-case “koppelen gebruiker - rol” te realiseren. Deze use-case is nodig om gebruikers van de Sharepoint portal te koppelen aan een rol in de webpart.

Door de ervaring die ik had opgedaan tijdens de eerste pilot, kon ik een betere inschatting maken over de benodigde tijd per use-case voor de tweede pilot. De planning die ik had opgesteld voor de eerste pilot was te ruim genomen, daardoor was ik een week eerder klaar.

Ik had een dag tijd gereserveerd voor het verbeteren van de kwaliteit van mijn applicatiecode. Er ging namelijk een code-review uitgevoerd worden door een medewerker van Qurius ETX. Daaruit zouden enkele verbeterpunten komen. Ik zou deze bestuderen en kiezen welke verbeteringen ik ging doorvoeren. Tijdens het opstellen van het pilotplan, was de code-review nog niet uitgevoerd en wist ik nog niet welke verbeterpunten er naar voren zouden komen. Ik zal hier later op terugkomen.

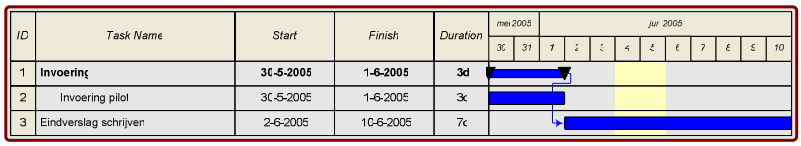
Niet alle punten uit de code-review was ik van plan te verbeteren. Mijn code werd namelijk beoordeeld aan de hand van de standaarden en richtlijnen van Qurius ETX. Aan het begin van dit project is afgesproken dat ik me daar niet aan hoef te houden.

De nieuwe planning voor de tweede pilot en het invoeren van deze pilot, is hieronder te vinden:



ID	Task Name	Start	Finish	Duration
1	**Pilotontwikkeling**	**25-4-2005**	**27-4-2005**	**25d**
2	Voorbereiden pilotontwerp-workshop	25-4-2005	25-4-2005	1d
3	Specificeren van globaa - functionele structuur pilot	26-4-2005	27-4-2005	2d
4	Specificeren van globaa - technische structuur pilo	28-4-2005	29-4-2005	2d
5	Pilotontwikkelplan opstellen	2-5-2005	2-5-2005	1d
6	Ontwerp software bouweenheder	3-5-2005	6-5-2005	4d
7	Bouw software bouweenheden	11-5-2005	19-5-2005	7d
8	Opstellen handleiding pilot	20-5-2005	23-5-2005	2d
9	Opstellen invoeringsprocedures	24-5-2005	25-5-2005	2d
10	Beoordelen en testen pilo	26-5-2005	27-5-2005	2d
11	Eindverslag	9-5-2005	10-5-2005	2d

Planning Pilot 2



ID	Task Name	Start	Finish	Duration
1	**Invoering**	**30-5-2005**	**1-6-2005**	**3d**
2	Invoering pilot	30-5-2005	1-6-2005	3d
3	Eindverslag schrijven	2-6-2005	10-6-2005	7d

Planning invoering

Het volledige pilotplan is te vinden in externe bijlage 4.

6.4 Testplan opstellen

Nadat ik in het pilotplan had afgesproken welke use-cases ik ging realiseren, heb ik het testplan opgesteld. Het testplan is opgesteld aan de hand van de uitwerking van de use-cases in het systeemconcept.
Het testplan van de eerste pilot was opgesteld nadat de use-cases gebouwd waren. Onbewust was ik toen snel geneigd om het testplan te baseren op hoe de applicatie was geworden en niet hoe het was afgesproken in het systeemconcept. Om dit te voorkomen heb ik het testplan nu vóór de pilotontwikkelfase geschreven, zodat het enkel en alleen gebaseerd was op het systeemconcept.

Het testplan, van de tweede pilot, is hetzelfde opgebouwd als de eerste versie van het testplan. Er worden acties genoemd die een bepaald resultaat hebben. Voor elke actie kan afgevinkt worden of het resultaat behaald wordt. Op deze manier wordt elke use-case, met de alternatieven voor elke use-case, doorlopen.
Omdat er in deze pilot onderscheid werd gemaakt tussen de verschillende actoren heb ik in het testplan hier ook rekening mee moeten houden. Om het testplan bruikbaar te houden en niet te groot te maken, werd niet elke actie voor elke actor getest. Er werd slechts voor twee verschillende actoren getest, één die de actie wel mocht uitvoeren en één die de actie niet mocht uitvoeren.

Het testplan is te vinden in externe bijlage 11.

7 Pilot 2: rechtenstructuur pilot

Nadat het pilotplan was opgesteld, met daarin een planning voor pilot twee, kon ik beginnen aan de pilotontwikkelfase. In dit hoofdstuk bespreek ik hoe ik tot het uiteindelijke ontwerp en de bouw van de tweede pilot ben gekomen. Na de bouw volgt nog een korte bespreking van het maken van de handleidingen en het uitvoeren van het testplan.

7.1 Globaal Functionele structuur opstellen

Het opstellen van de globaal functionele structuur had als doel om voor de werkelijke bouw duidelijk te krijgen wat en op welke manier gebouwd moet worden. Een groot deel van de functionele structuur was al beschikbaar vanuit de eerste pilot. Dit document heb ik uitgebreid met de functionele structuur van de tweede pilot.

Voor de tweede pilot heb ik onder andere beschreven hoe gebruikers aan een rol gekoppeld kunnen worden. Dit heb ik gedaan door eerst de stappen die ondernomen moeten worden globaal vast te leggen. Tijdens het ontwerp van de software-bouweenheden ben ik dit gaan verwerken in een sequentiediagram. Dit staat beschreven in paragraaf 7.4.3.

Hieronder volgt het stappenplan dat ik heb gemaakt voor het koppelen van gebruikers aan rollen. De stappen vertellen wat het systeem moet doen en zijn gedetailleerder dan de stappen in het systeemconcept. Dit stappenplan toont de stappen om alle gebruikers en mogelijke rollen op het scherm te tonen en de juiste rollen bij een gebruiker te selecteren.
Nadat de rollen van een gebruiker gewijzigd zijn, zijn de stappen beschreven die zorgen voor het updaten van de rollen.

Globaal stappenplan:

1) Alle mogelijke gebruikers ophalen
   Alle gebruikers die een rol kunnen krijgen moeten worden opgehaald
2) (Projectleider>) Gebruiker selecteren
   Vervolgens kan de projectleider een gebruiker selecteren
3) Nieuw gebruiker object aanmaken
   Na het selecteren van een gebruiker, wordt er een nieuw gebruiker object aangemaakt
4) Als de gebruiker al bestaat in de database, zijn gegevens inladen, anders nieuwe gebruiker in de database aanmaken.
   Niet alle gebruikers staan automatisch in de database van de applicatie, zie voor een uitleg hierover paragraaf 7.2.
5) Rechten die de gebruiker al heeft selecteren op de pagina

6) Na submit:
   Kijken naar het verschil tussen de geselecteerde rechten en de rechten die de gebruiker al heeft.
7) Aan de hand van het verschil de rechten van een gebruiker toevoegen of verwijderen.

De volledige functionele structuur is te vinden in externe bijlage 6.

7.2 Globaal Technische structuur opstellen

In de globaal technische structuur heb ik beschreven wat de overeenkomsten waren tussen de gebruikers die bestaan in Sharepoint en de gebruikers in mijn applicatie.

De Sharepoint portal houdt zelf een lijst bij met gebruikers van de portal, deze staan niet automatisch allemaal in de database van de webpart.
Ik ben een manier gaan bedenken om de gebruikers uit Sharepoint te gebruiken in de webpart. Eén manier zou zijn om een relatie te maken, in het datamodel, die verwijst naar de Sharepoint gebruiker tabel die in een andere database zit. Het nadeel van deze methode was, dat dit lastig te realiseren zou zijn, doordat Sharepoint zijn gebruikers op een geheel eigen manier beheert. De gebruikers worden versleuteld opgeslagen in een eigen database, een nette relatie naar het gebruikers ID zou dus niet mogelijk zijn.
Een tweede manier was om alle Sharepoint gebruikers ook op te slaan in de database van de webpart. Het voordeel hiervan was dat ik meer controle had over de gebruikers en minder afhankelijk was van de Sharepoint database.

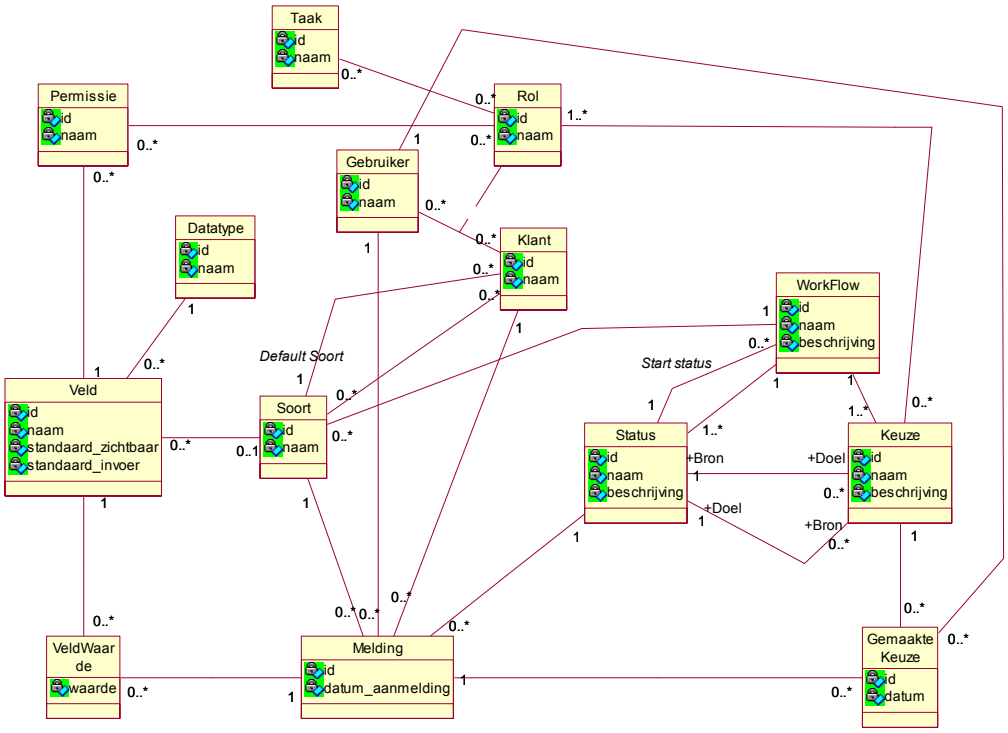
Vanuit de webpart kon ik gemakkelijk de Sharepoint gebruikersnaam ophalen van een gebruiker. Ik heb er voor gekozen om de Sharepoint gebruikersnaam op te halen en niet het ID, omdat ik zeker wist dat de gebruikersnaam altijd uniek zou zijn, ook al worden er gebruikers toegevoegd of verwijderd. Bij het ID wist ik dit niet. Een nieuwe gebruiker zou misschien het ID kunnen krijgen van een oude verwijderde gebruiker. Als ik dit ID zou gaan gebruiken in mijn database zouden er problemen kunnen ontstaan, omdat een verwijderde gebruiker in Sharepoint niet automatisch in mijn database wordt verwijderd.

Als uitbreiding bij mijn huidige datamodel heb ik de klasse Gebruiker gemaakt. Van een gebruiker wordt de naam en een uniek ID opgeslagen. Ook al is de naam uniek wordt er toch ook nog een uniek ID opgeslagen, zodat er binnen mijn applicatie verder met het unieke ID gewerkt kan worden. Een unieke sleutel mag namelijk geen betekenisvolle waarde zijn.
Alleen de koppeling met Sharepoint is dus op de gebruikersnaam.

De klasse Gebruiker kon gekoppeld worden met de klasse Melding. Een Gebruiker meldt namelijk Meldingen aan. In de eerste pilot was de klasse GemaakteKeuze reeds geïmplementeerd. Tussen deze klasse en de klasse Gebruiker kon een relatie worden gelegd, zodat er bijgehouden kon worden wie en wanneer welke keuze maakt.

Een wens van de opdrachtgever was om gebruikers verschillende rollen te kunnen geven per klant. Voor de ene klant kan een gebruiker projectleider zijn en voor de andere ontwikkelaar. Een gebruiker kan ook meerdere rollen per klant krijgen. Hij kan bijvoorbeeld projectleider en ontwikkelaar zijn voor dezelfde klant.
Dit is in het klassendiagram weergegeven door een associatie klasse. Op de manier waarop dit nu staat weergegeven lijkt het alsof er maar één rol mogelijk is voor een gebruiker per klant. Er zou eigenlijk een trio-associatie gebruikt moeten worden, daarmee wordt aangegeven dat er meerdere rollen mogelijk zijn per klant. Deze optie was echter niet beschikbaar in de modelleringstool, daarom heb ik gekozen om een associatie klasse te maken.
Deze relatie is het echter wel geïmplementeerd als trio-associatie, er zijn meerdere rollen voor een gebruiker per klant mogelijk.

Het model is hierna verbeterd, zo is er een aparte klasse gekomen voor de datatypes en is er een klasse Taak gekomen. Er zijn enkele relaties bijgekomen om bijvoorbeeld de standaard soort aan te geven. Het uiteindelijke klassendiagram staat op de volgende pagina.



Het uiteindelijke klassendiagram van het datamodel

De klasse Taak is verantwoordelijk voor de verschillende taken in het systeem. Een taak is bijvoorbeeld het invoeren van een melding. Het definiëren van de taken wordt besproken in paragraaf 7.4.4.
Ik heb met de opdrachtgever besproken of het aanpasbaar moet zijn welke rol welke taak mag uitvoeren. De opdrachtgever gaf aan dit erg mooi te vinden. Het ligt ook in de lijn met de rest van het systeem. De stappen die een gebruiker in de workflow mag nemen en welke gegevens een gebruiker mag zien en wijzigen zijn namelijk ook aanpasbaar. Ik heb er voor gekozen om deze wens mee te nemen en dus was de extra klasse Taak nodig.
Wanneer ik er voor had gekozen dit niet te doen, zouden er later zeer moeilijk nieuwe rollen kunnen worden toegevoegd aan het systeem. Een nieuwe rol zou dan geen taken in het systeem kunnen uitvoeren.

De volledige technische structuur is te vinden in externe bijlage 7.

7.3 Pilotontwikkelplan

Na de globaal technische en functionele structuur te hebben gedefinieerd heb ik het pilotontwikkelplan opgesteld. Hierin heb ik de pilot opgedeeld in pilotdelen. Er zouden twee nieuwe use-cases ontwikkeld worden, dit werden beiden pilotdelen.

In het pilotontwikkelplan heb ik ook beschreven welke concrete kwaliteits-verbeterpunten ik zou doorvoeren naar aanleiding van de code-review door een medewerker van Qurius ETX.

Met de volgende punten zou ik rekening houden. Ik heb deze ook met terugwerkende kracht in bestaande code verbeterd:

- Een typed DataSet niet untyped aanspreken.
- Het gebruik van vaste waarden zoals 1 of 2 vermijden.
- Betere implementatie van de error afhandeling

Daarnaast werden de bestaande use-cases aangepast, zodat de rollen en rechtenstructuur werd geïmplementeerd. Ik heb dit onderverdeeld in drie pilotdelen:

- Taak autorisatie
  Taak autorisatie houdt in dat de applicatie bepaalt of een gebruiker gerechtigd is een bepaalde actie in het systeem uit te voeren. Dit kan het invoeren van een nieuwe melding zijn of het bekijken van gegevens over een melding.

- Gegevens autorisatie
  Gegevens autorisatie bepaalt welke gegevens een gebruiker over een melding mag zien en wijzigen.

- Workflow autorisatie
  Workflow autorisatie bepaalt tenslotte welke stappen in de workflow gemaakt mogen worden door welke gebruikers.

Van de pilotdelen heb ik de bouwstenen gedefinieerd. Dit waren grofweg de acties die ondernomen moesten worden om het pilotdeel te voltooien. Voor de taak autorisatie waren er drie bouwstenen:

- Verschillende taken onderscheiden
- Methode bouwen die de taak goedkeurt of afkeurt
- Methode implementeren op alle pagina's

Na de pilotdelen te hebben vastgesteld ben ik een volgorde van ontwikkeling gaan bepalen. Deze pilotontwikkelfase was vastgesteld als een timebox van 5 weken. Er was immers geen uitloop mogelijk, doordat ik aan het einde kwam van de afstudeerperiode. Het was dus belangrijk om de ontwikkelvolgorde zorgvuldig te bepalen. Hierbij heb ik gekeken naar de bruikbaarheid van het systeem als er een pilotdeel niet gebouwd wordt.

Als het koppelen van gebruikers aan rollen niet gebouwd wordt, kan het systeem niet ingezet worden. Dit onderdeel zou dus sowieso gebouwd moeten worden.
De verschillende vormen van autorisatie waren ook belangrijk, deze waren de kern van de gehele pilot. Al bij het opstellen van pilot één stond vast dat de gebruikers en rechten in pilot twee aan bod moeten kwamen.
Tot slot bleef het zoeken over. Deze functionaliteit was ook belangrijk, maar minder dan de voorgaande. Het zou tevens minder moeite zijn om deze functionaliteit later in het systeem te integreren dan bijvoorbeeld taak autorisatie, wat veranderingen in het gehele systeem teweegbrengt.

Het koppelen van de gebruikers en rollen zou dus als eerste gebeuren, daarna de verschillende vormen van autorisatie en op het einde de mogelijkheid tot zoeken.

Het volledige pilotontwikkelplan is te vinden in externe bijlage 5.

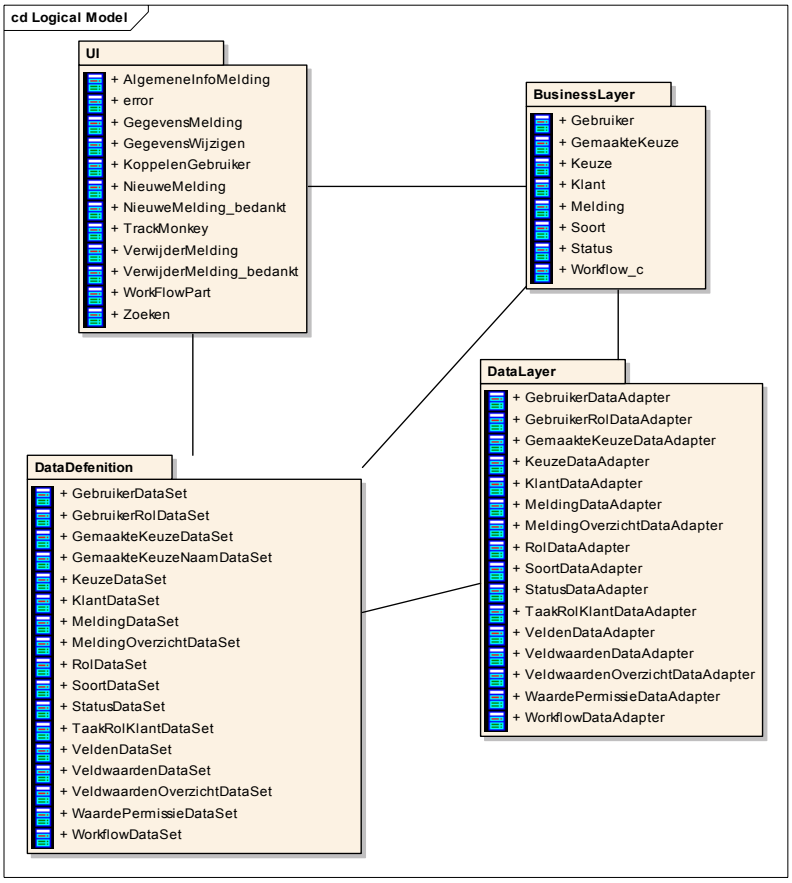
7.4 Ontwerp software-bouweenheden

Het ontwerp van de software-bouweenheden heeft als doel dieper in te gaan op het globale ontwerp uit de functionele en technische structuur.
In dit hoofdstuk bespreek ik onder andere de afhandeling van de authenticatie en autorisatie.

7.4.1 Drie lagen model

Zoals ook tijdens de eerste pilot is er gewerkt met het drie lagen model van User Interface, BusinessLayer, DataLayer en de ondersteunende DataDefenition.

De elementen in de User Interface en de BusinessLayer had ik reeds bepaald, dit zijn respectievelijk de verschillende pagina's en de klassen in het logische model.
De DataLayer bevat de DataAdapters en de DataDefenition de DataSets. In onderstaand schema is de relatie en inhoud van de drie lagen en de DataDefenition weergegeven.



Drie lagen model

7.4.2 Authenticatie

Voordat er geautoriseerd kan worden (bepalen wat een gebruiker mag), vindt er eerst authenticatie plaats (bepalen wie de gebruiker is). Doordat mijn applicatie op Sharepoint kwam te draaien, was het niet nodig authenticatie te implementeren. Dit deed Sharepoint namelijk al.
Binnen Sharepoint kon ik de gebruikersnaam opvragen van de gebruiker die de pagina opvroeg. Ik ben er vanuit gegaan dat Sharepoint een goede authenticatie had en ik er dus van op aan kon dat ik de werkelijke gebruiker die bij de gebruikersnaam hoorde, te pakken had.

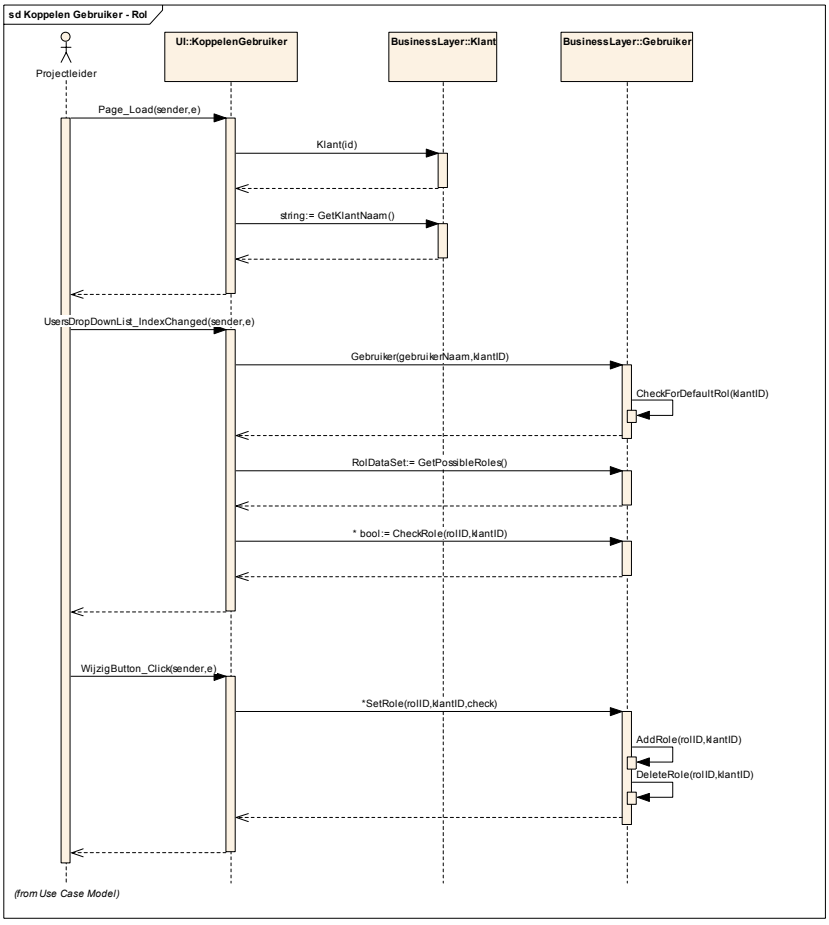
7.4.3 Koppelen gebruiker – rol

Aan de hand van het stappenplan, dat is opgesteld bij de globaal functionele structuur, heb ik een sequentiediagram gemaakt voor het koppelen van gebruikers aan rollen. Dit is onderaan deze pagina te zien.

In het sequentiediagram worden de acties weergegeven, die uitgevoerd worden na het selecteren van een gebruiker en na het wijzigen van de rollen van een gebruiker. Als een gebruiker geselecteerd wordt, wordt eerst opgevraagd welke rollen hij mogelijk kan krijgen. Daarna wordt opgevraagd welke van deze rollen hij al heeft. Dit wordt gepresenteerd als een lijst met checkboxes. Het screenshot geeft weer hoe dit er uit kwam te zien na het bouwen.

Screenshot van het uiteindelijke resultaat van de use-case "koppelen van een gebruiker aan een rol".

Na het klikken op de "wijzigen en opslaan" knop, zullen de checkboxes op de pagina worden vergeleken met de rollen die de gebruiker had. Aan de hand daarvan worden rollen voor deze gebruiker aangemaakt of verwijderd.



Sequentiediagram "koppelen gebruiker - rol"

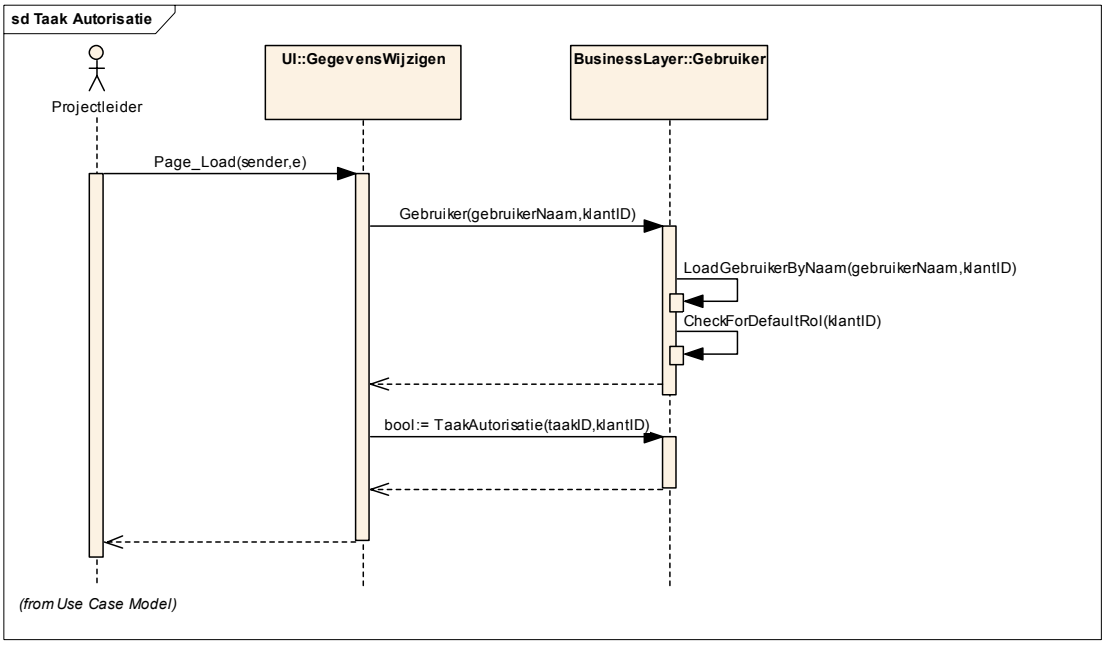
7.4.4 Taak autorisatie

De taak autorisatie is verantwoordelijk voor het bepalen of een gebruiker een taak mag uitvoeren. In het use-case diagram in het systeemconcept staat duidelijk aangegeven welke actor welke use-case mag uitvoeren. Tijdens het opstellen van de globaal technische structuur is echter afgesproken dat de taken die een rol mag uitvoeren aanpasbaar moeten zijn.

Ik heb er voor gekozen om elke taak een nummer te geven in zowel de applicatiecode als in de database. Het openen van de pagina met gegevens over een melding is bijvoorbeeld taaknummer 1. In de database zal deze taak ook staan met nummer 1. In een andere tabel zal bijgehouden worden welke rollen taaknummer 1 mogen uitvoeren.

Om elke taak van een nummer te voorzien ben ik gaan inventariseren welke taken er bestonden, dit heb ik gedaan aan de hand van de use-cases in het systeemconcept. Elke use-case was een taak. Enkel bij de use-case zoeken waren er twee taken, namelijk normaal zoeken en geavanceerd zoeken.

Om te controleren of een taak uitgevoerd mag worden, wordt er een operatie aangeroepen bij de klasse Gebruiker. Deze operatie krijgt als parameter het taaknummer en het klantnummer mee, omdat een gebruiker per klant andere rollen kan hebben.

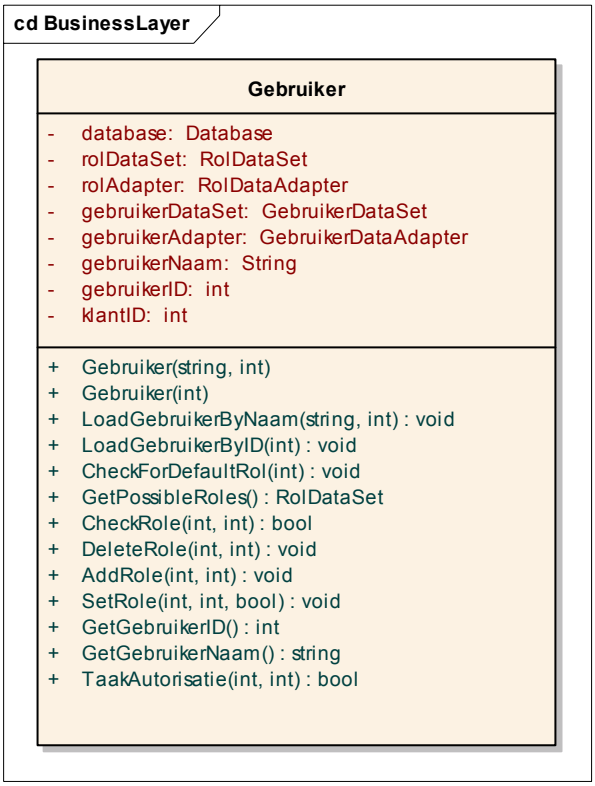


Sequentiediagram voor de Taak Autorisatie

De operatie TaakAutorisatie geeft een boolean terug, dit is een true of false. Deze boolean bepaalt of de taak wel of niet uitgevoerd mag worden.

In de verschillende pagina's zijn de vervolg stappen van de taak autorisatie geïmplementeerd. Als een gebruiker bijvoorbeeld geen toegang mag hebben, wordt hij doorgestuurd naar een pagina met een melding waarop te zien is dat hij geen toegang heeft tot de opgevraagde pagina.

De operatie voor taak autorisatie is geïmplementeerd in de klasse Gebruiker. Hieronder zijn de attributen en operaties van deze klasse te zien.



Attributen en operaties van de klasse Gebruiker

7.5 Bouw software-bouweenheden

Tijdens de bouw van de software-bouweenheden ben ik de bouweenheden, zoals gedefinieerd in het pilotontwikkelplan, gaan bouwen.
In dit hoofdstuk zal ik het bouwen van de workflow en gegevens autorisatie bespreken. Vervolgens komt de functionaliteit voor het zoeken aan bod en tot slot bespreek ik wat ik aan kwaliteitsverbetering heb gedaan.

7.5.1 Bouw Workflow Autorisatie

Onderdeel van de pilotontwikkelfase was het bouwen van de workflow autorisatie. Deze zorgt er voor dat enkel een gerechtigd persoon een stap in de workflow mag maken.
In pilot één heb ik reeds de functionaliteit geïmplementeerd om de workflow te kunnen doorlopen. Hierbij werd echter nog geen rekening gehouden met de verschillende rollen. Bij de handleiding van de eerste pilot had ik een stored procedure gemaakt, die de workflow in de database wegschrijft. Om de juiste rollen en rechten in de database te krijgen om mee te testen, heb ik deze stored procedure als eerste uitgebreid, zodat er ook weggeschreven werd welke rol welke stap mocht uitvoeren.
De stappen die getoond worden in de applicatie, worden opgehaald met een andere stored procedure. Deze stored procedure heb ik aangepast zodat het gebruikers ID meegegeven kon worden en bij deze gebruiker de juiste rol gezocht werd. De rollen van een gebruiker kunnen anders zijn per klant, dus ook het klantnummer zal meegegeven moeten worden. Vervolgens werd van elke rol opgehaald welke keuzes er gemaakt mochten worden in de workflow.
In het systeemconcept is afgesproken dat een gebruiker meerdere rollen mag hebben, hierdoor kan het voorkomen dat een gebruiker twee rollen heeft die allebei een bepaalde keuze mogen maken. In mijn stored procedure heb ik aangegeven dat elke keuze slechts éénmaal mag voorkomen in de resultaten.

Door de scheiding tussen code en database queries, door middel van stored procedures, waren er vrijwel geen aanpassingen nodig in de code. De code moest alleen zo worden aangepast dat het gebruiker- en het klantnummer werd meegegeven aan de stored procedure.

7.5.2 Bouw Gegevens Autorisatie

Gegevens autorisatie houdt in dat enkel gebruikers met de juiste rol, gegevens mogen zien of wijzigen. Bij de gegevens autorisatie speelde het probleem dat een gebruiker conflicterende rollen kon hebben. Het verschil ten opzichte van de workflow autorisatie was dat het niet simpelweg mocht of niet mocht. Er waren drie mogelijkheden, namelijk: niet zichtbaar, zichtbaar en zichtbaar en wijzigbaar.

Voor de use-case “gegevens melding wijzigen” wilde ik de gegevens over een melding presenteren. Gegevens die niet zichtbaar mogen zijn worden niet getoond, gegevens die gewijzigd mogen worden, worden wijzigbaar getoond. De gegevens die alleen bekeken mogen worden, worden niet wijzigbaar gepresenteerd.

Om dit te kunnen realiseren, had ik nodig:

- Veldnaam van de zichtbare en wijzigbare velden (tabel Velden)
- Waarde van dat veld (tabel Veldwaarden)
- De rechten op dat veld (tabel Permissie)
- Het datatype van de waarde (tabel Velden)

Om deze gegevens te verzamelen, had ik de optie om een stored procedure te schrijven die alle benodigde gegevens ophaalde in één DataSet. Het voordeel hiervan was, dat je alle gegevens die bij elkaar horen in één rij van je DataSet hebt staan. Dit is makkelijker om mee te werken.

De andere mogelijkheid was, om de veldnamen op te halen en later deze velden te doorlopen om de waarden en rechten bij dat veld ophalen. Dit is vergelijkbaar met de opbouw van het overzicht van alle meldingen (zie paragraaf 5.5.6).
Het voordeel van de tweede oplossing was, dat ik bij deze oplossing gebruik kon maken van reeds bestaande DataSets en DataAdapters. Deze oplossing had echter als nadeel dat de velden in de code één voor één doorlopen moesten worden om de veldwaarde op te halen. Dit zou meer database activiteit veroorzaken.

De eerste oplossing had als nadeel dat de te maken stored procedure zeer complex zou worden. Tevens zou er een DataSet gemaakt moeten worden die niet rechtstreeks overeenkwam met één tabel in de database. De DataSet zou bestaan uit een combinatie van tabellen.
Ik heb daarom gekozen voor de tweede oplossing. De stored procedure zou hierdoor overzichtelijk blijven en ik zou gebruik kunnen maken van bestaande DataSets en DataAdapters.

De velden die bij de melding horen worden nu opgehaald. Daarna wordt voor elk veld de waarde opgehaald en bepaald wat de rechten van de gebruiker op het veld zijn. Aan de hand van de rechten wordt de waarde gepresenteerd in een wijzigbare textbox of als normale tekst.

7.5.3 Implementatie Datatype

Elk veld dat wordt bijgehouden over een melding is van een bepaald datatype, dit datatype bepaalt onder andere de presentatie van de data. Zo wordt een veld met als datatype “longtext” gepresenteerd in een groot tekstvlak in plaats van een kleine tekstregel.

Tijdens de evaluatie van de eerste pilot, gaf de opdrachtgever aan het erg prettig te vinden als er de mogelijkheid zou zijn tot de controle van verplichte invoer of de controle van invoer op enkel getallen.
De lijst met velden werd opgebouwd door een DataList. Deze DataList herhaalt de plaatsing van de veldnaam en textbox voor alle velden. Hieraan heb ik de plaatsing van twee validators toegevoegd. Eén validator om te controleren of een veld niet leeg is en één om te controleren of er een getal wordt ingevoerd.
Deze controle is niet voor elk veld nodig. Aan de hand van het datatype wordt bepaald of de validator ingeschakeld of uitgeschakeld wordt.

Het controleren op de invoer van een getal heb ik opgelost met een regular expressie. Dit is een patroon waar een tekst aan moet voldoen. Het patroon heb ik zo opgebouwd dat er zowel positieve als negatieve getallen ingevoerd konden worden. Ook is het mogelijk om een komma of punt te gebruiken, voor het gebruik van decimalen.

7.5.4 Aanpassen aan Sharepoint Layout

De opdrachtgever wilde niet dat gebruikers afhankelijk waren van de terugknop in de browser. In de applicatie zelf moest ook een mogelijkheid komen om terug te gaan naar waar je vandaan kwam.

Doordat de webpart op verschillende pagina's kan komen te staan, zal de terugknop niet altijd leiden naar dezelfde pagina. Ik moest een oplossing bedenken om er voor te zorgen dat gebruikers van de webpart altijd weer terug konden naar de webpart, onafhankelijk van de plaats waar de webpart stond.

De mogelijke oplossingen voor dit probleem heb ik onderzocht. Het was mogelijk om dit op te lossen door middel van een sessie object. Een sessie object is een object dat wordt bijgehouden op de server en gekoppeld is aan een individuele bezoeker van de webpart. Tijdens het bezoek aan de webpart zal het sessie object bestaan en na afloop van het bezoek zal het verwijderd worden.
Bij het openen van de webpart kan het sessie object aangemaakt worden, met daarin de locatie van de webpart. Op pagina's met een terugknop kan het sessie object vervolgens uitgelezen worden en de juiste link naar de webpart geplaatst worden.
Het nadeel van deze oplossing is, is dat het sessie object server gebonden is. Wanneer de webpart op een pagina staat geïnstalleerd die verdeeld is over meerdere servers kan dit problemen opleveren. De eerste pagina wordt op server één opgevraagd en de volgende pagina op server twee. Het sessie object dat op server één is aangemaakt, zal niet bekend zijn bij server twee. Hierdoor zal het alsnog onmogelijk zijn om terug naar de webpart te gaan.

Dit probleem kon worden verholpen door voor een zogenaamde stateless oplossing te gaan. Stateless houdt in dat elk verzoek op de server wordt behandeld als een individueel verzoek. Er hoeft geen verzoek aan vooraf gegaan te zijn en er hoeven geen gegevens over voorgaande verzoeken te worden bijgehouden.

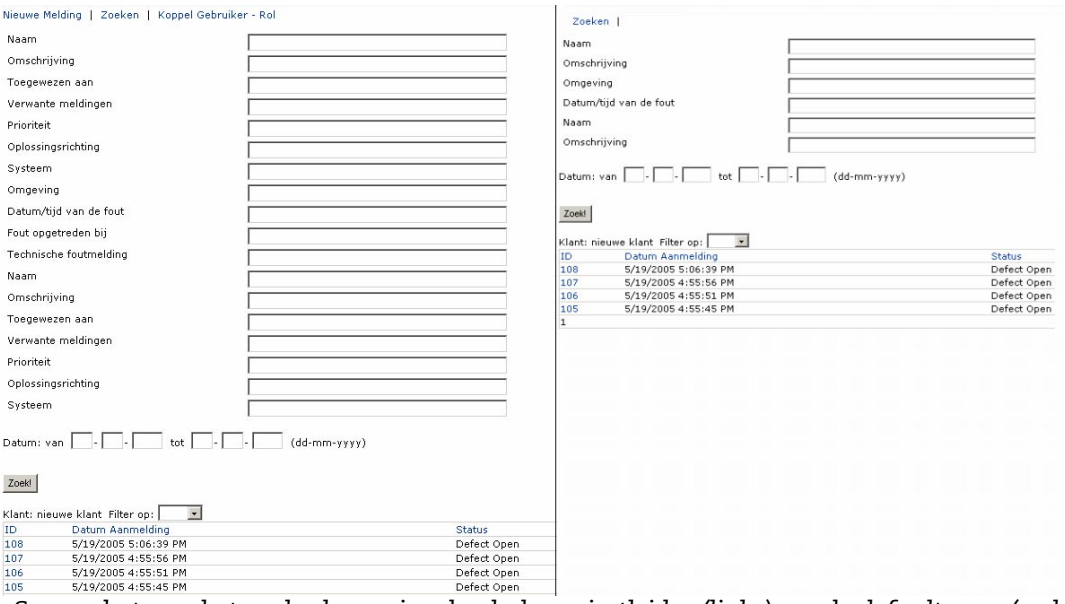
Door de plaats waar de webpart staat mee te geven in de URL als variabele, weten de andere webpagina's ook waar de webpart staat. Het nadeel van deze oplossing was, dat het ingewikkelder werd naarmate er meer doorklikmogelijkheden waren. Vanuit het bekijken van gegevens over een melding kan je ze wijzigen, vanuit daar kan je een melding verwijderen en dit bevestigen. Na de bevestiging van het verwijderen moet het mogelijk zijn om terug te gaan naar de webpart. De plek waar de webpart staat zal dus met elke stap meegegeven moeten worden. Ook als er tijdens een stap op een terugknop wordt geklikt moet het mogelijk zijn om uiteindelijk weer bij de webpart uit te komen.

Uiteindelijk heb ik gekozen voor de stateless oplossing. Door de ervaring die ik inmiddels had opgedaan met Sharepoint, achtte ik de kans groot dat Sharepoint en het Smartpart specifieke instellingen nodig had om het sessie object te laten werken. Om dit te voorkomen heb ik gekozen om voor de stateless oplossing te gaan, deze werkt sowieso. Er hoeven geen wijzigingen op de server voor plaats te vinden.

7.5.5 Zoeken

Een nieuwe use-case die tijdens deze pilot geïmplementeerd werd, was de use-case zoeken. In het systeemconcept stond beschreven dat de velden waarop gezocht kon worden, afhingen van de rollen die een gebruiker had. Als de gebruiker geen recht heeft om een veld van een melding te zien, mag hij er ook niet op zoeken.
Ik heb een stored procedure geschreven, die de velden waarop gezocht mocht worden ophaalt. Deze stored procedure houdt rekening met de rol van de gebruiker.

Het resultaat is in onderstaande screenshots te zien.



Screenshot van het zoekscherm, ingelogd als projectleider (links) en als default user (rechts)

7.5.6 Kwaliteitsborging

Na afloop van de eerste pilot, is er door een medewerker van Qurius ETX een kwaliteitscontrole uitgevoerd. Het doel van deze kwaliteitscontrole was voor de medewerker het oefenen van het beoordelen van code aan de hand van de standaarden en richtlijnen die bij de opdrachtgever in gebruik zijn.
Omdat ik niet werkte volgens de standaarden en richtlijnen van Qurius ETX, hoefde ik mij niet al het commentaar aan te trekken. Er kwamen wel enkele nuttige tips uit het gesprek dat ik had over mijn code.
Een punt van commentaar op mijn code was dat ik de waarde 1 en 2 gebruikte voor de invoervelden bij een melding, deze nummers stonden respectievelijk voor het unieke ID van de velden titel en omschrijving in de database. Het was niet mogelijk om deze nummers dynamisch op te vragen, omdat er meerdere velden met dezelfde naam mogelijk zijn. Er kon dus niet vanuit gegaan worden dat een veld met de naam “titel” het juiste veld was.

Doordat de velden die over een melding worden bijgehouden aanpasbaar zijn, kwam ik op het idee om ook de invoervelden aanpasbaar te maken. Hiervoor moest het datamodel aangepast worden, zodat er van elk veld bijgehouden kon worden of dit ingevoerd mocht worden.

Het gevolg van deze wijziging was dat het systeemconcept aangepast moest worden. De invoervelden stonden immers nu niet meer vast. Ook de handleiding bij deze pilot zou dit moeten vermelden, zodat de opdrachtgever weet hoe hij dit kan instellen.

7.6 Handleiding en invoeringsprocedures opstellen

Bij de opgeleverde pilot horen drie handleidingen. Vooral de beheer en installatie handleiding waren erg belangrijk. Het beheer van de applicatie vindt namelijk plaats via de database. De installatiehandleiding is belangrijk, omdat er specifieke Sharepoint instellingen gewijzigd moeten worden.

7.6.1 Gebruikershandleiding

In de gebruikershandleiding is beschreven hoe gebruikers de applicatie kunnen gebruiken. De applicatie spreekt meestal voor zich, maar voor alle volledigheid heb ik toch een gebruikershandleiding gemaakt. De handleiding is opgesteld aan de hand van het systeemconcept en het uiteindelijke resultaat van de bouw van de software-bouweenheden.

Per pagina heb ik een screenshot toegevoegd en de werking van alle onderdelen op de pagina uitgelegd.

De gebruikershandleiding is te vinden in externe bijlage 8.

7.6.2 Beheerhandleiding

In de webpart zelf zijn zeer beperkte beheermogelijkheden. Het is enkel mogelijk om gebruikers aan rollen te koppelen. Er is echter veel meer in te stellen, hoe dit gedaan kan worden heb ik beschreven in de beheerhandleiding.
Ik heb voor veel voorkomende taken, zoals het toevoegen van een klant en het aanmaken van een nieuwe veld, precies beschreven wat er moet gebeuren. Ook de afhankelijkheden en vervolgstappen heb ik opgeschreven. Zoals dat een nieuw veld gekoppeld aan een soort kan worden.

Voor alle veel voorkomende handelingen die beschreven staan in de handleiding, heb ik stored procedures gemaakt. Er hoeft dus geen SQL code uitgevoerd of aangepast te worden. Per stored procedure staat aangegeven welke parameters er meegegeven kunnen worden.
De stored procedures heb ik zo veel mogelijk werk laten doen. Bij het aanmaken van een nieuwe klant dient er een default_soort_id opgegeven te worden. Dit ID dient echter ook nog gekoppeld te worden in de tabel SoortKlant. Ik heb de stored procedure zo gemaakt, dat dit automatisch gedaan wordt bij het toevoegen van een klant.
Hierdoor is het makkelijker om een nieuwe klant toe te voegen en is het ook makkelijker om later een user interface te maken. Vanuit de user interface hoeft enkel de stored procedure aangeroepen te worden.

De beheerhandleiding is te vinden in externe bijlage 10.

7.6.3 Installatiehandleiding

De installatiehandleiding heb ik opgezet door een stappenplan op te stellen voor de verschillende onderdelen van de installatie, zoals de smartpart, de webpart en de database.

Voordat de smartpart en webpart werken op de Sharepoint server, moeten er beveiligings instellingen worden veranderd. Om zeker te weten welke dit precies waren, heb ik Sharepoint opnieuw geïnstalleerd. De instellingen stonden toen weer op hun standaard waarde. Vervolgens heb ik elke verandering in de instellingen als stap in de installatie opgeschreven.
Om de juistheid van de handleiding te controleren, heb ik nogmaals Sharepoint geïnstalleerd. Ditmaal heb ik mijn installatiehandleiding getest, door alle stappen uit te voeren.

Na de installatie van de webpart dienen er op de database server de nodige tabellen aangemaakt te worden. Er is bij Qurius ETX een programma in gebruik genaamd “SQL Project Explorer”. Hierin kunnen sql script bestanden geladen worden. Deze bestanden bevatten bijvoorbeeld de code om tabellen aan te maken. Door middel van de SQL Project Explorer kunnen deze scripts gemakkelijk uitgevoerd worden.
Ik heb een script laten genereren wat al mijn tabellen en stored procedures in de database kan terugzetten.

Nadat alle tabellen en stored procedures in de database gezet zijn, moeten er nog enkele tabellen gevuld worden, de zogenaamde configuratietabellen. Bijvoorbeeld de tabel met de mogelijke permissies. In de installatiehandleiding heb ik precies beschreven welke stored procedures uitgevoerd moeten worden om de configuratie tabellen te vullen.
Tevens heb ik de optie beschreven om enkele extra stored procedures uit te voeren die standaard de workflow van een fout en wijzigingsverzoek en de juiste velden bij een soort aanmaakt.

De installatiehandleiding is te vinden in externe bijlage 9.

7.7 Beoordelen en Testen

Tijdens het ontwikkelen zijn de verschillende bouwstenen van de applicatie getest. Een onderdeel dat ik bijvoorbeeld heb getest op fouten, is de regular expression die gebruikt werd bij de invoer van bepaalde velden. Deze regular expression zou er voor moeten zorgen dat er enkel getallen ingevoerd mochten worden. Dit heb ik gecontroleerd door verschillende sets met waarden door de regular expression te laten testen. In onderstaande tabel zouden de waarden in set 1 moeten worden goedgekeurd en de waarden in set 2 afgekeurd.

Set 1	Set 2
1	Fout
450	22a
99999999999999999	22a22
1.9	22,a
-1.9	A,22
1,9	23!
-1,9	-a
-999999999999,9999999999	-1,a

Het testplan, om de gehele applicatie te testen, was reeds opgesteld tijdens de definitie studie. Aan de hand van het systeemconcept zijn de stappen in het testplan beschreven. Door middel van het testplan kan getest worden of de applicatie werkt, maar ook of het voldoet aan de afspraken die gemaakt zijn in het systeemconcept.

Het testplan is weer uitgevoerd door Michiel Bruins. Hij heeft alle stappen in het testplan doorlopen en commentaar toegevoegd. Uit het commentaar bleek dat de hij nog enkele wijzigingen wilde zien.
Zo wilde hij dat je na het wijzigen van gegevens weer bij het overzicht uitkwam van alle meldingen, in plaats van bij de gegevens over de melding. Zulke wijzigingen heb ik doorgevoerd en ook aangepast in het systeemconcept, zodat alles consistent met elkaar bleef.

8 Invoering

Nadat Michiel Bruins het testplan had uitgevoerd, hebben we het over de invoering van de tweede pilot gehad. Hij zei dat er vanuit de organisatie enkele wensen waren bijgekomen. Eén wens was bijvoorbeeld een kolom die een berekening uitvoert op basis van waarden in andere kolommen.
Deze nieuwe wens was pas sinds kort bekend, daarom was daar tot nu toe geen rekening mee gehouden.

Het resultaat van de tweede pilot bood echter genoeg voordelen ten opzichte van de huidige applicatie. De applicatie zal dus wel ingezet gaan worden voor enkele projecten. Ik had nog slechts een beperkte tijd beschikbaar. Omdat ik in de laatste week van mijn afstuderen geen grote toevoegingen meer wilde doen aan mijn eindverslag, heb ik besloten om de invoering van de applicatie buiten beschouwing te laten.

Er hoefde daardoor niet gehaast te worden om de applicatie in te voeren voor het einde van mijn afstuderen. Daardoor kon ik de tijd die ik nog had, besteden aan de verbeteringen naar aanleiding van de test van de tweede pilot, zodat de tweede pilot in ieder geval goed gebruikt kon worden.

9 Evaluatie

In dit hoofdstuk is de evaluatie van mijn afstuderen te vinden. De evaluatie is onderverdeeld in proces en product.

9.1 Proces

De proces evaluatie evalueert onder andere de gevolgde methode, de begeleiding en de verschillende fases van de ontwikkelmethode.

9.1.1 Methode

Tijdens dit project heb ik gewerkt met de methode IAD. Op school had ik hier al ervaring mee opgedaan, maar waren de projecten meestal te kort om er intensief gebruik van te kunnen maken. IAD hielp mij bij het gestructureerd te werk gaan, zo heb ik eerst de systeemeisen bepaald en de eisen met de belangrijkste prioriteit in het systeemconcept verwerkt. In het pilotplan koos ik, op basis van het systeemconcept, wat ik ging ontwikkeling in de pilots.
Er was continu een duidelijke lijn zichtbaar. Alles wat uiteindelijk is gerealiseerd in de applicatie is terug te leiden via de bouwstenen van de pilot, naar het pilotontwikkelplan, naar het pilotplan, naar het systeemconcept, tot uiteindelijk een eis in de systeemeisen.
Ik ben blij dat ik niet heb gekozen om met de ontwikkelmethode van Qurius ETX te werken. Het zou te veel tijd hebben gekost om deze ontwikkelmethode te leren aan het begin van het project. Ook zou ik dan veel meer tijd zijn kwijt geweest aan het definiëren van de systeemeisen en het systeemconcept. In de ontwikkelmethode van Qurius ETX is het namelijk gebruikelijk om alle systeemeisen van tevoren te definiëren. Hierdoor zou ik waarschijnlijk te weinig tijd over hebben gehad om de applicatie werkelijk te realiseren.
De opdrachtgever had aangegeven graag tastbaar resultaat te zien aan het einde van de afstudeerperiode. Door het uitvoeren van twee pilots en de definitie studie niet onnodig groot te maken, is dit gelukt. IAD was dus de juiste keuze voor mijn afstuderen.

9.1.2 Begeleiding

Het afstudeerbedrijf heeft de keuze gemaakt om mij enkel met één contactpersoon over de opdracht te laten communiceren, namelijk Michiel Bruins. Aan de ene kant beperkte dit de mogelijkheden, zo was het niet mogelijk om een échte workshop te houden met toekomstige gebruikers en al hun wensen te inventariseren.
Aan de andere kant vond ik het een prettige manier van werken. Ik zat op dezelfde kamer als Michiel Bruins en hierdoor was het gemakkelijk om kort te overleggen. Ook het maken van een afspraak met één persoon is een stuk gemakkelijker dan elke keer met een groep personen.
Gemiddeld sprak ik elke week ongeveer een uur met Michiel Bruins. Dit leverde veel bruikbare informatie op.

Om mij bij het proces te begeleiden was Karel van der Lelij mijn begeleider vanuit het bedrijf. In een wekelijks gesprek bespraken we de voortgang van de afstudeeropdracht aan de hand van de planning. Omdat ik geen grote problemen of tegenslagen heb gehad, konden de gesprekken vaak kort zijn. Toch heb ik veel aan de begeleiding gehad, omdat ik tips kreeg over een beter verloop van mijn afstuderen en er terechtkon met vragen over bijvoorbeeld mijn eindverslag.

9.1.3 Planning

Na elke fase heb ik mijn planning aangepast. Om een planning te maken voor de pilotontwikkelfase maakte ik een schattig per use-case. Tevens probeerde ik rekening te houden met factoren zoals het bekend raken met Sharepoint en ASP.NET. De planning voor de eerste pilot was te ruim genomen, ik bleek een week eerder klaar te zijn.
Bij het maken van de planning van de tweede pilot, heb ik rekening gehouden met de ervaring uit de eerste pilot. Per use-case heb ik minder tijd vrijgemaakt en voor andere activiteiten, zoals het schrijven van de handleidingen en de kwaliteitsverbetering, heb ik enkele dagen meer tijd gereserveerd.

De planning van pilot twee bleek veel beter te kloppen ten opzichte van de werkelijkheid. Ik ben dan ook tevreden met de manier van plannen.

9.1.4 Definitie studie Fase

Door de goede communicatie met Michiel Bruins is de definitie studie goed verlopen. Hij gaf mij veel eisen en wensen en kon goed aangeven welke belangrijker waren dan andere.
Niet alle eisen en wensen zijn ver uitgediept. De definitie studie heb ik bewust beperkt gehouden, om niet te veel tijd kwijt te zijn met het inventariseren van eisen en wensen waarvoor ik waarschijnlijk toch geen tijd zou hebben om die te realiseren tijdens mijn afstuderen. Hierdoor heb ik tijd bespaard, anders had ik nog makkelijk een maand of langer bezig kunnen zijn met de definitie studie.

Tijdens de definitie studie heb ik ook een onderzoek uitgevoerd of de gewenste applicatie het beste bereikt kon worden met maatwerk of met een bestaand pakket. Ik ben blij dat ik dit onderzoek heb uitgevoerd, omdat het mijn afstudeerproject als het ware rechtvaardigde. Doordat uit het onderzoek bleek dat maatwerk de juiste keuze was, weet ik dat mijn afstudeerproject niet overbodig is geweest en het sneller opgelost had kunnen worden met een standaard pakket.

9.1.5 Pilotontwikkeling Fase

Voor ik begon aan deze afstudeeropdracht had ik nog geen enkele ervaring met ASP.NET. Tijdens het opstellen van de functionele en technische structuur van de eerste pilot, heb ik daarom veel geëxperimenteerd met ASP.NET. Doordat ik al wel veel ervaring had met Java, wat net als ASP.NET object georiënteerd is, had ik geen grote problemen bij het uitdenken van het ontwerp.

De basis van ASP.NET had ik snel onder de knie. De ASP.NET (C#) code wordt op dezelfde manier opgebouwd als Java code. Ik heb dan ook vrijwel zelfstandig, met behulp van documentatie, ASP.NET geleerd. Ik ben niet iemand die snel om hulp vraagt, ik probeer het liever eerst zelf. Pas als ik er écht niet uitkom, vraag ik om hulp. Zo is het nu ook gegaan. De meeste problemen die ik had, heb ik zelfstandig opgelost. Bij enkele dingen, waar ik echt niet snel uitkwam, heb ik hulp gevraagd.

Bij het bedrijf is zeer veel expertise aanwezig met betrekking tot ASP.NET. Zij konden mij dan ook altijd helpen en soms zelfs handige hulpmiddelen geven, zoals bijvoorbeeld het ETX Database component.
Al met al is ASP.NET me zeer goed bevallen en zou ik het in de toekomst zeker nogmaals gebruiken.

9.1.6 Testen

Het testplan voor de eerste pilot had ik opgesteld nadat alle functionaliteit al gerealiseerd was. Hierdoor was ik geneigd om het testplan op te stellen aan de hand van de gerealiseerde applicatie. Het testplan zou eigenlijk opgesteld moeten worden aan de hand van de use-cases, zodat er getest kan worden of de applicatie voldoet aan wat was afgesproken in het systeemconcept.
Om dit voor de tweede pilot beter te doen, heb ik dit anders aangepakt. Nadat was afgesproken, in het pilotplan, welke functionaliteit ik ging realiseren in de tweede pilot, ben ik het testplan op gaan stellen. Dit is heel erg goed bevallen. Het testplan was nu helemaal gebaseerd op de use-cases, zonder mogelijke invloed van hoe het werkelijk was geworden.

9.2 Product

De product evaluatie evalueert het plan van aanpak en de producten die ik heb opgeleverd aan de opdrachtgever.

9.2.1 Plan van aanpak

Het plan van aanpak diende als basis voor de opdracht. Het had altijd de actuele versie van de planning en werd wekelijks gebruikt bij de gesprekken met Karel van der Lelij.
In het plan van aanpak stonden de afspraken met betrekking tot de opdracht, zoals dat ik communiceerde met Michiel Bruins over de opdracht en IAD gebruikt zou worden.
Naar mijn idee voldeed het plan van aanpak voor deze opdracht. Doordat het altijd de actuele versie van de planning had, gebruikte ik het document regelmatig.

9.2.2 Definitie Studie

De systeemeisen zijn het eerste product van de definitie studie. Hierin stonden alle eisen aan het systeem, zoals de opdrachtgever deze had verteld. De eisen stonden gecategoriseerd naar onderwerp. Omdat het niet makkelijk te zien was of alle eisen in het document aanwezig waren, heb ik deze opgesomd in een tabel en de prioriteit bij elke eis gezet. Deze tabel was erg handig bij het bespreken van de eisen met de opdrachtgever. Zonder deze tabel was het document niet makkelijk doorzoekbaar.

In de systeemeisen heb ik zo veel mogelijk proberen te vermijden om al rekening te houden met de technische implementatie. Doordat niet alle functionaliteit gerealiseerd kon worden zijn er wel compromissen gesloten welke meer gericht waren op de technische implementatie. Zo staat er in de systeemeisen beschreven dat een beheerderinterface niet perse nodig is, maar dat er dan wel aanpassingen gemaakt moeten kunnen worden via het datamodel.

Aan de hand van de systeemeisen is het systeemconcept opgesteld. Daarin is de functionaliteit uitgewerkt in use-cases en een use-case diagram. Het gevaar is dat een opdrachtgever de use-cases niet begrijpt en daardoor geen voorstelling kan krijgen van het systeem.
De opdrachtgever was echter bekend met use-cases en had daarom geen problemen met het lezen of begrijpen van het systeemconcept. Wanneer dit niet het geval zou zijn geweest en de opdrachtgever geen ervaring had met use-cases, zou ik meer grafische elementen in het systeemconcept hebben gebruikt om de functionaliteit van het systeem aan te geven.

9.2.3 Onderzoek maatwerk / bestaand pakket

Het onderzoek naar het inzetten van maatwerk of een bestaand was niet heel uitgebreid, maar beantwoorde wel een belangrijke vraag. Het onderzoek is opgesteld aan de hand van enkele criteria. Bij de beoordeling van elk criteria heeft de voorkeur van de opdrachtgever een grote rol gespeeld. De voorkeur lag duidelijk bij maatwerk. Uit het onderzoek is gebleken, dat met de eisen die de opdrachtgever had, maatwerk de juiste keuze was. Het is een goede keuze geweest om dit onderzoek uit te voeren.

9.2.4 Pilotontwikkelplan

In het pilotplan heb ik beschreven in welke volgorde ik de verschillende pilotdelen ging ontwikkelen. Het pilotplan werd ondersteund door de functionele en technische structuur. Het document was vooral een hulpmiddel om gestructureerd te blijven werken en niet zomaar te gaan beginnen. Door het opstellen van het pilotplan heb ik nagedacht waarom ik een bepaalde volgorde aanhield. Tijdens het ontwikkelen hielp het document om deze volgorde ook echt te volgen.

9.2.5 TrackMonkey.NET

De opgeleverde webpart binnen de Sharepoint portal is het belangrijkste resultaat voor de opdrachtgever. De functionaliteit die gerealiseerd is, werkt goed. De applicatie is generiek opgezet en makkelijk aanpasbaar en uitbreidbaar. Klanten kunnen verschillende soorten meldingen hebben en ook de workflow die meldingen volgen is aanpasbaar.
De doelstellingen die aan het begin van mijn afstuderen zijn opgesteld, zijn behaald.

Bepaalde keuzes zou ik achteraf anders hebben genomen. Zo heb ik er voor gekozen om sommige velden van een melding niet te koppelen aan een soort. Daardoor is dat veld bij elke melding, van elk soort, beschikbaar. Dit leek heel handig tijdens de eerste pilot, maar achteraf gezien beperkt het de mogelijkheden. Nu is het niet mogelijk om een veld te koppelen aan meerdere soorten, maar niet alle.

De webpart werkt goed samen met Sharepoint. Tijdens de eerste pilot vond ik het vervelend dat de webpart niet wilde samenwerken met het ETX Database component. Gelukkig is dit opgelost in pilot twee. Hierdoor is de applicatie netjes verdeeld in drie lagen.
Door de code-review van een medewerker van Qurius ETX is de kwaliteit van de code gecontroleerd. Aan de hand van deze review heb ik aanpassingen gedaan waardoor ik zeer tevreden ben over de kwaliteit van de opgeleverde code.

9.2.6 Testplan

Het testplan is opgesteld per use-case. Het testplan voor de eerste pilot kon gemakkelijk doorlopen worden. Daar had je nog niet te maken met verschillende rollen.
Het testplan voor de tweede pilot was minder makkelijk te doorlopen. Er moest namelijk steeds van rol worden gewisseld om dezelfde functionaliteit te testen met meerdere rollen. Dit is niet handig en kost veel tijd. De volgende keer dat ik een testplan zal opstellen, zal ik er op letten dat het testplan gemakkelijk te doorlopen is voor één rol en er maar één keer gewisseld hoeft te worden van rol.

9.2.7 Functionele en Technische Structuur

In de functionele en technische structuur is de werking van de applicatie en het ontwerp van het datamodel beschreven. Om de werking van de applicatie te beschrijven, heb ik gebruik gemaakt van sequentiediagrammen.
Voordat ik een sequentiediagram ging maken, werkte ik de functionaliteit eerst uit in een stappenplan. Voor sommige use-cases was dit stappenplan voldoende en had ik het sequentiediagram niet nodig om het ontwerp te kunnen implementeren. Toch heb ik van elk stappenplan een sequentiediagram gemaakt, omdat dit de systeem documentatie volledig maakt. Als er later iemand anders aan het systeem moet werken, zullen de sequentiediagrammen de werking van het systeem zeker verduidelijken. Een stappenplan zou dit veel minder snel doen.

In de technische structuur heb ik het datamodel bescgreven. Naast een visuele weergave heb ik ook in tekst alle klassen en relaties beschreven. Na het lezen van deze tekst zou het datamodel duidelijk moeten zijn, zodat daar eventueel aanpassingen aan gemaakt kunnen worden.

9.2.8 Installatiehandleiding

De applicatie zou niet compleet zijn geweest zonder installatiehandleiding. Hierin wordt duidelijk omschreven hoe de smartpart geïnstalleerd kan worden en welke wijzigingen er eventueel nodig zijn op Sharepoint.
Ook de setup van de database komt ter sprake. De handleiding is naar mijn idee duidelijk en in een logische volgorde opgebouwd. Door middel van simpele stappenplannen kan de applicatie geïnstalleerd worden.

9.2.9 Beheerhandleiding

De beheerhandleiding bevat de meest voorkomende beheer acties die niet in de applicatie zelf uitgevoerd kunnen worden. Deze handleiding was essentieel, omdat de opdrachtgever anders geen idee had hoe hij nieuwe klanten kon aanmaken en andere data kon aanpassen.
De handleiding is duidelijk opgebouwd en behandeld per activiteit de te ondernemen stappen. Naast het stappenplan wordt ook elke keer de optie van een stored procedure gegeven, die de stappen automatisch uitvoert.
Doordat er voor elke beheeractie een stored procedure beschikbaar is, is het gemakkelijk om in de toekomst een user interface bij het beheergedeelte te ontwikkelen.

9.2.10Gebruikershandleiding

De gebruikershandleiding lijkt redelijk op het systeemconcept. Er wordt per activiteit beschreven wat de mogelijke handelingen zijn. Het systeemconcept is echter niet geschikt voor normale gebruikers. De gebruikershandleiding behandelt de mogelijke acties aan de hand van screenshots. Er wordt duidelijk verteld wat de mogelijkheden zijn op een pagina en welke acties er ondernomen kunnen worden.

Het systeem is al heel erg duidelijk. Ik denk niet dat veel gebruikers deze handleiding nodig zullen hebben. Toch heb ik de handleiding gemaakt, omdat het mijn afstudeerproject volledig maakt. Bij het opleveren van software hoort nou eenmaal ook een gebruikershandleiding.
Aan het begin van dit project heb ik ook afgesproken dat deze handleiding zou worden gemaakt. Voor de lezers van mijn eindverslag kan de handleiding tevens duidelijkheid bieden. Doordat er veel screenshots in staan, kan een goed beeld worden verkregen van de opgeleverde applicatie.

10 Begrippenlijst

De begrippen die nodig zijn om een ander begrip te begrijpen, worden voor dat begrip genoemd.

- ASP.NET
  Framework van Microsoft waarbinnen verschillende programmeertalen beschikbaar zijn, zoals C#.

- Workflow
  Alle mogelijke routes die een melding kan volgen.

- Stap in de workflow
  Een keuze in de workflow waardoor er een bepaalde route wordt genomen.

- Veld
  Een veld hoort bij een melding. Een veld kan bijvoorbeeld “naam” of “omschrijving” of “prijs” zijn.

- Waarde
  Bij een melding horen verschillende waarden, zoals bijvoorbeeld “300”. Een waarde hoort bij een veld, zo kan deze waarde bijvoorbeeld bij het veld “prijs” horen.

- Sharepoint
  Op een Sharepoint Server kunnen Sharepoint Portals draaien. Een Sharepoint Portal is een website waar alle gegevens van een project geordend bij elkaar staan.

- Webpart
  Een webpart is een webapplicatie die kan draaien binnen een Sharepoint pagina.

- Web-user-control
  Een herbruikbaar onderdeel van een ASP.NET pagina. Bijvoorbeeld een deel van een pagina dat terugkomt op meerdere pagina’s. Een web-user-control kan tekst/plaatjes/tabellen etc bevatten. Het kan ook het enige onderdeel op een ASP.NET pagina zijn.

- Smartpart
  Een smartpart is een webpart waar een web-user-control ingeladen kan worden.

11 Literatuurlijst

(The rational guide to) building Sharepoint web parts
Darrin Bishop
ISBN 0 9726888 6 2

Memo RFC Procedure
Michiel Bruins

Database Systems
Thomas Connolyy en Carolyn Begg
ISBN 0 201 70857 4

De UML Toolkit
Hans-Erik Eriksson en Magnus Penker
ISBN 90 395 1015 6

ETX Database.NET Component
Robert van der Kleij

Software Engineering
Ian Sommerville
ISBN 0 201 42765 6

IAD
R.J.H. Tolido
ISBN 90 395 0401 6

Developing web applications
Jeff Webb with Microsoft Corporation
MCAD/MCSD Self-Paced Training Kit

Afstudeerverslag: Incident Management Solution
Idde Zijlstra

Timebox Rapport – TrackMonkey.NET
Idde Zijlstra

Websites:
Smartpart:

- http://www.gotdotnet.com/workspaces/workspace.aspx?id=6cfaabc8-db4d-41c3-8a88-3f974a7d0abe
- http://weblogs.asp.net/jan/archive/2004/07/14/183331.aspx
- http://www.microsoft.com/belux/nl/msdn/community/columns/u2u/smartpart.mspx#w08-2005
- http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon/html/vbwlkwalkthroughcreatingwebusercontrols.asp

Sharepoint:

- http://support.microsoft.com/default.aspx?scid=kb;en-us;820774
- http://support.microsoft.com/?id=828810

C#:

- http://aspalliance.com/625
- http://www.microsoft.com/belux/nl/msdn/community/columns/hyatt/ntier1.mspx

Interne Bijlagen

Afstudeerverslag Michiel Post

– Het realiseren van een applicatie om meldingen over fouten en wijzigingsverzoeken gestructureerd af te handelen. –

M.R. Post
20011937
IVIT

Qurius ETX
Lange Kleiweg 56b
2288 GK Rijswijk

Qurius ETX

Inhoudsopgave

1. Plan van aanpak

1 Inleiding

Naar aanleiding van de opleiding Informatica aan de HHS, wordt er in het 4e jaar een afstudeeropdracht uitgevoerd.
De opdracht die deze periode uitgevoerd zal worden, is het ontwerpen en ontwikkelen van een applicatie om gestructureerd foutmeldingen wijzigingsverzoeken af te handelen. Deze afstudeeropdracht wordt gedurende 18 weken uitgevoerd bij Qurius ETX. Dit plan van aanpak heeft betrekking op de gehele afstudeeropdracht.

Er zal eerst een korte beschrijving worden gegeven van Qurius ETX. Vervolgens wordt het probleem en de opdracht besproken. Hierna worden de activiteiten besproken die uitgevoerd zullen worden om tot de eindproducten te komen.
Een bespreking van de projectinrichting en de kwaliteitsborging wordt besproken in hoofdstuk 5 en 6.

2 Qurius ETX

Qurius ETX voert softwareontwikkelingsprojecten uit op het Microsoft platform. Zij hebben een platform ontwikkeld (SoftwareStraat.Net) op basis waarvan zij al hun software ontwikkelen. Ze richten zich met name op de integratie van informatiesystemen, portals en andere e-commerce applicaties.

Er werken bij Qurius ETX 31 mensen in verschillende projectgroepen. Er wordt veel aandacht besteed aan je persoonlijke ontwikkeling binnen het bedrijf. Afstudeerders worden beschouwd als gewone werknemers. Ze werken echter aan een interne afstudeeropdracht en niet aan commerciële projecten. Vanuit het bedrijf zijn er twee begeleiders, één voor het proces en één voor de inhoudelijke opdracht.

3 Opdrachtomschrijving

In dit hoofdstuk is de aanleiding te vinden tot het probleem. Hierna staat de probleemstelling beschreven en welk doel bereikt dient te zijn aan het einde van deze afstudeerperiode.

3.1 Aanleiding

Qurius ETX voert projecten uit voor klanten. Alle gegevens van een project zijn centraal benaderbaar via een zogenaamde Sharepoint Portal. Via dit interne portal zijn ook verschillende applicaties beschikbaar. Het portal is beschikbaar voor zowel klanten als medewerkers.

Tijdens de loop van een project heeft de klant continu toegang tot de applicatie die ontwikkeld wordt, zo kan hij controleren of zijn eisen en wensen juist zijn geïnterpreteerd. Over afwijkingen en gewijzigde inzichten communiceert hij met een projectleider bij Qurius ETX.
Wanneer er een melding, over een fout of wijziging, van een klant of ontwikkelaar binnen komt, besluit de projectleider wat er mee moet gebeuren. Hij wijst een ontwikkelaar aan die deze melding moet afhandelen. Wanneer de ontwikkelaar klaar is met de afhandeling, wordt er een controle uitgevoerd. Na goedkeuring wordt de melding afgesloten.

3.2 Probleemstelling

Op dit moment is er onvoldoende mogelijkheid om meldingen, van klanten of ontwikkelaars, over fouten of wijzigingen vast te leggen. De opdrachtgever heeft geen inzicht in de status van de meldingen en kan klanten er ook geen inzicht in geven.

3.3 Doelstelling

De doelstelling van de opdracht is het ontwikkelen van de applicatie TrackMonkey. Deze applicatie zal klanten en ontwikkelaars de mogelijkheid geven om meldingen te doen over fouten of wijzigingsverzoeken in een door Qurius ETX ontwikkelde applicatie. De meldingen zullen worden geregistreerd en gestructureerd afgehandeld worden door het ontwikkelteam van deze applicatie. De workflow die gevolgd dient te worden moet aanpasbaar zijn.
Het is belangrijk dat de applicatie TrackMonkey klanten en ontwikkelaars inzicht kan geven in de status van een melding, zodat klanten weten wat er met hun melding gebeurd en ontwikkelaars snel een overzicht hebben van hun taken.
TrackMonkey moet beschikbaar zijn als ‘webpart’ binnen de Sharepoint Portal.

Van een foutmelding en wijzigingverzoek moeten verschillende gegevens worden opgeslagen. Er moet een rechtenstructuur aanwezig zijn, die regelt wie welke gegevens van een fout of wijzigingsverzoek mag zien, wie het mag aanpassen etc.

De belangrijkste succesfactoren van het project zijn:

- Klant moet zelf melding kunnen aanmaken via de portal
- Status en gegevens van meldingen moet opgevraagd kunnen worden
- Rechtenstructuur aanwezig
- Werkende workflow

3.4 Op te leveren producten

De volgende producten zullen worden opgeleverd aan de opdrachtgever:

- Plan van aanpak
- Definitiestudie
- Pilotontwikkelplan
- Systeemdocumentatie
- Installatie handleiding
- Gebruikers handleiding
- Webpart binnen de Sharepoint Portal

4 Activiteiten

Opstellen plan van aanpak

Definitiestudie:
Documentatie bestuderen
Voorbereiden pilotplan-workshop
Definiëren van systeemeisen
Opstellen Systeem Concept
Onderzoek maatwerk / bestaand systeem uitvoeren
Pilotplan opstellen

Pilotontwikkeling:
Voorbereiden pilotontwerp-workshop
Specificeren van globaal-functionele structuur pilot
Specificeren van globaal-technische structuur pilot
Pilotontwikkelplan opstellen
Ontwerp software bouweenheden
Bouw software bouweenheden
Opstellen handleidingen pilot
Opstellen invoeringsprocedures
Beoordelen en testen pilot

Invoering:
Invoeren pilot

Eindverslag schrijven

4.1 Methode

Voor dit project zal de methodiek IAD gebruikt worden. Qurius ETX heeft zijn eigen ontwikkelmethode ontwikkeld, genaamd Softwarestraat.Net. Er is gekozen om hier niet mee te werken, omdat ik het belangrijk vind genoeg ervaring te hebben met een ontwikkelmethode om daar goed gebruik van te kunnen maken.

Het project heeft de eigenschap dat het sterk tijdgebonden is. Tevens is veel interactie met de opdrachtgever vereist, zodat alle eisen en wensen in kaart gebracht kunnen worden.
Er zal tijdens dit project worden gewerkt met nieuwe technieken, zoals ASP.NET. De opdrachtgever heeft aangegeven het belangrijk te vinden om uiteindelijk een bruikbaar eindproduct te hebben en niet slechts een aantal probeersels. Het werken met nieuwe technieken brengt echter het gevaar met zich mee dat de gemaakte planning niet uitkomt. Door gebruik te maken van iteratief ontwikkelen zal dit risico worden beperkt.

4.1.1 Iteratiestrategie

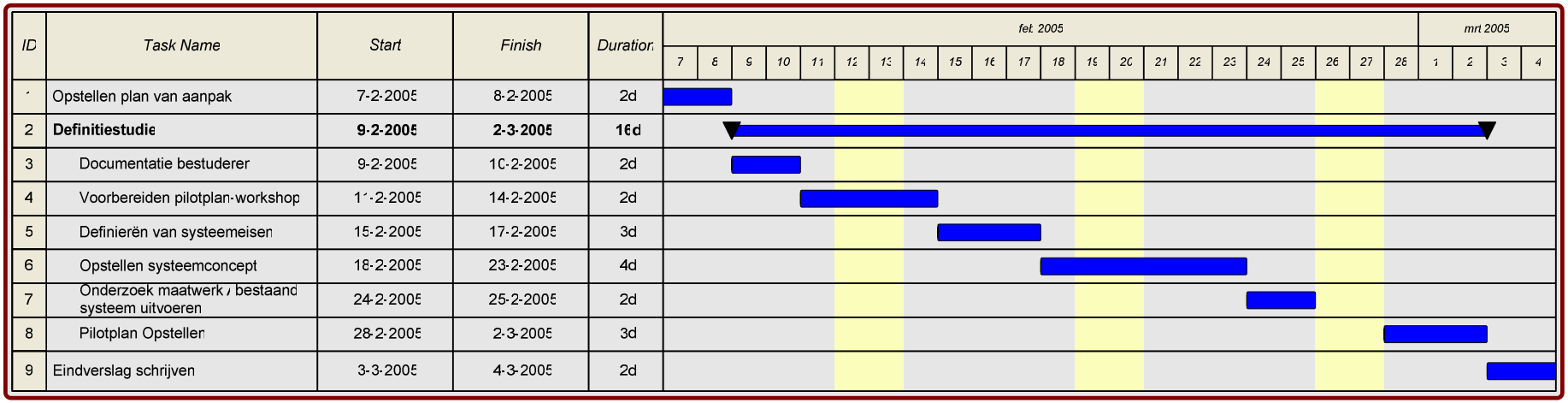
Er is gekozen voor evolutionair ontwikkelen. Het voordeel van deze strategie is dat in de definitie studie de eisen niet tot in alle detail uitgewerkt hoeven te worden. Er kan relatief snel aan de eerste pilot begonnen worden. Er zullen snel concrete resultaten geboekt worden en dat is wat de opdrachtgever heeft aangegeven graag te willen.
De opdrachtgever bepaalt uiteindelijk of de pilot ingevoerd wordt of dat er nog een pilot nodig is om de functionaliteit uit te breiden voor de pilot ingevoerd wordt.

4.2 Planning

In onderstaande Gantt Chart is de planning uitgewerkt.

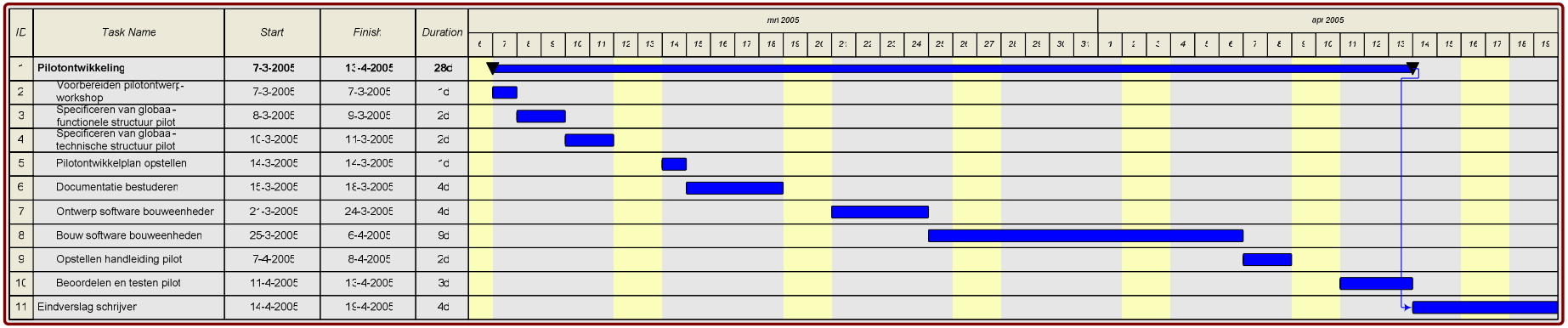
Definitiestudie 1:

ID	Task Name	Start	Finish	Duration
1	Opstellen plan van aanpak	7-2-2005	8-2-2005	2d
2	**Definitiestudie**	**9-2-2005**	**2-3-2005**	**16d**
3	Documentatie bestuderen	9-2-2005	10-2-2005	2d
4	Voorbereiden pilotplan-workshop	11-2-2005	14-2-2005	2d
5	Definiëren van systeemeisen	15-2-2005	17-2-2005	3d
6	Opstellen systeemconcept	18-2-2005	23-2-2005	4d
7	Onderzoek maatwerk / bestaand systeem uitvoeren	24-2-2005	25-2-2005	2d
8	Pilotplan Opstellen	28-2-2005	2-3-2005	3d
9	Eindverslag schrijven	3-3-2005	4-3-2005	2d



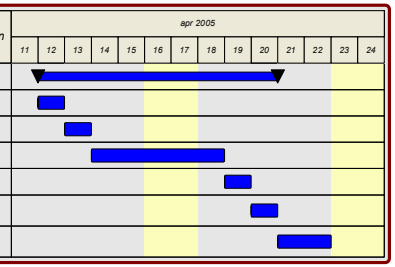
Pilot 1:

ID	Task Name	Start	Finish	Duration
1	**Pilotontwikkeling**	**7-3-2005**	**13-4-2005**	**28d**
2	Voorbereiden pilotontwerp-workshop	7-3-2005	7-3-2005	1d
3	Specificeren van globaa-functionele structuur pilot	8-3-2005	9-3-2005	2d
4	Specificeren van globaa-technische structuur pilot	10-3-2005	11-3-2005	2d
5	Pilotontwikkelplan opstellen	14-3-2005	14-3-2005	1d
6	Documentatie bestuderen	15-3-2005	18-3-2005	4d
7	Ontwerp software bouweenheder	21-3-2005	24-3-2005	4d
8	Bouw software bouweenheden	25-3-2005	6-4-2005	9d
9	Opstellen handleiding pilot	7-4-2005	8-4-2005	2d
10	Beoordelen en testen pilot	11-4-2005	13-4-2005	3d
11	Eindverslag schrijver	14-4-2005	19-4-2005	4d



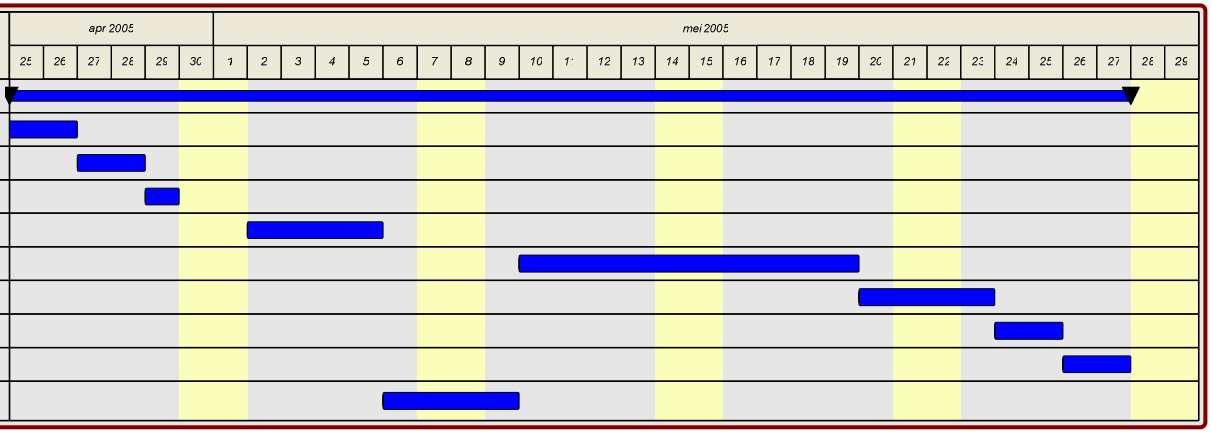
Definitiestudie 2:

ID	Task Name	Start	Finish	Duration
1	**Definitiestudie**	**12-4-2005**	**20-4-2005**	**7d**
2	Voorbereiden pilotplan-workshop	12-4-2005	12-4-2005	1d
3	Definiëren van systeemeisen	13-4-2005	13-4-2005	1d
4	Opstellen systeemconcept	14-4-2005	18-4-2005	3d
5	Pilotplan Opstellen	19-4-2005	19-4-2005	1d
6	Testplan opstellen	20-4-2005	20-4-2005	1d
7	Eindverslag schrijven	21-4-2005	22-4-2005	2d



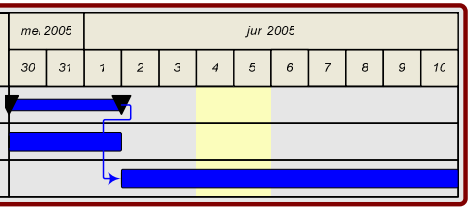
Pilot 2:

ID	Task Name	Start	Finish	Duration
1	**Pilotontwikkeling**	**25-4-2005**	**27-5-2005**	**25d**
2	Specificeren van globaa - functionele structuur pilot	25-4-2005	26-4-2005	2c
3	Specificeren van globaa - technische structuur pilot	27-4-2005	28-4-2005	2c
4	Pilotontwikkelplan opstellen	29-4-2005	29-4-2005	1c
5	Ontwerp software bouweenheder	2-5-2005	5-5-2005	4c
6	Bouw software bouweenheder	10-5-2005	19-5-2005	8c
7	Opstellen handleiding pilot	20-5-2005	23-5-2005	2c
8	Opstellen invoeringsprocedures	24-5-2005	25-5-2005	2c
9	Beoordelen en testen pilot	26-5-2005	27-5-2005	2c
10	Eindverslag	6-5-2005	9-5-2005	2c



Invoering:

ID	Task Name	Start	Finish	Duration
1	**Invoering**	**30-5-2005**	**1-6-2005**	**3d**
2	Invoering pilot	30-5-2005	1-6-2005	3c
3	Eindverslag schrijver	2-6-2005	10-6-2005	7c



5 Projectinrichting

In dit hoofdstuk worden een aantal factoren besproken die betrekking hebben op de projectinrichting.

5.1 Geld

Het aspect geld is niet van toepassing op dit project.

5.2 Organisatie

Binnen Qurius ETX communiceer ik met Karel van der Lelij over de procesgang. Er is afgesproken om wekelijks de voortgang te bespreken.
Michiel Bruins functioneert als opdrachtgever, met hem communiceer ik over de inhoudelijke zaken van de opdracht. Gemiddeld genomen zal ik ongeveer één uur per week nodig hebben van zijn tijd.

De opdracht voer ik zelfstandig uit.

5.3 Tijd

Er zijn 18 weken beschikbaar voor het uitvoeren van de afstudeeropdracht. Hiervan zijn 3 weken gereserveerd voor het maken van het eindverslag. De eerste dag van het project is 7 februari 2005 en het afstuderen zal eindigen op 10 juni 2005.

5.4 Informatie

Binnen de organisatie zijn documenten beschikbaar over het proces van afhandelen van foutmeldingen en wijzigingsverzoeken. Ontbrekende informatie zal verkregen worden via interviews en workshops.

5.5 Communicatie

De communicatie zal vooral persoonlijk contact zijn. Ik zit op dezelfde kamer als Michiel Bruins, wat communiceren makkelijk maakt. Er is tevens de beschikking over e-mail en telefoon.

5.6 Risico

Een risico van dit project is dat er niet op tijd genoeg kennis zal worden verkregen van de programmeertaal ASP.NET om het volledige eindproducten te kunnen realiseren. Er zal daarom ook onderzoek gedaan worden naar de mogelijkheid om bestaande workflow pakketten te integreren op de Sharepoint portal. Aan het einde van de definitie studie zal er een keuze worden gemaakt of er voor maatwerk of voor een bestaand systeem zal worden gegaan.
Wanneer er gekozen zal worden om de webpart volledig zelf te gaan ontwikkelen is er nog steeds het risico dat er niet op tijd genoeg kennis zal worden verkregen om het volledige eindproduct te kunnen realiseren. Er zal daarom gewerkt worden in iteraties, zodat er aan het einde altijd een werkend deel van het eindproduct opgeleverd kan worden.

6 Kwaliteitsborging

Dit hoofdstuk beschrijft de kwaliteitsborging van zowel het proces als het product.

6.1 Proces

Het gebruiken van een methode helpt de kwaliteit te waarborgen. Als methode voor dit project wordt IAD gebruikt. Er zal wekelijks overleg zijn met Karel van der Lelij over de procesgang. Hij zal de kwaliteit van het proces in de gaten houden.

6.2 Producten

Van het op te leveren systeem zullen ontwerpen worden gemaakt met behulp van UML. De kwaliteit van de tussenproducten zal worden gewaarborgd door mijzelf Michiel Bruins. Michiel Bruins is verantwoordelijk voor de inhoudelijke begeleiding.

Kwaliteitseisen aan het eindproduct zijn:

- De opgeleverde code is gedocumenteerd
- De workflow dient beschreven te zijn
- Er wordt eenduidige naamgeving gebruikt
- Er wordt object georiënteerd gewerkt

2. Opdrachtomschrijving

Inleiding (organisatorische omgeving, kader, historie):
Qurius ETX is specialist in gegarandeerde, professionele softwareontwikkelingsprojecten op het Microsoft platform. Zij hebben een platform ontwikkeld (SoftwareStraat.Net) op basis waarvan zij al hun software ontwikkelen. Qurius ETX richt zich met name op de integratie van informatiesystemen, portals en andere e-commerce applicaties

Alle gegevens van een project zijn centraal benaderbaar via een zogenaamde Sharepoint Portal. Via deze portal zijn ook verschillende applicaties benaderbaar. De portal is beschikbaar voor zowel klanten als medewerkers.

Tijdens de loop van een project heeft de klant continu toegang tot de applicatie die ontwikkeld wordt, zo kan hij controleren of zijn eisen en wensen juist zijn geïnterpreteerd. Over afwijkingen en gewijzigde inzichten communiceert hij met een projectleider bij Qurius ETX.
Wanneer er een melding, over een fout of wijziging, van een klant of ontwikkelaar binnen komt, besluit de projectleider wat er mee moet gebeuren. Hij wijst een ontwikkelaar aan die deze melding moet afhandelen. Wanneer de ontwikkelaar klaar is met de afhandeling, wordt er een controle uitgevoerd. Na goedkeuring wordt de melding afgesloten.

Probleemstelling:
Op dit moment is er onvoldoende mogelijkheid om meldingen, van klanten of ontwikkelaars, over fouten of wijzigingen vast te leggen. De opdrachtgever heeft geen inzicht in de status van de meldingen en kan klanten er ook geen inzicht in geven.

Doelstelling van de opdracht:
De doelstelling van de opdracht is het ontwikkelen van de applicatie TrackMonkey. Deze applicatie zal klanten en ontwikkelaars de mogelijkheid geven om meldingen te doen over fouten of wijzigingsverzoeken in een door Qurius ETX ontwikkelde applicatie. Deze meldingen zullen worden geregistreerd en gestructureerd afgehandeld worden door het ontwikkelteam van deze applicatie. De workflow die gevolgd dient te worden moet aanpasbaar zijn.
Het is belangrijk dat de applicatie TrackMonkey klanten en ontwikkelaars inzicht kan geven in de status van een melding, zodat klanten weten wat er met hun melding gebeurd en ontwikkelaars snel een overzicht hebben van hun taken.
TrackMonkey moet beschikbaar zijn als ‘webpart’ binnen de Sharepoint Portal.

Uitgangssituatie:

Benodigde software
- Microsoft Windows 2003 Server
- Microsoft Office
- Microsoft Internet Explorer
- Microsoft Visual Studio .NET
- Microsoft SQL Server
- Microsoft Sharepoint Server
- Rational Rose
- TrackRecord
- Enterprise Architect

Benodigde hardware
- Desktop Computer
- Internetverbinding

Beschikbare rapporten

Aanwezige documenten vorige afstudeerder
Sharepoint Portal documentatie
ASP.NET documentatie
Documentatie over de verschillende procedures

Aanwezige ideeën

Concrete werkzaamheden:

Uit te voeren activiteiten:

Opstellen plan van aanpak

Definitiestudie:
Documentatie bestuderen
Voorbereiden pilotplan-workshop
Definieren van systeemeisen
Opstellen Systeem Concept
Onderzoek maatwerk / bestaand systeem uitvoeren
Pilotplan opstellen

Pilotontwikkeling:
Voorbereiden pilotontwerp-workshop
Specificeren van globaal-functionele structuur pilot
Specificeren van globaal-technische structuur pilot
Pilotontwikkelplan opstellen
Ontwerp software bouweenheden
Bouw software bouweenheden
Opstellen handleidingen pilot
Opstellen invoeringsprocedures
Beoordelen en testen pilot

Invoering:
Invoeren pilot

Te hanteren methodieken
IAD

Te gebruiken technieken
Interviews
UML
Workshops
Timeboxing

Te vermelden nadrukken

Eventueel aanvullende opmerkingen:
De webpart zal komen te draaien op de Sharepoint Server.

Resultaten voor de opdrachtgever (op te leveren producten):
Plan van aanpak
Definitiestudie
Pilotontwikkelplan
Systeemdocumentatie
Installatie handleiding
Gebruikers handleiding
Webpart binnen de Sharepoint Portal

Externe Bijlagen

Afstudeerverslag Michiel Post

– Het realiseren van een applicatie om meldingen over fouten en wijzigingsverzoeken gestructureerd af te handelen. –

M.R. Post
20011937
IVIT

Qurius ETX
Lange Kleiweg 56b
2288 GK Rijswijk

Qurius ETX

Inhoudsopgave

1. Systeemeisen

1 Inleiding

In dit document staat een analyse van de eisen en wensen aan het op te leveren systeem. Deze analyse is opgesteld aan de hand van een aantal gesprekken over de opdracht en een interne memo over de RFC Procedure.
De eisen en wensen naar aanleiding van de eerste pilot, zijn ook in dit document verwerkt.

Dit document dient om overeenstemming te krijgen over de eisen en als basis voor het opstellen van het systeemconcept.

2 Systeemeisen

In dit hoofdstuk staan alle eisen aan het systeem.

2.1 Platform: Sharepoint portal

Het op te leveren systeem zal draaien als Webpart binnen de Sharepoint portal.

Een Sharepoint portal is een website waar verschillende applicaties benaderbaar zijn. Een webpart is een applicatie die binnen de sharepoint portal weergegeven kan worden. Gebruikers die inloggen op de sharepoint portal hebben een persoonlijk overzicht van de door hun ingestelde applicaties.

2.2 Sharepoint look and feel

De toekomstige applicatie moet zoveel mogelijk de look and feel van Sharepoint overnemen. Nieuwe pagina's worden geopend in hetzelfde venster en er moet de mogelijkheid zijn om terug naar de vorige pagina te gaan.
In Sharepoint is het gebruikelijk dat het mogelijk is om de header van een webpart aan te klikken om de webpart in een volledig venster te zien. Het zou handig zijn als de te ontwikkelen webpart dit ook kan.

2.3 Foutmeldingen en Wijzigingsverzoeken Invoeren

In het systeem moeten zowel foutmeldingen als wijzigingsverzoeken kunnen worden ingevoerd. Het invoerscherm zal echter altijd dat van een foutmelding zijn. Een melding komt dus altijd als foutmelding in het systeem binnen, later kan er beslist worden dat de melding een wijzigingsverzoek is.

2.4 Andere gegevens opslaan foutmelding / wijzigingsverzoek

Van een foutmelding en wijzigingsverzoek dienen andere gegevens opgeslagen te worden:

2.4.1 Foutmelding

Bij het invoeren van een foutmelding moeten minstens de volgende gegevens worden opgeslagen:

- Nummer
- Naam
- Omschrijving
- Datum van Aanmelding
- Naam van Aanmelder

In de loop van het proces moet het mogelijk zijn de volgende gegevens te kunnen opslaan over een foutmelding:

- Toegewezen aan
- Categorie
- Verwante foutmelding
- Prioriteit (Hoog / Middel / Laag)
- Opmerkingen
- Project

2.4.2 Wijzigingsverzoek

Mocht de foutmelding een wijzigingsverzoek blijken, dan moet de volgende informatie kunnen worden opgeslagen:

- Oplossingsrichting, tekstveld
- SE Uren
- TE Uren
- PL Uren
- Prijs
- Commentaarveld voor de prijs
- Toegewezen aan
- Categorie
- Verwante foutmelding
- Prioriteit (Hoog / Middel / Laag)
- Opmerkingen

2.5 Soort wisselen

Meldingen worden altijd ingevoerd als foutmeldingen. Het zal dus mogelijk moeten zijn om meldingen een wijzigingsverzoek te laten worden en andersom.

2.6 Gegevens melding opvragen

Na invoer van een melding moet het mogelijk zijn alle gegevens van een melding op te vragen. Dit mag enkel als je daar de juiste rechten voor hebt.

2.7 Gegevens melding wijzigen

Tijdens het gehele workflow proces zijn de gegevens die over een melding opgeslagen worden aan te passen. Dit kan alleen als je daar de rechten voor hebt.

2.8 Controle op invoer bij nieuwe melding of wijziging

Als de gegevens gewijzigd zijn, moet er een controle worden uitgevoerd of de gegevens voldoen aan het juiste formaat. Zo mag een prijs bijvoorbeeld enkel getallen bevatten, en de naam en de omschrijving van een melding mogen nooit leeg zijn.

2.9 Automatisch invullen velden

Wanneer het kan, dienen velden automatisch te worden aangevuld. Bijvoorbeeld bij het aanmelden van een melding is er bij het systeem bekend wie dit aanmeldt en wanneer, dit dient dan automatisch opgeslagen te worden.

2.10 URL / Bijlage toevoegen bij een melding

Het zou handig zijn als er een URL of een bijlage bij een melding toegevoegd kan worden. Zo kan een klant bijvoorbeeld een screenshot meezenden.

2.11 Mogelijkheid om van velden een combo-box te maken

Sommige velden van een melding, zoals bijvoorbeeld de prioriteit, hebben slechts een beperkt aantal keuze mogelijkheden. Het zou handig zijn als dit gepresenteerd kan worden als een combobox waarin slechts één keuze mogelijk is.
Op die manier is het niet mogelijk de prioriteit te veranderen in zomaar iets.

2.12 Meldingen verwijderen

Meldingen die ingevoerd zijn moeten verwijderd kunnen worden.

2.13 Rollen toekennen aan gebruikers

Verschillende gebruikers moeten verschillende rechten kunnen hebben. De volgende rollen zijn te onderscheiden:

1. Default
2. Projectleider
3. Ontwikkelaar
4. Medewerker Klant
5. Projectleider Klant

De gebruiker is bekend bij de Sharepoint Portal. Er hoeft geen aparte inlogprocedure te komen voor deze webpart. Er moet echter wel een koppeling komen tussen de rol en de individuele gebruiker.

Om een gebruiker toegang te verlenen tot de Sharepoint portal heeft de beheerder een toevoegscherm (van Sharepoint). Het zou mooi zijn om dan meteen te kunnen aangeven wat de rol is van die gebruiker in het workflow systeem. Dit een nice to have en heeft geen hoge prioriteit.
Het is wel een eis dat gebruikers aan rollen gekoppeld kunnen worden, maar dit mag ook via een eigen scherm in de webpart.

2.13.1 Default

Dit is de standaard rol. Deze rol heb je als je nog geen rol toegewezen hebt gekregen, maar wel toegang hebt tot de portal.

2.13.2 Projectleider

De projectleider mag alles.

2.13.3 Ontwikkelaar

Een ontwikkelaar mag acties uitvoeren zoals het behandelen van wijzigingsverzoeken en oplossen van foutmeldingen. Er mogen ook overzichten worden opgevraagd.

2.13.4 Medewerker Klant

Een medewerker klant mag foutmeldingen aanmelden en overzichten opvragen.

2.13.5 Projectleider Klant

Een projectleider klant mag zowel foutmeldingen aanmelden als wijzigingsverzoeken goedkeuren. Er mogen ook overzichten worden opgevraagd.
Een eis is wel, dat wanneer een projectleider klant een wijzigingsverzoek goedkeurt, er een melding komt bij de projectleider.

2.14 Veldrechten aanpassen GUI / datamodel

Het moet mogelijk zijn dat een bepaalde rol wel recht heeft om gegevens te bekijken maar niet om ze te wijzigen.
Hiervoor zullen er 3 soorten rechten zijn op een veld:

- Niet zichtbaar
- Zichtbaar maar niet wijzigbaar
- Zichtbaar en wijzigbaar

Deze rechten gelden per veld, zoals naam en prijs. Het is dus mogelijk dat een gebruiker wel de naam van een melding mag zien maar niet de prijs.

2.15 Melding status wijzigen

Een melding doorloopt verschillende statussen. In elke status zijn er één of meerdere rollen die de workflow status mogen wijzigen.
Een foutmelding en een wijzigingsverzoek hebben een verschillende workflow. De nieuwe status van een melding mag alleen een volgende stap in de workflow zijn.

De workflow zoals te zien is in bijlage 1 van de systeemeisen moet mogelijk zijn om te doorlopen.

2.16 Historie van stappen in de workflow opslaan

Er moet opgeslagen worden, wie wanneer welke stap in de workflow maakt. Alle stappen in de workflow moeten bijgehouden worden. Dit moet zichtbaar zijn bij het bekijken van de gegevens van een melding.

2.17 Aanpasbare workflow GUI / datamodel

De workflow houdt in, dat een melding steeds een status heeft plus een aantal mogelijke acties om naar een andere status te gaan. Om de workflow te kunnen doorlopen is een eis dat de status van een melding is aan te passen, mits je de juiste rol in het systeem hebt. De nieuwe status van een melding mag alleen een volgende stap in de workflow zijn.
De workflow zoals te zien is in bijlage 1 moet mogelijk zijn om te doorlopen.

De workflow moet aangepast kunnen worden, per klant. Een user interface om deze wijzigingen te kunnen doen, is een nice to have maar heeft geen hoge prioriteit, het is voldoende als dit via de database kan.

2.18 Overzichten Opvragen

Alle rollen moeten de volgende overzichten kunnen bekijken. Per rol verschilt het welke velden zichtbaar zijn op het overzicht. Niet elke rol heeft namelijk recht om een veld te mogen zien.

- Alle foutmeldingen
- Alle wijzigingsverzoeken
- Alle foutmeldingen en wijzigingsverzoeken

Een eis is dat overzichten geprint moeten kunnen worden.
Het moet mogelijk zijn de overzichten te sorteren per kolom. Het zou daarbij handig zijn als de resultaten verdeeld worden over meerdere pagina's.

2.19 Kolommen in overzicht kunnen aanpassen

In de getoonde overzichten moet aangegeven kunnen worden door de beheerder welke kolommen getoond worden.
De volgende opties moeten mogelijk zijn:

- Alle kolommen waarvan aangegeven is dat deze zichtbaar moeten zijn.
- Alle kolommen van één bepaald soort waarvan aangegeven is dat deze zichtbaar moeten zijn.
- De basis kolommen id/datum aanmelding/naam/omschrijving

2.20 Aanpasbaar overzicht per klant

Per klant moeten andere overzichten zichtbaar zijn. Er moet dan per klant geselecteerd worden welke kolommen zichtbaar zijn in het overzicht.

2.21 Sorteren

Het overzicht met alle meldingen moet gesorteerd kunnen worden, per kolom.

2.22 Overzichten printen

Het moet mogelijk zijn om de overzichten te printen, zonder last te hebben van de rest van de pagina waar de webpart op staat. Enkel het overzicht mag op papier komen.

2.23 Configureerbaar webpart

Het webpart wordt geïnstalleerd per klant. Er moet dus aangegeven kunnen worden voor welke klant dit webpart meldingen laat zien.

Bij het weergeven van de webpart op een portal moet aangegeven kunnen worden dat het getoonde overzicht alleen de wijzigingsverzoeken of alleen de foutmeldingen laat zien.

2.24 Zoeken

Alle gebruikers moeten kunnen zoeken in de meldingen, er moet gezocht kunnen worden op de volgende velden:

- Nummer
- Naam
- Omschrijving
- Toegekend Aan
- Datum

De eerste 3 hebben een hoge prioriteit. Het zoeken op de laatste 2 velden heeft een minder hoge prioriteit.

2.25 Extra velden toevoegen GUI / datamodel

Het zou prettig zijn als een projectleider een extra veld kan toevoegen bij de gegevens die worden opgeslagen over een foutmelding of wijzigingsverzoek via de applicatie. Er dient wel rekening mee gehouden te worden, dat van het nieuwe veld ook bepaalt moet worden voor wie en wanneer het zichtbaar is.

Het databasemodel moet aan de eis voldoen dat het mogelijk is een extra veld toe te voegen. Dit veld hoeft echter geen logica te bevatten. Er moet minstens een extra tekstveld kunnen worden toegevoegd. Het is een nice to have om een datatype te kunnen opgeven bij het extra veld.

2.26 Wijzigingsgeschiedenis bijhouden van gegevens

Wanneer er wijzigingen worden gedaan aan een aantal velden moet de oorspronkelijke waarde van dat veld ook bewaard blijven. Bij elke wijziging moet er een kopie van de oude waarde worden opgeslagen.

3 Eisen en wensen n.a.v. pilot 1

De volgende eisen en wensen kwamen naar voren naar aanleiding van de beoordeling en testworkshop bij pilot 1.

1. Terugknoppen bij het invoeren van een nieuwe melding en bekijken van de gegevens van een melding (volgens Sharepoint layout).
2. Duidelijkere mogelijkheid om naar de volgende pagina met meldingen te gaan (bij het overzicht van alle meldingen). < en > zijn te onduidelijk.
3. Mogelijkheid om header van webpart aan te klikken en dan de overzichtlijst in een gehele nieuwe pagina te krijgen.
4. Optie om enkel de basiskolommen (nummer / datum_aanmelding / naam / omschrijving) van een melding te tonen.
5. Zoeken tussen 2 datums.
6. De velden naam / omschrijving verplicht maken bij het wijzigen.
7. Controle op dat de prijs alleen een getal mag zijn.

Hierbij werd aangegeven dat vooral eis 1, 4 en 5 belangrijk waren.
De eisen en wensen zijn verwerkt in dit document.

Daarnaast waren er nog enkele eisen met een lage prioriteit:

- Het overzicht van meldingen instellen per klant
- Mogelijkheid om een veld een combobox te laten zijn
- Een mogelijkheid om bij een melding een URL mee te geven, zodat bijlagen gelinkt kunnen worden

4 Prioriteit

In onderstaand diagram worden de eisen kort weergegeven met hun prioriteit. De kolom referentie verwijst naar een paragraaf in dit document.

Functionele eisen:

	Prioriteit	Naam	Systeem Concept
1	Hoog	Foutmeldingen en wijzigingsverzoeken invoeren	Ja
2	Hoog	Soort wisselen	Ja
3	Hoog	Rollen toekennen aan gebruikers	Ja
4	Hoog	Gegevens melding opvragen	Ja
5	Hoog	Gegevens melding wijzigen	Ja
6	Hoog	Meldingen kunnen verwijderen	Ja
7	Hoog	Melding status wijzigen	Ja
8	Hoog	Overzichten opvragen	Ja
9	Hoog	Zoeken	Ja
10	Middel	Overzichten printen	Ja
11	Middel	Sorteren	Ja
12	Middel	Automatisch invullen velden (datum / naam)	Ja
13	Middel	Controle op invoer bij nieuwe melding of wijziging	Ja
14	Middel	Historie van stappen in de workflow opslaan.	Ja
15	Middel	Wijzigingsgeschiedenis bijhouden van gegevens	Nee
16	Laag	Extra velden toevoegen (GUI)	Nee
17	Laag	Veldrechten aanpassen (GUI)	Nee
18	Laag	Aanpasbare workflow (GUI)	Nee
19	Laag	URL / bijlage toevoegen bij een melding	Nee

Niet functionele eisen

	Prioriteit	Naam	Systeem Concept
1	Hoog	Platform: Sharepoint Portal	Ja
2	Hoog	Configureerbaar webpart (welk overzicht standaard getoond wordt)	Ja
3	Hoog	Andere gegevens opslaan over een wijzigingsverzoek dan over een foutmelding	Ja
4	Hoog	Aanpasbare workflow (via database)	Ja
5	Hoog	Veldrechten aanpassen (via database)	Ja
6	Hoog	Extra velden toevoegen (via database)	Ja
7	Middel	Kolommen in overzicht kunnen aanpassen (via database)	Ja
8	Middel	Sharepoint look and feel	Ja
9	Laag	Aanpasbaar overzicht per klant	Nee
10	Laag	Mogelijkheid om van velden een combo-box te maken.	Nee

Eisen met een hoge prioriteit zijn absoluut noodzakelijk om het systeem te kunnen laten draaien.
Eisen met de prioriteit middel maken het systeem makkelijker in gebruik.
Eisen met de prioriteit laag zijn handig om te hebben, maar niet noodzakelijk voor het slagen van dit project.

2. Systeem Concept

1 Inleiding

Het doel van dit document is het beschrijven van een oplossingsrichting. In dit document worden de functies van het systeem weergegeven vanuit het gezichtspunt van de gebruiker. Door middel van dit document kan er overeenstemming worden bereikt met de opdrachtgever over het op te leveren systeem.
De verschillende actoren worden besproken, met hun bijbehorende use-cases.

2 Use-Case Model

Het use-case model beschrijft de te leveren functionaliteit van het systeem zoals deze door actoren wordt waargenomen.

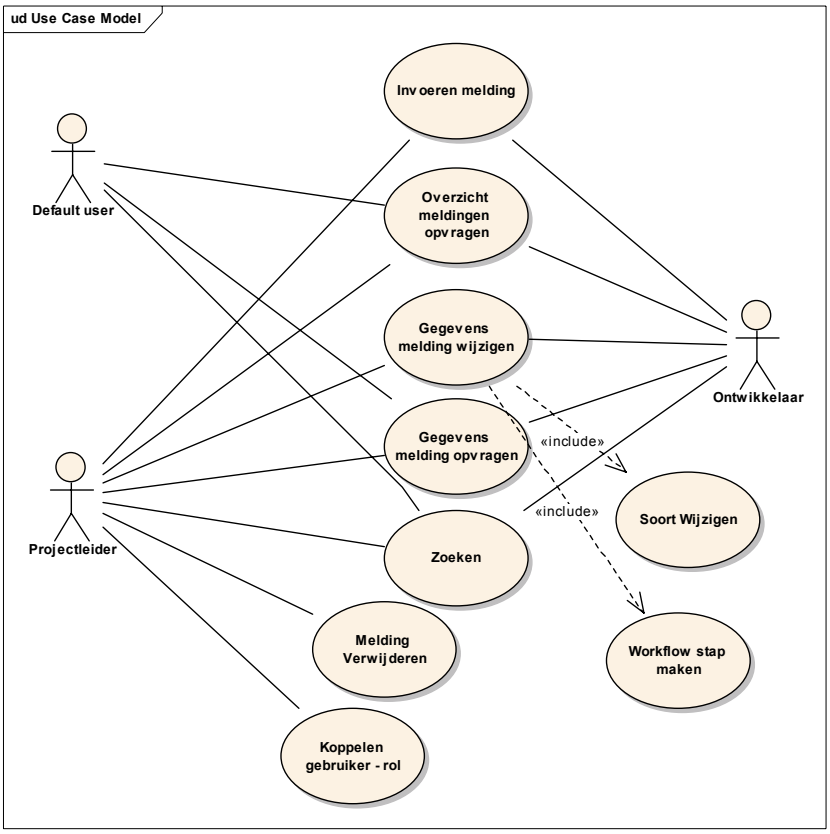
2.1 Actoren

De volgende actoren hebben toegang tot het systeem:

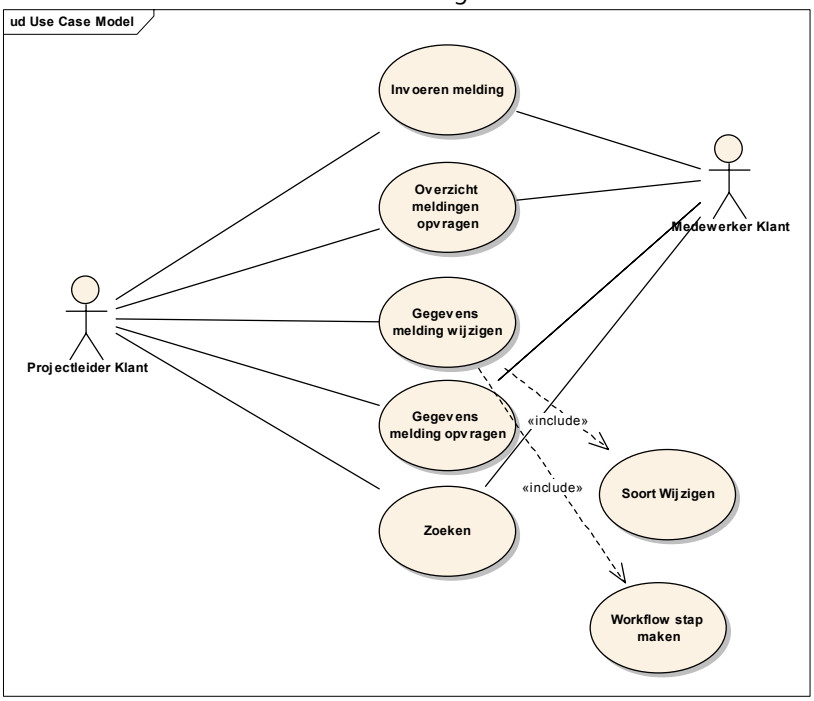
6. Default user
7. Ontwikkelaar
8. Projectleider
9. Medewerker Klant
10. Projectleider Klant

2.2 Use-cases

De use-cases in onderstaand diagram geven weer welke acties een actor kan uitvoeren.



Use-Case Diagram



Use-Case diagram

2.2.1 Koppelen Gebruiker - Rol

2.2.1.1 Doel

Het koppelen van een Sharepoint Portal gebruiker aan een rol in de webpart en het wijzigen of verwijderen van deze koppeling.

2.2.1.2 Samenvatting

De Sharepoint portal waar gebruikers toegang toe hebben, weet niet automatisch welke rol een gebruiker heeft. Om gebruik te kunnen maken van de juiste functionaliteit binnen de webpart moet er dus aan elke gebruiker een rol worden toegekend.
Er zal een scherm worden getoond waarin het mogelijk is om een gebruiker te kiezen uit een lijst met gebruikers en een rol uit een lijst met mogelijke rollen.

Het is mogelijk dat één gebruiker meerdere rollen toegewezen krijgt.

De volgende rollen zijn beschikbaar:
1. Default user
2. Ontwikkelaar
3. Projectleider
4. Medewerker Klant
5. Projectleider Klant

Een gebruiker zal altijd een default user zijn. Deze rol is niet te verwijderen. Hierdoor zal een gebruiker (ook al was hij nog niet bekend bij de webpart) toch de webpart kunnen bekijken.

2.2.1.3 Scenario

Aanmaken nieuwe rol

Stap	Actor	Beschrijving
1	Projectleider	Klikt op de actie ‘koppelen gebruiker - rol’ in het webpart scherm.
2	Systeem	Haalt lijst op met alle huidige gebruikers van deze portal.
3	Systeem	Toont lijst van mogelijke gebruikers in een drop down box.
4	Projectleider	Selecteert gebruiker uit de lijst.
5	Systeem	Toont een lijst met checkboxes voor de rollen. Checkt de checkboxes van de rollen die de huidige gebruiker al heeft.
6	Projectleider	Kent rol toe aan gebruiker(s) en drukt op de knop ‘Opslaan’ om de gebruiker aan de rol te koppelen.
7	Systeem	Maakt een koppeling tussen de gebruiker en de rol aan in de database.

2.2.1.4 Alternatief:

Wijzigen van een rol.

Stap	Actor	Beschrijving
1	Projectleider	Klikt op de actie 'koppelen gebruiker - rol' in het webpart scherm.
2	Systeem	Haalt lijst op met alle huidige gebruikers van deze portal.
3	Systeem	Toont lijst van mogelijke gebruikers in een drop down box.
4	Projectleider	Selecteert gebruiker uit de lijst.
5	Systeem	Toont een lijst met checkboxes voor de rollen. Checkt de checkboxes van de rollen die de huidige gebruiker al heeft.
6	Projectleider	Kent rol toe en drukt op de knop 'koppelen' op de gebruiker aan de rol te koppelen.
7	Systeem	Wijzigt de koppeling tussen de gebruiker en de rol in de database.

2.2.1.5 Open vragen

- Is het mogelijk om een gebruiker meerdere rollen per klant te geven?

  Ja, dit moet mogelijk zijn.

- Een gebruiker is standaard een "default user". Hoe kan de Projectleider geïdentificeerd worden, als hij initieel ook een default user is?

  Het mooist zou zijn als de portal administrator automatisch de projectleider wordt en dus gebruikers mag koppelen. Een andere optie zou een super-user zijn die dit altijd mag. Andere mogelijkheden zijn: de eerste persoon die de webpart bezoekt of een instructie in de handleiding.

- Waar moet de lijst met mogelijke gebruikers vandaan komen? Van "Website users" of "Site Settings > Manage Profile Database"

  Waarschijnlijk maakt dit niet veel uit. Als de gebruikers uit de website users lijst komen is het goed.

2.2.2 Invoeren melding

2.2.2.1 Doel

Het invoeren van een foutmelding, dit vormt het begin van het workflow proces.

2.2.2.2 Samenvatting

De webpart werkt met ingevoerde foutmeldingen. Deze meldingen dienen als input voor het workflowproces. Een melding wordt ingevoerd door een actor en zal vervolgens de status “Defect Open” krijgen.

Bij het invoeren van een foutmelding kan er een naam en een omschrijving worden meegegeven. De datum van aanmelding en de aanmelder zullen worden opgeslagen door het systeem.

De volgende gegevens zullen handmatig door de gebruiker ingevoerd moeten worden:

- Naam van de melding
- Omschrijving

Via het datamodel is aan te passen welke gegevens ingevuld mogen worden bij het invoeren van een melding.

De volgende gegevens zullen automatisch door het systeem worden opgeslagen:

- Naam aanmelder
- Datum

2.2.2.3 Scenario

Stap	Actor	Beschrijving
1	Projectleider	Klinkt op de link “Nieuwe melding”.
2	Systeem	Toont de velden Titel en Omschrijving om een nieuwe melding in te voeren
3	Projectleider	Geeft een titel en omschrijving van de melding op in de invoervelden en drukt op Opslaan en Sluiten.
4	Systeem	Slaat ingevoerde gegevens, plus de datum en aanmelder op in de database.
5	Systeem	Zet de status van de ingevoerde melding.
6	Systeem	Toont op een webpagina dat de invoer succesvol is verlopen.

Alternatieven:

Geen invoer

Stap	Actor	Beschrijving
1	Projectleider	Klinkt op de link “Opslaan en sluiten”.
2	Systeem	Toont de velden Titel en Omschrijving om een nieuwe melding in te voeren
3	Projectleider	Vult niks in en drukt op opslaan.
4	Systeem	Toont aan de gebruiker dat de velden leeg zijn en er iets ingevuld dient te worden.

Bij stap 1 en 3 kan de actor Projectleider vervangen worden door:

- Ontwikkelaar
- Medewerker Klant
- Projectleider Klant

2.2.2.4 Open vragen

- Wat krijgt de klant te zien na het invoeren van een melding?

  Een pagina waarop te zien is dat de invoer succesvol is geweest.

2.2.3 Gegevens melding opvragen

2.2.3.1 Doel

Het opvragen en bekijken van de gegevens die worden opgeslagen over een melding.

2.2.3.2 Samenvatting

De gegevens die in het loop van de workflow worden ingevoerd zijn in detail te bekijken. Vanuit dit scherm heb je in één keer alle gegevens van een melding en deze zijn bijvoorbeeld te printen.

2.2.3.3 Scenario

Stap	Actor	Beschrijving
1	Default user	Klikt op een melding uit de overzichtslijst.
2	Systeem	Bepaalt de status in de workflow.
3	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden.
4	Systeem	Vraagt de historie op van deze melding (alle gemaakte keuzes).
5	Systeem	Toont gegevens, de status in de workflow en de historie van de melding.

Alternatieven:

Extra stappen voor het printen:

Stap	Actor	Beschrijving
6	Default user	Kiest er voor om via de browser te printen.
7	Browser van de actor	Print de pagina.

Bij stap 1 en 6 kan de actor Default user vervangen worden door:

- Ontwikkelaar
- Projectleider
- Medewerker Klant
- Projectleider Klant

2.2.3.4 Open vragen

2.2.4 Gegevens melding wijzigen

2.2.4.1 Doel

De gegevens van een ingevoerde melding aanpassen.

2.2.4.2 Samenvatting

Wanneer een foutmelding in het systeem wordt ingevoerd, worden er enkele gegevens meegestuurd. Van elke melding zijn er na invoering nog meer gegevens op te slaan. Elke rol heeft zijn eigen rechten op bepaalde velden van de gegevens. Niet iedereen kan dus alle gegevens aanpassen. Zie voor een gedetailleerde uitleg over de rechtenstructuur hoofdstuk 4 in dit document.

2.2.4.3 Scenario

Stap	Actor	Beschrijving
1	Projectleider	In het scherm met detailgegevens over een melding wordt er op de link “Wijzig” geklikt.
2	Systeem	Bepaalt aan de hand van de rechten welke velden getoond en gewijzigd mogen worden. Enkel de velden die gewijzigd mogen worden zien er wijzigbaar uit. De andere velden zijn platte tekst.
3	Projectleider	Wijzigt de gegevens en klikt op “Wijzigen en Terug”.
4	Systeem	Slaat de nieuwe waardes op van de gegevens.
5	Systeem	Stuurt de gebruiker naar de pagina met het overzicht van alle meldingen.

Bij stap 1 en 3 kan de actor Projectleider vervangen worden door:

- Ontwikkelaar
- Projectleider Klant

Stap	Actor	Beschrijving
1	Projectleider	In het scherm met detailgegevens over een melding wordt er op de link “Wijzig” geklikt.
2	Systeem	Bepaalt aan de hand van de rechten welke velden getoond en gewijzigd mogen worden.
3	Projectleider	Wijzigt de gegevens van een melding maar haalt de velden “titel” en “omschrijving” leeg en vult een tekst in bij het veld “prijs”.
4	Systeem	Toont dat de velden titel en omschrijving niet leeg mogen zijn en dat er een getal ingevoerd dient te worden bij de prijs.
5	Projectleider	Wijzigt de gegevens en klikt op “Wijzigen en Terug”.
6	Systeem	Slaat de nieuwe waardes op van de gegevens.
7	Systeem	Stuurt de gebruiker naar de pagina met het overzicht van alle meldingen.

Bij stap 1, 3 en 5 kan de actor Projectleider vervangen worden door:

- Ontwikkelaar
- Projectleider Klant

2.2.4.4 Open vragen

- Vanuit welke schermen moet deze functie toegankelijk zijn?

  Vanuit het bekijken van de detailinformatie.

- Kan er bij het bekijken van de detailgegevens direct aanpassingen worden gemaakt? Of moet dit in een apart scherm?

  Dit is het beste om te doen vanuit een apart scherm.

- Gebeurt het wijzigen van de status van een melding in een eigen scherm? Of moet het direct mogelijk zijn bij het bekijken van de detailinformatie over een melding?

  Dit gebeurt in hetzelfde scherm als het wijzigen van gegevens over een melding.

2.2.5 Soort Wijzigen

2.2.5.1 Doel

De projectleider moet kunnen aangeven dat een melding van een ander soort is. Deze use-case vindt plaats als onderdeel van de use-case “Gegevens melding wijzigen”.

2.2.5.2 Samenvatting

Een melding zal altijd van een bepaald soort zijn. Bij een soort hoort een te volgen workflow. Alleen de projectleider kan aangeven dat de melding van een ander soort is.
Het systeem zal de melding dan in de workflow plaatsen die hoort bij dat soort.

De gegevens van de melding die reeds zijn ingevoerd zullen bewaard blijven, mits het nieuwe soort deze velden ook implementeert. Als een veld bij geen specifiek soort hoort, dan zal het veld zichtbaar zijn bij elk soort.

2.2.5.3 Scenario

Stap	Actor	Beschrijving
1	Projectleider	In het scherm met detailgegevens over een melding wordt er op de link “Wijzig” geklikt.
2	Systeem	Vraagt het soort van de melding op.
3	Systeem	Vraagt op welke mogelijke soorten deze melding kan hebben.
4	Systeem	Toont de mogelijke keuzes voor een soort in een dropdown box met de huidige soort standaard geselecteerd.
5	Projectleider	Kiest een nieuw soort.
6	Projectleider	Klikt op de knop “Wijzigen en Terug”.
7	Systeem	Zet de melding onder het nieuwe soort.
8	Systeem	Zet de status van de melding op de start status van de bijbehorende workflow van dat soort.
9	Systeem	Stuurt de gebruiker naar de pagina met het overzicht van alle meldingen.

2.2.5.4 Open vragen

2.2.6 Workflow stap maken

2.2.6.1 Doel

De status van een melding wijzigen, zodat er een bepaalde route in de workflow gevolgd kan worden.
Deze use-case vindt plaats als onderdeel van de use-case “Gegevens melding wijzigen”.

2.2.6.2 Samenvatting

Een melding in het systeem zal altijd in een bepaalde status zijn. Vanuit die status zijn er verschillende mogelijke keuzes beschikbaar om naar een volgende status te gaan. Op deze manier is de workflow te doorlopen.

Zie het hoofdstuk ‘workflow’ voor een beschrijving van de interface waarin deze use-case plaatsvindt.

2.2.6.3 Scenario

Wijzigen van de status.

Stap	Actor	Beschrijving
1	Projectleider	In het scherm met detailgegevens over een melding wordt er op de link “Wijzig” geklikt.
2	Systeem	Bepaalt de status in de workflow en de mogelijke keuzes.
3	Systeem	Toont de mogelijke keuzes voor de workflow status in een dropdown box.
4	Projectleider	Wijzigt de status en klikt op “Wijzigen en Terug”.
5	Systeem	Bepaalt of er keuze is gemaakt en slaat de gemaakte keuze op, met daarbij de tijd en wie deze keuze heeft gemaakt.
6	Systeem	Bepaalt welke nieuwe status bij deze keuze hoort en wijzigt de status van de melding in deze nieuwe status.
7	Systeem	Stuurt de gebruiker naar de pagina met het overzicht van alle meldingen.

Bij stap 1 en 4 kan de actor Projectleider vervangen worden door:

- Ontwikkelaar
- Projectleider Klant

Aan de hand van de ingevoerde permissies in het datamodel bepaalt de applicatie welke rol welke stap mag uitvoeren.

2.2.6.4 Open vragen

- Wat moet er gebeuren als een gebruiker meerdere rollen heeft en deze conflicteren?

  De rol die de stap wél mag maken overheerst.

2.2.7 Overzicht alle meldingen opvragen

2.2.7.1 Doel

Overzicht opvragen van alle wijzigingsverzoeken in het systeem. Dit overzicht zal tevens getoond worden bij het openen van de applicatie.

2.2.7.2 Samenvatting

Gebruikers van het systeem willen niet enkel de individuele incidenten aanpassen. Om een goed beeld te krijgen van al het werk is het handig om overzichten te kunnen opvragen.

Van meldingen worden gegevens bijgehouden, zoals onder andere de prioriteit en verschillende data. Alle meldingen staan door elkaar en het is dus functioneel om te kunnen sorteren op bepaalde velden.

Per scherm zullen elke keer 10 meldingen getoond worden. De volgende 10 kunnen opgevraagd worden door op de pagina nummers te klikken, die onder aan de lijst komen te staan.

2.2.7.2.1 Kolommen

Voor elk veld, dat bij een soort hoort, kan aangegeven worden of het standaard zichtbaar moet zijn in het overzicht.
Met behulp van dit gegeven zullen er drie verschillende overzichten mogelijk zijn in het systeem:

- Bij een overzicht van alle soorten van een klant, zullen alle velden die standaard zichtbaar zijn voor deze soorten voor elke melding getoond worden.
- Bij een overzicht van één soort melding, zullen alle velden die standaard zichtbaar zijn voor dit specifieke soort getoond worden als kolom.
- Via een optie in de webpart (zie het hoofdstuk Webpart) zal het mogelijk zijn aan te geven dat enkel de basis kolommen getoond mogen worden.

Welk overzicht getoond wordt, kan aangegeven worden in het configuratie scherm van de webpart. Zie hiervoor de niet functionele eisen van de webpart.

2.2.7.3 Scenario

Stap	Actor	Beschrijving
1	Default user	Opent de pagina waar de webpart op geïnstalleerd is.
2	Systeem	Bepaalt aan de hand van de webpart instellingen, van welke klant de meldingen getoond worden en van welk soort.
3	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden in het overzicht.
4	Systeem	Toont de eerste 10 meldingen in het overzicht.
5	Default user	Klikt op het nummer “2” onder aan de overzichtlijst om naar de 2e pagina met meldingen te gaan.
6	Systeem	Toont melding 11 tot en met 20 in het overzicht.

Bij stap 1 en 5 kan de actor Default user vervangen worden door:

- Ontwikkelaar
- Projectleider
- Medewerker Klant
- Projectleider Klant

Alternatief 1:
Enkel de wijzigingsverzoeken meegeven.

Dit alternatief zal niet beschikbaar zijn als er al een soort is gespecificeerd bij de configuratie van de webpart. Het is dan niet mogelijk een ander soort op te vragen in dezelfde webpart.

Stap	Actor	Beschrijving
1	Default user	Opent de pagina waar de webpart op geïnstalleerd is.
2	Systeem	Bepaalt aan de hand van de webpart instellingen, van welke klant de meldingen getoond worden en van welk soort.
3	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden in het overzicht.
4	Systeem	Toont de eerste 10 meldingen in het overzicht.
5	Default user	Selecteer “RFC” in de drop down list.
6	Systeem	Bepaalt aan de hand van de webpart instellingen, van welke klant de meldingen getoond worden.
7	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden in het overzicht en haalt vervolgens het overzicht op met het toegepaste filter (alleen RFC's).
8	Systeem	Toont de eerste 10 meldingen in het overzicht.

Bij stap 1 en 5 kan de actor Default user vervangen worden door:
- Ontwikkelaar
- Projectleider
- Medewerker Klant
- Projectleider Klant

Alternatief 2: Sorteren
Na het sorteren zal de eerste pagina worden getoond.

Stap	Actor	Beschrijving
1	Default user	Opent de pagina waar de webpart op geïnstalleerd is.
2	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden in het overzicht.
3	Systeem	Toont overzicht.
4	Default user	Kiest er voor om te sorteren op een kolom door op de kolomnaam te klikken.
5	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden in het overzicht en haalt deze gesorteerd op uit de database.
6	Systeem	Toont overzicht gesorteerd op de kolom op de eerste pagina.

Bij stap 1 en 4 kan de actor Default user vervangen worden door:
- Ontwikkelaar
- Projectleider
- Medewerker Klant
- Projectleider Klant

Alternatief: nieuw scherm

Stap	Actor	Beschrijving
1	Default user	Opent de pagina waar de webpart op geïnstalleerd is.
2	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden in het overzicht.
3	Systeem	Toont overzicht.
4	Default user	Klikt op de titel van de webpart in de Sharepoint pagina.
5	Systeem	Opent de pagina met het overzicht in een nieuw scherm (verder zoals standaard scenario).

2.2.7.4 Open vragen

- Wat toon je in de lijst als kolommen van een bug en RFC andere velden hebben?

  Zowel de velden van de bugs als de RFC's.

- Moeten er overzichten mogelijk zijn per project?

  Nee dit hoeft niet, project mag een veld zijn van melding.

2.2.8 Zoeken

2.2.8.1 Doel

Door middel van het zoeken kunnen meldingen in het systeem gevonden worden.

2.2.8.2 Samenvatting

In het systeem zullen een groot aantal meldingen komen. Deze zijn niet altijd makkelijk te vinden via de overzichten. Daarom kan er worden gezocht naar meldingen.

Het zal mogelijk zijn om te zoeken tussen twee bepaalde data.

De velden waarop gezocht kan worden zijn dynamisch. Deze worden opgebouwd uit de velden waar de gebruiker recht heeft om deze te bekijken. Enkel de velden worden getoond die bij een soort horen die bij de klant hoort.
Er kan op één veld tegelijk worden gezocht. Er kan altijd een datum range aangegeven worden. Er kan ook altijd op nummer gezocht worden.

2.2.8.3 Scenario

Stap	Actor	Beschrijving
1	Default user	Open de pagina waar de webpart op staat.
2	Systeem	Toont een textbox voor het zoeken op trefwoord en de knoppen “Zoek!” en “Geavanceerd Zoeken”
3	Default user	Toetst zoekwoorden in de textbox en klikt op “Zoek!”.
4	Systeem	Zoekt naar meldingen die voldoen aan de zoekwoorden.
5	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden in het overzicht van de resultaten.
6	Systeem	Toont overzicht van de resultaten.

Bij stap 1 en 3 kan de actor Default user vervangen worden door:

- Ontwikkelaar
- Projectleider
- Medewerker Klant
- Projectleider Klant

Alternatief: Uitgebreid zoeken

Stap	Actor	Beschrijving
1	Projectleider	Open de pagina waar de webpart op staat.
2	Projectleider	Klikt op de link “Geavanceerd Zoeken”.
3	Systeem	Toont het zoekscherm en de velden waarop gezocht mag worden (zie boven).
4	Systeem	Verbergt het zoeken op trefwoord.
5	Projectleider	Toetst zoekwoorden en datum in en klikt op “Zoek!”.
6	Systeem	Zoekt naar meldingen die voldoen aan de zoekwoorden.
7	Systeem	Bepaalt aan de hand van de rechten welke velden getoond mogen worden in het overzicht van de resultaten.
8	Systeem	Toont overzicht van de resultaten.
9	Projectleider	Klikt op “Zoeken verbergen”
10	Systeem	Verbergt het zoeken en toont het zoeken op trefwoord.
11	Systeem	Verandert de tekst “zoeken verbergen” in “Zoeken tonen”.

Bij stap 1, 2 en 4 kan de actor Projectleider vervangen worden door:

- Ontwikkelaar
- Medewerker Klant
- Projectleider Klant

2.2.8.4 Open vragen

- Welke velden worden getoond bij de resultatenlijst na het zoeken?

  Dezelfde als in de overzichtlijst.
  Het zou mooi zijn om vanuit de webpart op steekwoord te kunnen zoeken en vanuit een apart scherm op alle velden.

2.2.9 Melding verwijderen

2.2.9.1 Doel

Het doel van deze use-case is het kunnen verwijderen van een ingevoerde melding uit het systeem.

2.2.9.2 Samenvatting

Om overbodige meldingen uit het systeem te kunnen halen, is er de optie om meldingen te verwijderen.
Bij het verwijderen van een melding, zullen alle opgeslagen gegevens over deze melding ook worden verwijderd. De gegevens zijn niet meer terug te halen en zullen definitief worden verwijderd.

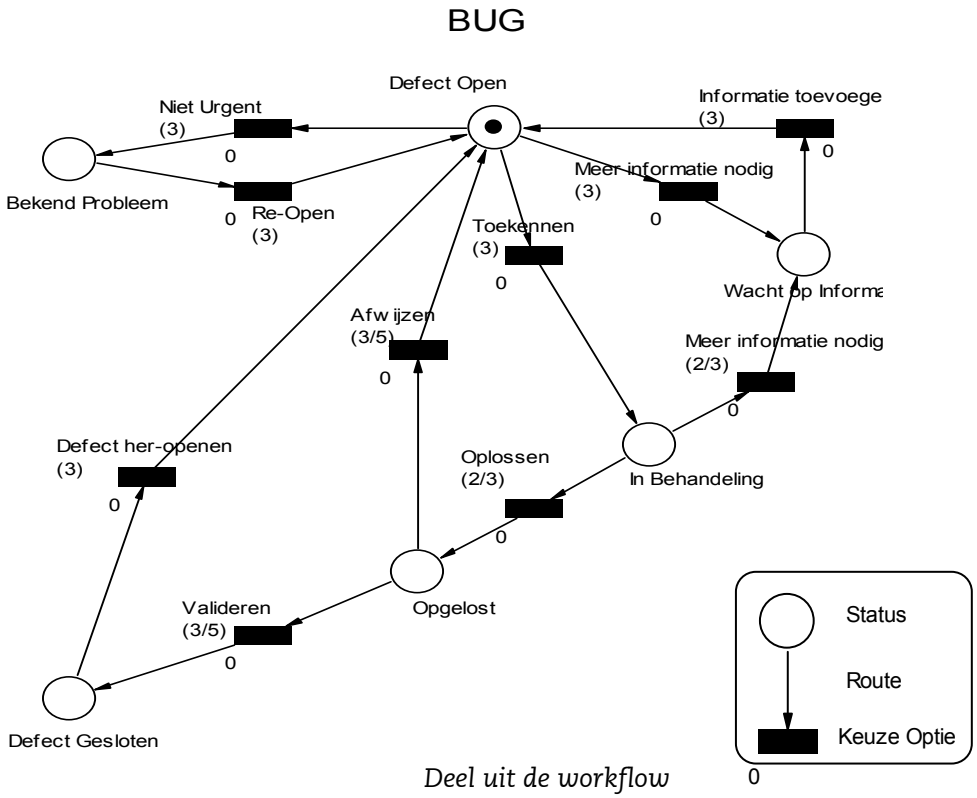
2.2.9.3 Scenario

Stap	Actor	Beschrijving
1	Projectleider	Klikt op de link “Verwijder” in het scherm met de gegevens over een melding / wijzig scherm.
2	Systeem	Toont een scherm met de gegevens over de melding en het verzoek voor definitieve goedkeuring van de verwijdering.
3	Projectleider	Klikt op de knop “Verwijder!”.
4	Systeem	Verwijderd de melding en de bijbehorende gegevens
5	Systeem	Toont op een webpagina dat de melding succesvol verwijderd is.

2.2.9.4 Open vragen

3 Workflow

Een ingevoerde melding zal altijd in een bepaalde status zijn. Vanuit een status zijn er één of meerdere keuzes beschikbaar om de melding in een andere status te krijgen.



Deel uit de workflow

In het scherm waar de detailgegevens van een melding te zien zijn zal er een dropdown-box komen met daarin de mogelijke keuzes. Het is mogelijk om één van de keuzes te selecteren.

Status: Defect Open
Keuze:

- Niet Urgent
- Meer informatie nodig
- Toekennen

Na de keuze voor '*Toekennen*' zal de nieuwe status '*In Behandeling*' zijn. Vervolgens zal het weer mogelijk zijn een keuze te maken en naar een nieuwe status te gaan.

Op deze manier is het mogelijk om alle stappen van de workflow te doorlopen.

De volledige workflow is te vinden in de bijlage.

4 Rechtenstructuur

Onderstaand schema geeft alle velden weer die worden opgeslagen over een melding met daarbij de rechten van de verschillende rollen. Dit schema toont wie welke rechten heeft op de velden nadat de melding in het systeem is ingevoerd.

1. Niet zichtbaar
2. Zichtbaar maar niet wijzigbaar
3. Zichtbaar en wijzigbaar

Algemeen:

Veldnaam	Soort Veld	Projectleider	Ontwikkelaar	Default User	Projectleider Klant	Medewerker Klant
Nummer	Int	2	2	2	2	2
Naam	Mandatory	3	3	2	2	2
Omschrijving	Mandatorylongtext	3	3	2	2	2
Datum Aanmelding	Datum	2	2	2	2	2
Naam Aanmelder	Koppeling met gebruikerstabel	2	2	2	2	2
Toegewezen aan	Text	3	2	1	1	1
Verwante meldingen	Text	3	3	1	1	1
Prioriteit	Text	3	2	1	1	1
Oplossingsrichting	Longtext	3	3	1	1	1
Systeem	Text	3	3	1	1	1

Foutmelding:

Veldnaam	Soort Veld	Projectleider	Ontwikkelaar	Default User	Projectleider Klant	Medewerker Klant
Omgeving	Text	3	3	2	2	2
Datum tijd van de fout	Text	3	3	2	2	2
Fout opgetreden bij	Text	3	3	1	1	1
Technische foutmelding	Longtext	3	3	1	1	1

Wijzigingsverzoek:

Veldnaam	Soort Veld	Projectleider	Ontwikkelaar	Default User	Projectleider Klant	Medewerker Klant
Categorie	Text	3	2	1	1	1
SE Uren	Int	3	2	1	2	1
TE Uren	Int	3	2	1	2	1
PL Uren	Int	3	2	1	2	1
Prijs	Int	3	2	2	2	2
Prijs Commentaar	Longtext	3	3	1	1	1

Zie de paragraaf Datatypen voor een uitleg over de mogelijke datatypen.

4.1.1.1 Open Vragen

- Kan het voorkomen dat in een bepaald stadium de rechten op een veld verschillen met de normale situatie?

  Het zou een nice to have zijn, maar het is niet belangrijk.

- Wat gebeurt er als een wijzigingsverzoek een foutmelding wordt?

  Het zou het mooist zijn als een gebruiker de melding krijgt welke data niet meer beschikbaar is.

5 Niet-functionele eisen

In dit hoofdstuk staan de niet-functionele eisen beschreven met betrekking tot het systeem.

5.1 Webpart

Een Sharepoint portal is een website waar verschillende applicaties benaderbaar zijn. Een webpart is een applicatie die binnen de sharepoint portal weergegeven kan worden. Gebruikers die inloggen op de sharepoint portal hebben een persoonlijk overzicht van de door hun ingestelde applicaties.

De applicatie wordt beschikbaar als webpart binnen de Sharepoint portal. Klanten hebben een eigen portal met daarop een webpart.

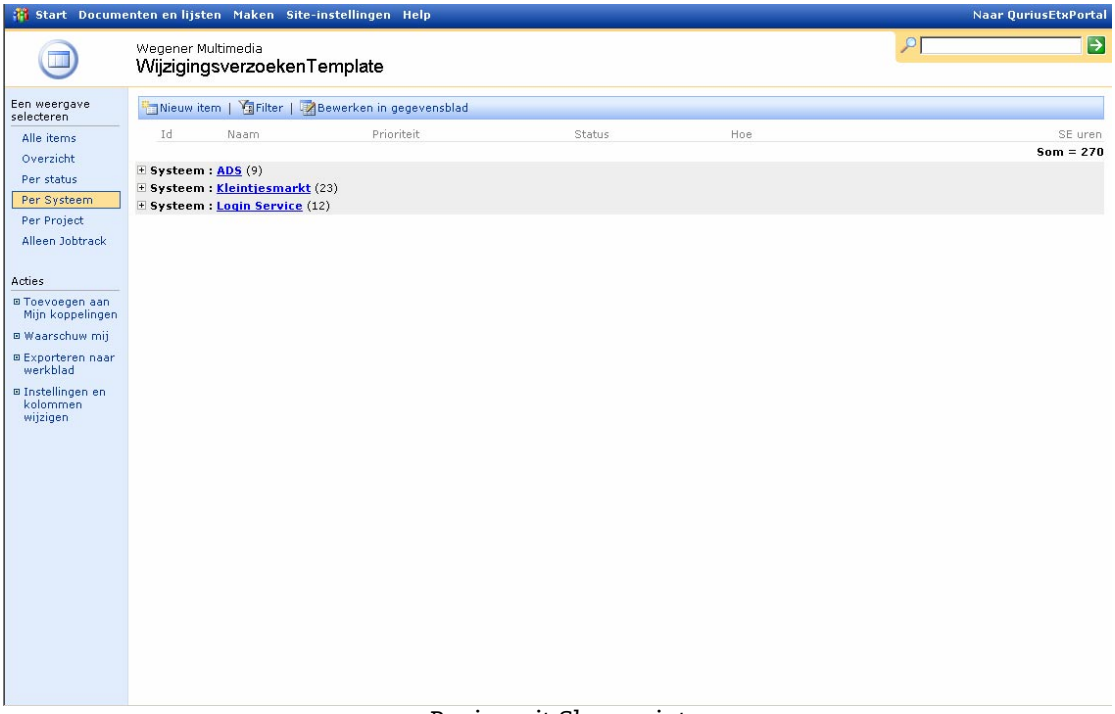
Bij de installatie van de webpart zullen de volgende instellingen kunnen worden gemaakt:

- Voor welke klant is deze webpart.
- Welke soort moet er zichtbaar zijn in de webpart (alle soorten van deze klant, of één specifiek soort).
- Moeten enkel de basis kolommen getoond worden? (Zie de use-case “Overzicht alle meldingen opvragen”).

Wanneer er op de titel van de webpart wordt geklikt, wordt het overzichtscherm in een nieuw venster geopend.

5.2 Layout

Bij het aanmaken van nieuwe pagina’s zal de Sharepoint layout aangehouden worden. Een voorbeeld van een Sharepoint pagina is hieronder te zien:



Pagina uit Sharepoint

Knoppen om naar een nieuwe pagina te gaan, of om gemaakte wijzigingen door te voeren, komen in de lichtblauwe balk boven aan het scherm.

5.3 Datamodel

Via het datamodel is het volgende mogelijk:

- Workflow aanpassen
  De route die in de workflow gevolgd kan worden is aan te passen.
- Veldrechten aanpassen
  Per veld kan aangegeven worden welke rol er welke rechten op heeft
- Extra velden toevoegen
  Er kunnen extra velden worden toegevoegd die bij een specifiek soort horen of voor elk soort zichtbaar zijn.
- Kolommen die getoond worden in het overzicht aanpassen
  Welke kolommen in het overzicht met meldingen weergegeven worden, is aan te passen door per veld aan te geven of deze zichtbaar mag zijn in het overzicht.

Per klant is het volgende in te stellen:

- Welke soorten meldingen de klant kan hebben
- Wat de standaard soort van een klant is

Per soort is in te stellen:

- Hoe de workflow verloopt
- Welke velden er worden opgeslagen over een melding

Workflow = Alle mogelijke routes die een melding kan volgen.
Velden = (bijv Omschrijving) Een veld hoort bij een melding en bevat een waarde over een melding.
Kolommen = Een kolom van alle waardes die bij een specifiek veld en melding horen.

5.4 Datatype

De volgende datatypes kunnen aan een veld gegeven worden:

1) Text
   Normale tekst, komt in een wijzig veld van 1 regel.
   Geen controle op de invoer.

2) Longtext
   Grote tekst, komt in een wijzig veld van meerdere regels.
   Geen controle op de invoer.

3) Mandatory
   Normale tekst, komt in een wijzig veld van 1 regel.
   Invoer van dit veld is verplicht.

4) Mandatorylongtext
   Grote tekst, komt in een wijzig veld van meerdere regels.
   Invoer van dit veld is verplicht.

5) Int
   Getal, komt in een wijzig veld van 1 regel.
   Invoer van dit veld moet een getal zijn.

3. Onderzoek maatwerk of bestaand pakket

1 Inleiding

In dit onderzoek wordt er onderzocht of er het beste voor een bestaand pakket gekozen kan worden om de workflow te implementeren, of dat een maatwerkoplossing een betere keuze is.
Deze keuze heeft gevolgen voor de invulling van mijn afstudeeropdracht. Bij de keuze voor een bestaand pakket zal er meer onderzoek worden verricht en dit pakket waarschijnlijk aangepast moeten worden. Bij maatwerk zal er meer zelf ontwikkeld worden.

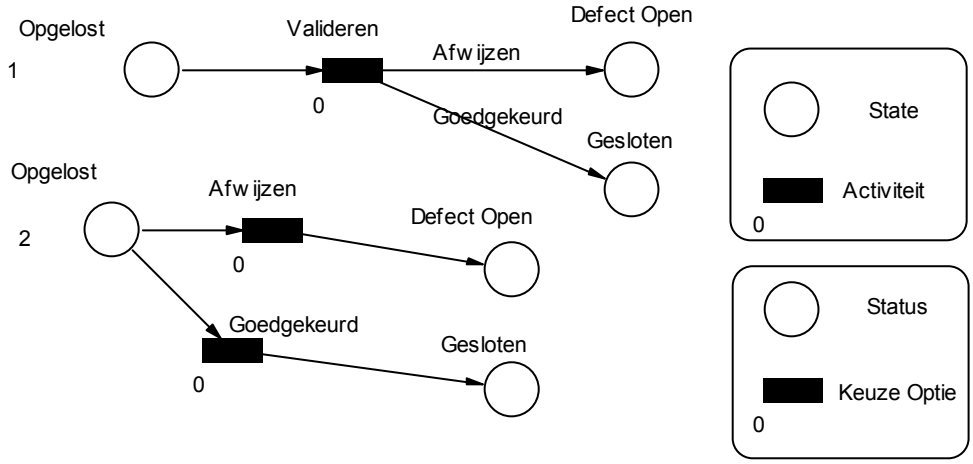
Eerst bespreek ik twee notatie varianten van de workflow die het systeem moet kunnen volgen. Daarna bespreek ik een aantal workflow patterns en tot slot wordt de keuze gemaakt tussen maatwerk en een bestaand pakket.

De keuze voor maatwerk of een bestaand pakket zal worden genomen aan de hand van een aantal criteria:

- Kan het de gewenste workflow volgen?
- De prijs?
- Complexiteit
- Kan het integreren met Sharepoint?

2 Varianten

Er zullen kort twee varianten van het workflow model aan bod komen. Dit zijn twee verschillende manieren om het workflow model grafisch weer te geven. Ze hebben echter gevolgen voor de manier waarop het workflowproces aan de gebruiker wordt gepresenteerd en hoe het datamodel ontworpen zal worden.
Ook heeft het gevolgen voor het kiezen van een bestaand pakket of een maatwerk oplossing, dit staat beschreven in het volgende hoofdstuk.



Deel uit de workflow

Verschil tussen variant 1 en 2:

1) Bij het eerste diagram is er een activiteit Valideren, deze activiteit heeft een begintijd en eindtijd. Je zal de activiteit moeten starten en beëindigen. Tijdens de activiteit kan je kiezen om de melding af te wijzen of goed te keuren. Dit bepaalt welke route de melding zal volgen. De status tijdens de activiteit zal “Valideren” zijn.
2) Bij het tweede diagram zijn afwijzen en goedgekeurd geen activiteiten. De melding is in de status “Opgelost”, vanuit die status kan je kiezen om de melding goed te keuren of af te wijzen. Dit bepaalt of de melding naar de status Open of Gesloten gaat. Bij deze variant hoef je geen activiteit te starten. Vanuit de status “Opgelost” is direct één van de twee keuze opties te kiezen.

Voordelen variant 1:

- Officieel petri-net model
- Complexere modellen zouden later mogelijk zijn

Nadelen variant 1:

- De activiteit zal echt gestart en beëindigd moeten worden.
- Complexer, langere ontwikkeltijd
- Omslachtige interface, kiezen uit activiteiten, dan activiteit starten, dan keuze maken in de activiteit en dan activiteit afsluiten.

Voordelen variant 2:

- Minder complex, daardoor snel duidelijk: steeds een optie en aantal keuzes
- Snellere interface, steeds enkel een nieuwe keuze maken en de melding is in een andere status.

Nadelen variant 2:

- Er wordt niet bijgehouden als iemand werkelijk gaat valideren. Het kan dus voorkomen dat 2 mensen aan het valideren zijn. Pas aan het einde van het validatie proces wordt de keuze gemaakt om goed te keuren of af te keuren.'

De twee varianten zijn met de opdrachtgever besproken. Hij gaf aan dat de tweede variant op dit moment gebruikt wordt. De complexiteit van variant één is niet nodig voor het te maken systeem. De uitgebreide functionaliteit, zoals o.a. het parallel laten lopen van activiteiten, is nu en ook in de nabije toekomst niet nodig.
Een voordeel van de tweede variant vond de opdrachtgever dat het mogelijk is om verschillende rechten te geven aan elke keuze. Iemand kan een melding bijvoorbeeld wel goedkeuren maar niet afkeuren. In de eerste optie is dit niet in te stellen, omdat je alleen kan definiëren wie er mag valideren.

Voor het te implementeren workflowmodel is daarom uitgegaan van de tweede variant.

3 Workflow Patterns

Een pattern is een beschrijving van een vaak voorkomend probleem met daarbij een oplossing die zichzelf heeft bewezen in de praktijk. Er bestaan veel design patterns voor software ontwikkeling.
Workflow patterns beschrijven een vaak voorkomend stukje routering in een workflow. Aan de hand van deze patterns kan een workflow systeem geselecteerd worden. Niet elk bestaand systeem ondersteund alle patterns, maar niet alle patterns zijn altijd nodig.

Er zijn 5 patterns die de basisfunctionaliteit van een workflowsysteem vormen. Daarnaast zijn er nog +- 16 patterns voor meer uitgebreidere functionaliteit.

De 5 basis patterns zijn:

- Sequence
  Een activiteit in de workflow mag uitgevoerd worden na het uitvoeren van de voorgaande activiteit.
- Parallel Split
  Een splitsing in de workflow zodat meerdere activiteiten parallel uitgevoerd kunnen worden.
- Synchronization
  Een punt in de workflow waar meerdere activiteiten samenkomen, de volgende activiteit wordt eenmalig uitgevoerd.
- Exclusion Choice
  Op basis van een beslissing of workflowdata wordt één van meerdere keuzes gekozen tijdens een activiteit.
- Simple Merge
  Een punt in de workflow waar meerdere activiteiten samenkomen en voor elke activiteit wordt de volgende activiteit weer uitgevoerd.

Belangrijk bij het workflow systeem dat gerealiseerd gaat worden is ook nog:

- Deferred Choice
  Dit is een zogenaamde uitgestelde keuze. Op basis van een keuze van de gebruiker wordt één van de activiteiten gekozen. De andere mogelijke activiteiten vallen daarmee weg.

Verschil tussen Exclusion Choice en Deferred Choice:
Het verschil tussen de Exclusion Choice en de Deferred Choice is dat bij de exclusion choice de keuze feitelijk wordt gemaakt tijdens een activiteit.
Bij deferred choice wordt de keuze gemaakt tijdens een bepaalde status, er wordt dan gekozen voor één van de mogelijke activiteiten.

Het workflowmodel dat geïmplementeerd moet worden heeft van de basisfunctionaliteit enkel het sequence pattern nodig. Alle keuzes in dit model zijn zogenaamde deferred choices.

4 Maatwerk of bestaand pakket?

In dit hoofdstuk wordt aan de hand van een aantal criteria een conclusie getrokken.

Kan de applicatie de gewenste workflow volgen?
Aan de hand van bovengenoemde patterns zijn workflow systemen te vergelijken. Op www.workflowpatterns.com is te zien welke pakketen de patterns ondersteunen. Er zijn maar een beperkt aantal pakketten die het deferred choice pattern ondersteunen, zie bijlage 1. De gewenste functionaliteit van een deferred choice is vaak wel via een workaround te bereiken.
Met maatwerk kan het systeem zo opgebouwd worden dat het precies de gewenste workflow kan volgen. De voorkeur bij dit criteria gaat dus uit naar maatwerk.

Complexiteit
De workflow die bij dit project geïmplementeerd dient te worden is betrekkelijk simpel. Er is steeds vanuit een bepaalde status te kiezen voor één of meerdere keuzes. Aan de hand van die keuze ga je naar de volgende status.
Het workflowmodel houdt zich niet bezig met het toekennen van taken, dit is wat de meeste pakketten juist implementeren. De meeste pakketten zijn dus heel complex en uitgebreid, dit is wat de opdrachtgever juist niet wil. Er is nu ook een pakket in gebruik wat te complex is.
Vaak zijn méér mogelijkheden een voordeel. Bij dit project vraagt de opdrachtgever juist om minder mogelijkheden. Hij wil dat het systeem doet wat het moet doen en geen overbodige mogelijkheden die het werk belemmeren. De complexiteit van de meeste bestaande pakketten is hier dus nadelig.
De voorkeur bij dit criteria gaat uit naar maatwerk. Bij maatwerk kan overbodige complexiteit weggelaten worden.

Prijs
De bovengenoemde pakketten zijn niet gratis en zouden aangeschaft moeten worden. De opdrachtgever heeft aangegeven liever niet te betalen voor een nieuw pakket. Er is al reeds de beschikking over Microsoft Biztalk server. Dit opdrachtgever is bekend met dit pakket en wil dit niet inzetten voor dit systeem. Biztalk is namelijk ook te uitgebreid en voldoet dus niet aan het tweede complexiteit criteria. De opdrachtgever heeft aangegeven geen nieuw pakket te willen aanschaffen.

Als reactie hierop heb ik gezocht naar gratis mogelijkheden. MANTIS is een gratis opensource pakket, geschreven in PHP. Het is speciaal gemaakt voor foutmelding afhandeling.
Ook dit pakket voldeed niet aan de gestelde eisen van de opdrachtgever. Zo is het in MANTIS niet standaard dat een melding een bepaalde workflow volgt, dit is wel via speciale codes te implementeren. De workflow is dan alleen niet te wijzigen via de database. Het voldoet dus niet aan de systeemeisen die opgesteld zijn tijdens de definitie studie.
Ook de rechtenstructuur in MANTIS is te beperkt, het is niet mogelijk om bepaalde veldrechten aan bepaalde rollen te geven.
MANTIS lost het probleem dat de opdrachtgever heeft dus niet op.

In de tijd die ik voor dit project heb kan ik zeer waarschijnlijk de eisen met de hoogste prioriteit realiseren. Dit heb ik aangegeven bij de opdrachtgever. Het is voor de opdrachtgever goedkoper om mij een systeem te laten bouwen dan een bestaand pakket aan te schaffen.

Integratie met Sharepoint
In het onderzoek naar de eisen en wensen van het op te leveren systeem kwamen een aantal specialistische eisen naar voren. Zoals bijvoorbeeld dat het systeem moet werken als webpart binnen de Sharepoint server en dat er verschillende workflows per klant aangegeven moeten kunnen worden.
De integratie met Sharepoint van een bestaand pakket zou opgelost kunnen worden door een interface te schrijven tussen een bestaand pakket en de Sharepoint server. Het schrijven van een interface kan evenveel tijd in beslag nemen als het ontwikkelen van maatwerk.
De voorkeur voor maatwerk of een bestaand pakket hangt af van het specifieke pakket en de andere criteria.

Conclusie
Wanneer alle criteria samen worden genomen valt de keuze op maatwerk. De integratie met Sharepoint van het pakket Biztalk is misschien makkelijker en sneller dan maatwerk software op Sharepoint laten draaien. Echter, door de grote complexiteit van Biztalk is dit pakket toch niet geschikt.

Tijdens de afweging tussen maatwerk of een bestaand pakket heeft de voorkeur van de opdrachtgever een belangrijke rol gehad. In een andere situatie, met een andere opdrachtgever met andere voorkeuren, zou de keuze anders kunnen vallen.

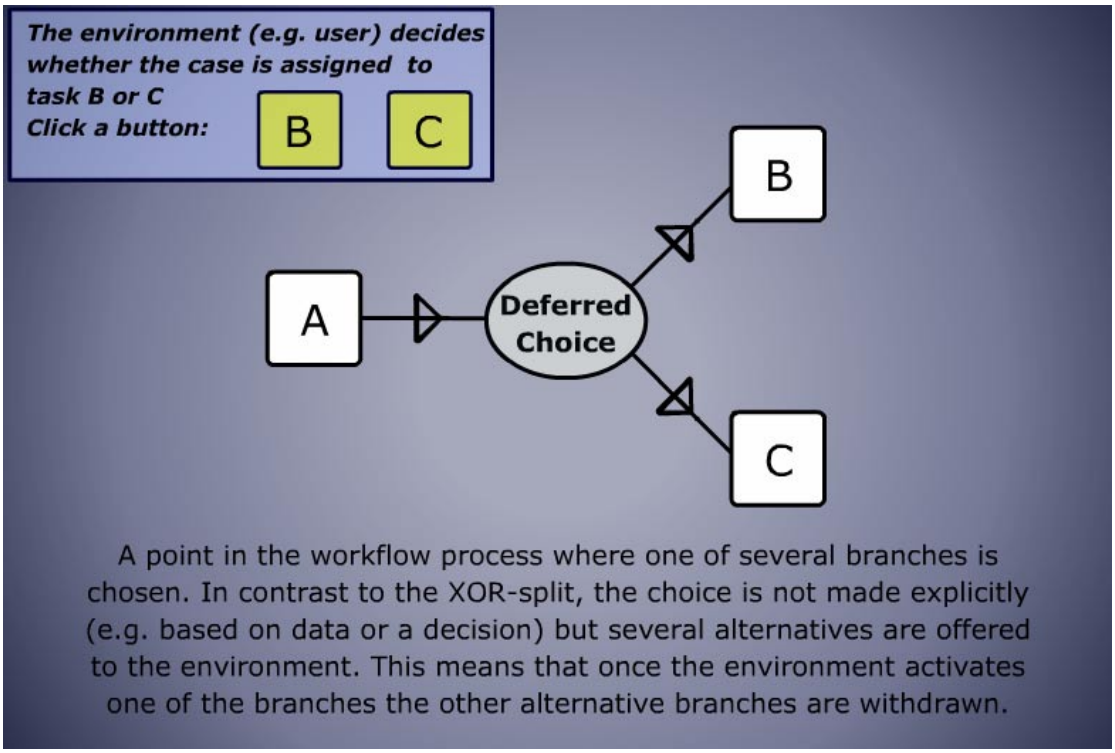
Ik heb de opdrachtgever aan de hand van dit onderzoek geadviseerd om voor een maatwerkoplossing te kiezen, hij is daarmee akkoord gegaan.

5 Bronnen

Workflow Patterns – http://www.workflowpatterns.com

Mantis Bug Tracker - http://www.mantisbt.org/

http://tmitwww.tm.tue.nl/research/patterns/download/IM-patterns.pdf
W.M.P. van der Aalst.
Patronen voor werkstroombesturing. (PDF, 217 Kb)
Management & Informatie, 9(4):4-12, 2001.



6 Product Tabel

http://tmitwww.tm.tue.nl/research/patterns./products.htm

pattern	*products*							
	Staffware	COSA	InConcert	Eastman	FLOWer	Domino	Meteor	Mobile
Sequence	+	+	+	+	+	+	+	+
Parallel Split	+	+	+	+	+	+	+	+
Synchronization	+	+	+	+	+	+	+	+
Exclusive Choice	+	+	+/-	+	+	+	+	+
Simple Merge	+	+	+/-	+	+	+	+	+
Deferred Choice	**-**	**+**	**-**	**-**	**+/-**	**-**	**-**	**-**

pattern	*products*						
	MQSeries	Forté	Verve	Vis. WF	Changemg.	I_Flow	SAP/R3
Sequence	+	+	+	+	+	+	+
Parallel Split	+	+	+	+	+	+	+
Synchronization	+	+	+	+	+	+	+
Exclusive Choice	+	+	+	+	+	+	+
Simple Merge	+	+	+	+	+	+	+
Deferred Choice	**-**	**-**	**-**	**-**	**-**	**-**	**-**

4. Pilotplan

1 Inleiding

Het pilotplan geeft de pilots weer die ontwikkeld en ingevoerd gaan worden. In dit pilotplan zijn de eerste twee pilots in detail beschreven. Per pilot staat aangegeven welke functionaliteit geïmplementeerd gaat worden. De use-cases die genoemd worden in dit document zijn uitgewerkt in het systeem concept.

2 Pilots

In dit hoofdstuk staan de pilots beschreven die uitgevoerd zullen worden. Als eerste zal er een pilot uitgevoerd worden m.b.t. de workflow functionaliteit. Het resultaat van deze pilot zal een volledig werkend systeem zijn. De tweede pilot zal zich meer richten op de rechten van de verschillende gebruikers.

2.1 Pilot 1: Workflow Pilot

2.1.1 Tijd

Voor deze pilot staat een maximum van zes weken.
Voor de eerste pilot is relatief veel tijd gereserveerd. Tijdens deze eerste pilot zal er namelijk nog veel nieuwe kennis moeten worden opgedaan over ASP.NET, de Sharepoint Portal en Webparts.

2.1.2 Doel

Het doel van deze pilot is het realiseren van de belangrijkste functies om meldingen in te voeren en de workflow te doorlopen

2.1.3 Te realiseren functionaliteit

2.1.3.1 Samenvatting

In de eerste pilot zal de implementatie van de workflow centraal staan. De benodigde use-cases om een melding in te voeren en deze door de workflow te voeren zullen gerealiseerd worden.

Het zal mogelijk zijn om een melding in te voeren, de gegevens hiervan op te vragen en de status te wijzigen. De status kan je enkel wijzigen in een volgende stap in de workflow.
Ook is het mogelijk een lijst op te vragen met alle ingevoerde meldingen. Het verwijderen van een melding zal ook mogelijk zijn.

In de eerste pilot zal ook al rekening worden gehouden met de verschillende soorten meldingen per project en per klant. Per soort melding kunnen andere gegevens worden opgeslagen. Bij het invoeren van de melding zal de datum automatisch worden ingevuld en bij het maken van een stap in de workflow zal dit ook worden bijgehouden.

Per use-case staat beschreven wat wel en niet wordt meegenomen in de eerste pilot.

2.1.3.2 Use-Cases

Hieronder staan de namen van de use-cases die betrekking hebben op deze pilot.
De tijd onder elke use-case is de geschatte tijd om deze use-case te ontwerpen en te bouwen.

Nieuw te realiseren use-cases:

- Invoeren Melding
  Tijd: 1 dag
  Deze use-case zal gerealiseerd worden. Met de uitzondering dat niet automatisch wordt bijgehouden wie de melding maakt en wanneer.

- Gegevens melding opvragen
  Tijd: 2 dagen
  Deze use-case zal gerealiseerd worden. Met de uitzondering dat er geen rekening wordt gehouden met welke rol welk veld mag opvragen.

- Gegevens melding wijzigen
  Tijd 3 dagen
  De functionaliteit van deze use-case zal gerealiseerd worden. Met de uitzondering dat er geen rekening wordt gehouden met welke rol welk veld mag wijzigen.
  Alle meldingen zullen in de eerste pilot nog van hetzelfde type zijn, er wordt geen onderscheid gemaakt tussen foutmeldingen en wijzigingsverzoeken.

- Soort Wijzigen
  Tijd: 1 dag
  Deze use-case wordt geïmplementeerd. Er wordt ook bij deze use-case geen rekening gehouden met de rechtenstructuur. Elke gebruiker zal een melding van soort kunnen verwisselen.

- Workflow stap maken
  Tijd: 1 dag
  Deze use-case wordt geïmplementeerd. Er wordt nog geen rekening gehouden met wie welke stap in de workflow mag maken. Wel zal de historie van de stappen in de workflow worden bijgehouden.

- Overzicht alle meldingen opvragen
  Tijd: 2 dagen
  Er wordt van deze use-case het overzicht van alle meldingen gerealiseerd. Er kan gekozen worden enkel de wijzigingsverzoeken of foutmeldingen te zien. Er kan ook gesorteerd worden. Bij het overzicht van de meldingen wordt geen rekening gehouden met welke gebruiker welke kolommen mag zien. Wel kan per veld aangegeven worden of deze in het overzicht zichtbaar mag zijn of niet, zodat niet alle gegevens van een melding in het overzicht staan.

- Melding verwijderen
  Tijd: 1 dag
  Deze use-case wordt geïmplementeerd. Er wordt ook bij deze use-case geen rekening gehouden met de rechtenstructuur. Elke gebruiker zal een melding kunnen verwijderen.

2.1.4 Datamodel

Bij deze pilot zal een datamodel worden ontworpen en gebouwd, ter ondersteuning van de use-cases.

Er moet rekening worden gehouden met de volgende eisen:
- Mogelijkheid tot het aanpassen van de workflow via het datamodel.
- Mogelijkheid om een extra veld toe te voegen via het datamodel.
- In de toekomst moeten er gebruikers en rollen bijgehouden kunnen worden die rechten hebben in de workflow.
- Mogelijkheid om de veldrechten van rollen aan te passen via het datamodel.

2.1.5 Platform

Tijd: 2 dagen
De eerste pilot zal kunnen draaien als webpart binnen de Sharepoint portal. Tevens zullen de pagina's zoveel mogelijk het Sharepoint uiterlijk krijgen. Per webpart zal aangegeven kunnen worden voor welke klant er meldingen getoond moeten worden.

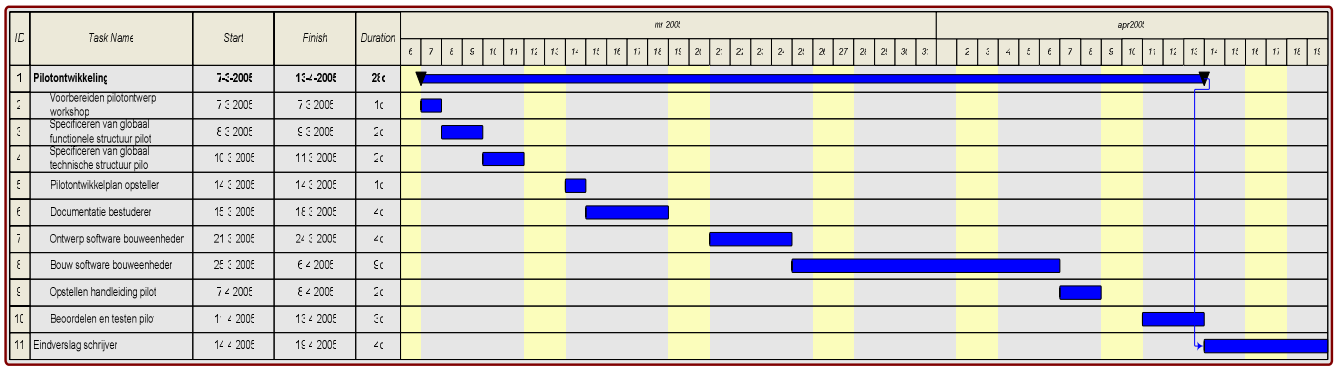
2.1.6 Handleiding

Bij deze pilot zal een beheerders en installatie handleiding worden gemaakt.

2.1.7 Risico

Een risico van deze pilot is dat door te weinig kennis van ASP.NET de voorgenomen functionaliteit hier gerealiseerd kan worden in de afgesproken tijd. Er is daarom in de planning van deze pilot extra tijd vrijgehouden voor het bestuderen van documentatie.

2.1.8 Planning



Het resultaat van deze pilot zal niet ingevoerd worden.

2.2 Pilot 2: Gebruikers en Rechten Pilot

2.2.1 Tijd

Voor deze pilot staat een maximum van vijf weken.

2.2.2 Doel

Het realiseren van de functies die nodig zijn voor de rechtenstructuur, bestaande functies aanpassen zodat deze ook gebruik maken van de rechtenstructuur.

2.2.3 Te realiseren functionaliteit

2.2.3.1 Samenvatting

Tijdens deze pilot zal de rollen- en rechtenstructuur ontwikkeld worden. De functionaliteit om een gebruiker aan een rol te koppelen en om verschillende overzichten van alle meldingen te tonen, zal gerealiseerd worden.
Bestaande functionaliteit zal aangepast worden, zodat deze enkel beschikbaar is voor de juiste rollen. De verschillende stappen om de workflow te doorlopen zullen alleen nog beschikbaar zijn voor de juiste rol.

Tijdens deze pilot zullen ook een aantal extra wensen worden gerealiseerd die naar voren kwam tijdens de beoordeling van de eerste pilot:

- Op elke pagina zal een terug knop komen
- Controle op invoer bij het wijzigen van gegevens
- Optie om enkel de basis kolommen in het overzicht te tonen.

2.2.3.2 Use-cases

Hieronder staan de namen van de use-cases die betrekking hebben op deze pilot. De tijd onder elke use-case is de geschatte tijd om deze use-case te ontwerpen en te bouwen.

Nieuw te realiseren use-cases:

- Koppelen gebruiker – rol
  Tijd: 1.5 dag
  Deze use-case zal volledig gerealiseerd worden.

- Zoeken
  Tijd: 3 dagen
  Deze use-case zal volledig gerealiseerd worden.

Uit te breiden of aan te passen use-cases:

- Invoeren Melding
  Tijd: (onderdeel taak autorisatie)
  Er zal voor worden gezorgd dat enkel de juiste rollen deze use-case mogen uitvoeren. Darnaast zal er bijgehouden gaan worden wie een melding invoert.

- Gegevens melding opvragen
  Tijd: (onderdeel gegevens autorisatie)
  Elke actor mag deze use-case uitvoeren. Er zal wel voor worden gezorgd dat alleen de rollen met de juiste rechten de verschillende velden van een melding mogen bekijken.

- Gegevens melding wijzigen
  Tijd: (onderdeel gegevens autorisatie) + 1 dag
  Er zal voor worden gezorgd dat enkel de juiste rollen deze use-case mogen uitvoeren. Daarnaast zullen alleen de velden waar de rol wijzigrechten op heeft wijzigbaar worden. De andere velden zullen worden gepresenteerd als platte tekst.
  Op de gewijzigde velden zal een controle plaatsvinden of deze ingevuld zijn en voldoen aan het juiste formaat (getal bij een prijs veld bijvoorbeeld).

- Soort wijzigen
  Tijd: (onderdeel taak autorisatie)
  Deze use-case is onderdeel van de use-case “gegevens melding wijzigen”.
  Het wijzigen van een soort zal enkel uitgevoerd mogen worden door een projectleider.

- Workflow stap maken
  Tijd: (onderdeel taak en workflow autorisatie)
  Deze use-case is onderdeel van de use-case “gegevens melding wijzigen”.
  Deze use-case zal zo worden aangepast dat je enkel een stap in de workflow mag maken als je hier recht toe hebt.

- Overzicht alle meldingen opvragen
  Tijd: 1.5 dag
  Alle actoren mogen deze use-case uitvoeren. Niet elke actor heeft echter recht om elk veld te zien. Deze use-case zal dus aangepast moeten worden zodat enkel de velden zichtbaar zijn die gezien mogen worden. De opbouw van de kolommen in de overzichten moet aangepast worden, zoals in het systeemconcept omschreven, zo moet er bijvoorbeeld de mogelijkheid komen om enkel de basiskolommen te tonen.

- Melding verwijderen
  Tijd: (onderdeel taak autorisatie)
  Deze use-case moet zodanig aangepast worden, zodat deze alleen uitgevoerd mag worden door de projectleider.

- Taak Autorisatie
  Tijd: 1 dag
  Het aanpassen van verschillende use-cases (zie boven) zodat enkel de juiste actor de use-case mag uitvoeren.

- Workflow Autorisatie
  Tijd: 0.5 dag
  Enkel de stappen zichtbaar maken in de workflow die uitgevoerd mogen worden door de desbetreffende actor.

- Gegevens Autorisatie
  Tijd: 1,5 dag
  Er voor zorgen dat enkel een actor met rechten gegevens van een bepaald veld mag zien of wijzigen.

2.2.4 Niet functionele eisen

Tijd: 1 dag
De niet functionele eisen die voor deze pilot gelden zijn dezelfde als voor de vorige pilot.

- Look and feel van Sharepoint (o.a. uit te breiden met terug-knoppen)
- Configureerbaar webpart
- Datamodel waarin de workflow en de velden die bijgehouden worden over een melding aangepast kunnen worden.

Voor een uitgebreide beschrijving van de functionele eisen verwijs ik naar het systeem concept.

2.2.5 Kwaliteit

Tijd: +- 1 dag + doorlopend
Door een medewerker van Qurius ETX wordt er een code-review uitgevoerd op de eerste pilot, aan de hand van de standaarden en richtlijnen van Qurius ETX. Dit zal enkele opmerkingen opleveren, over de code van pilot één.
Omdat er aan het begin van dit project is afgesproken dat ik niet volgens de standaarden en richtlijnen hoef te werken, zal er een afweging worden gemaakt, met welke opmerkingen er rekening gehouden gaat worden en of er aanpassingen worden gemaakt in bestaande code. Op deze manier wordt de kwaliteit gewaarborgd.

2.2.6 Handleiding

Bij deze pilot zal een gebruikershandleiding worden geschreven. Daarnaast zal er ook een beheerders en installatie handleiding worden opgeleverd.

2.2.7 Risico

Een risico van deze pilot is dat er geen uitloop mogelijk is doordat dan het einde van de afstudeerperiode is bereikt. Mochten er onverwachte tegenslagen komen zal dit gevolgen hebben voor de inhoud van de pilot, ik zal hier daarom rekening mee houden bij de volgorde van ontwikkeling.
Ik zal geen derde pilot uitvoeren, dus er is geen mogelijkheid om activiteiten, die niet aan bod zijn gekomen in de tweede pilot door de maximale tijd van vijf weken, mee te nemen naar een derde pilot. Aan het einde van deze pilot zal ik daarom een lijst opleveren met functionaliteit die in de toekomst nog geïmplementeerd kan worden om het systeem verder uit te breiden.

2.2.8 Planning

ID	Task Name	Start	Finish	Duration
1	**Pilotontwikkeling**	**25-4-2005**	**27-5-2005**	**25d**
2	Specificeren van globaa - functionele structuur pilot	25-4-2005	26-4-2005	2c
3	Specificeren van globaa - technische structuur pilot	27-4-2005	28-4-2005	2c
4	Pilotontwikkelplan opsteller	29-4-2005	29-4-2005	1c
5	Ontwerp software bouweenheder	2-5-2005	5-5-2005	4c
6	Bouw software bouweenheder	10-5-2005	19-5-2005	8c
7	Opstellen handleiding pilo	20-5-2005	23-5-2005	2c
8	Opstellen invoeringsprocedures	24-5-2005	25-5-2005	2c
9	Beoordelen en testen pilot	26-5-2005	27-5-2005	2c
10	Eindverslag	6-5-2005	9-5-2005	2c

Als alle functionaliteit uit de tweede pilot gerealiseerd is, zal de pilot ingevoerd worden.

3 Overzicht

In onderstaande schema's is een overzicht te zien van alle functionaliteit met hun prioriteit en of ze geïmplementeerd worden in de verschillende pilots.

- Beperkt betekent dat de functionaliteit beperkt gerealiseerd zal worden, bij de eerste pilot zal dit vooral betekenen dat er nog geen rekening wordt gehouden met de rechtenstructuur.
  Voor een precieze omschrijving verwijs ik naar hoofdstuk 2 van dit pilotplan.
- Volledig betekent dat de functionaliteit volledig gerealiseerd zal worden zoals deze is omschreven in het systeem concept.

Use-Cases:

	Naam	Pilot 1	Pilot 2
1	Koppelen gebruiker - rol		Volledig
2	Invoeren melding	Beperkt	Volledig
3	Gegevens melding opvragen	Beperkt	Volledig
4	Gegevens melding wijzigen	Beperkt	Volledig
5	Soort wijzigen	Beperkt	Volledig
6	Workflow stap maken	Beperkt	Volledig
7	Overzichten opvragen	Beperkt	Volledig
8	Zoeken		Volledig
9	Melding verwijderen	Beperkt	Volledig

Systeem eisen:

	Naam	Pilot 1	Pilot 2
1	Andere gegevens opslaan over een wijzigingsverzoek dan over een foutmelding	Volledig	
2	Aanpassen workflow	Volledig	
3	Veldrechten aanpassen		Volledig
4	Extra velden toevoegen	Volledig	
5	Taak autorisatie		Volledig

Webpart eisen:

	Naam	Pilot 1	Pilot 2
1	Mogelijkheid om de webpart te koppelen aan één specifiek soort / klant / (zodat enkel meldingen van die klant / soort / zichtbaar zijn).	Volledig	
2	Draait op Sharepoint	Volledig	
3	De layout van het systeem zal overeenkomen met de Sharepoint layout.	Volledig	Verbeteren

5. Pilotontwikkelplan

1 Inleiding

In het pilotplan worden de pilotdelen van de pilots uiteengezet. De pilotdelen kunnen afzonderlijk gekeurd en getest worden door de opdrachtgever. Dit pilotplan wordt ondersteund door de functionele en technische structuur.

2 Pilot 1

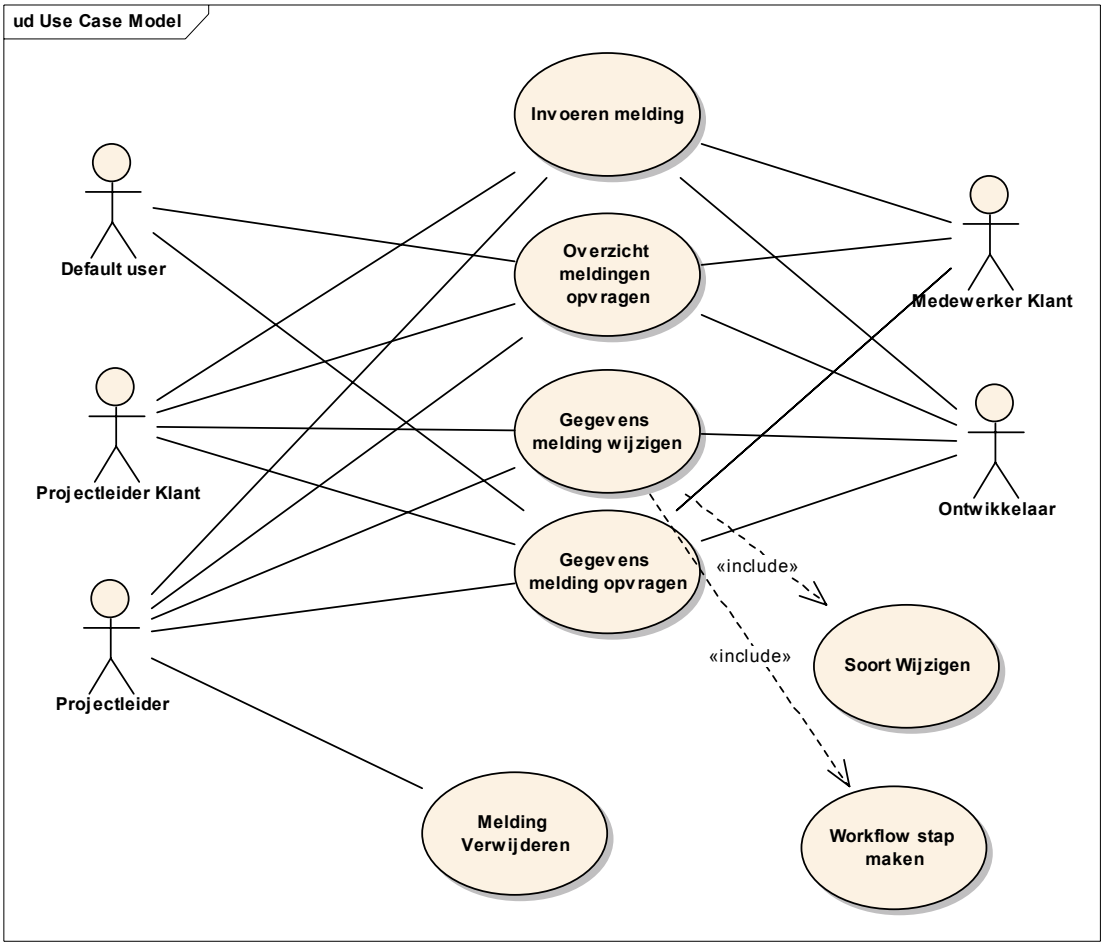
In de eerste pilot zal de implementatie van de workflow centraal staan. De benodigde use-cases om een melding in te voeren en deze door de workflow te voeren zullen gerealiseerd worden.

Het zal mogelijk zijn om een melding in te voeren, de gegevens hiervan op te vragen en de status te wijzigen. De status kan je enkel wijzigen in een volgende stap in de workflow.
Ook is het mogelijk een lijst op te vragen met alle ingevoerde meldingen.

Bij het invoeren van de melding zal de datum automatisch worden ingevuld. Bij de overzichten is het mogelijk om te sorteren.

2.1 Pilotdelen

Tijdens de eerste pilot zullen er zeven use-cases uitgewerkt worden. Elke use-case heeft tevens zijn eigen scherm. Er is daarom voor gekozen om elke use-case een pilotdeel te laten zijn.
Een uitzondering vormen de use-cases “soort wijzigen” en “workflow stap maken”. Deze zijn onderdeel van de use-case “gegevens melding wijzigen”.



Use-cases van pilot 1

Hieronder volgen de use-cases in volgorde van ontwikkeling. Er is voor gekozen voor een volgorde van moeilijkheidsgraad.

Het opvragen van de gegevens over een melding is lastig omdat de gegevens over één melding verspreid staan in de tabel Melding (met de algemene gegevens over een melding), Veldwaarden (met de waarden van een bepaald Veld), de naam van een veld staat uiteindelijk in de tabel Veld. Daarom zal dit uitgesteld worden tot het einde van de pilot.

Als eerste zal het opvragen van de gegevens worden gerealiseerd. Wanneer dit is gelukt en ik meer ervaring heb opgedaan met ASP.NET ga ik verder met het implementeren van het wijzigen van melding gegevens.
Vervolgens zal ik het invoeren en verwijderen van een melding implementeren en het wijzigen van het soort van een melding. Uiteindelijk zal ik het overzichtscherm maken, waar alle meldingen getoond worden.

Volgorde van ontwikkeling:
- Gegevens melding opvragen
- Gegevens melding wijzigen
- Invoeren melding
- Verwijderen melding
- Workflow stap maken
- Soort Wijzigen
- Overzicht meldingen opvragen

2.2 Bouwstenen

De pilotdelen zijn opgebouwd uit de volgende bouwstenen:

- User Interface
- DataSet
- DataAdapter
- BusinessLayer

Elk pilotdeel zal één of meerdere DataSets gebruiken. DataSets gebruiken op hun beurt DataAdapters. Deze DataAdapters worden aangesproken door de klassen.

Ik verwijs naar de globaal technische structuur pilot voor een uiteenzetting van de te ontwikkelen DataSets/DataAdapters en Klassen.

2.3 Beoordelen en testen

Na afronding van een pilotdeel zal dit worden beoordeeld en getest door Michiel Bruins. Eventuele opmerkingen zullen zover mogelijk nog in deze pilot worden meegenomen. Wanneer dit niet mogelijk is zal het in de volgende iteratie aan bod komen als eis of wens.

2.4 Invoering

Deze pilot zal nog niet ingevoerd worden.

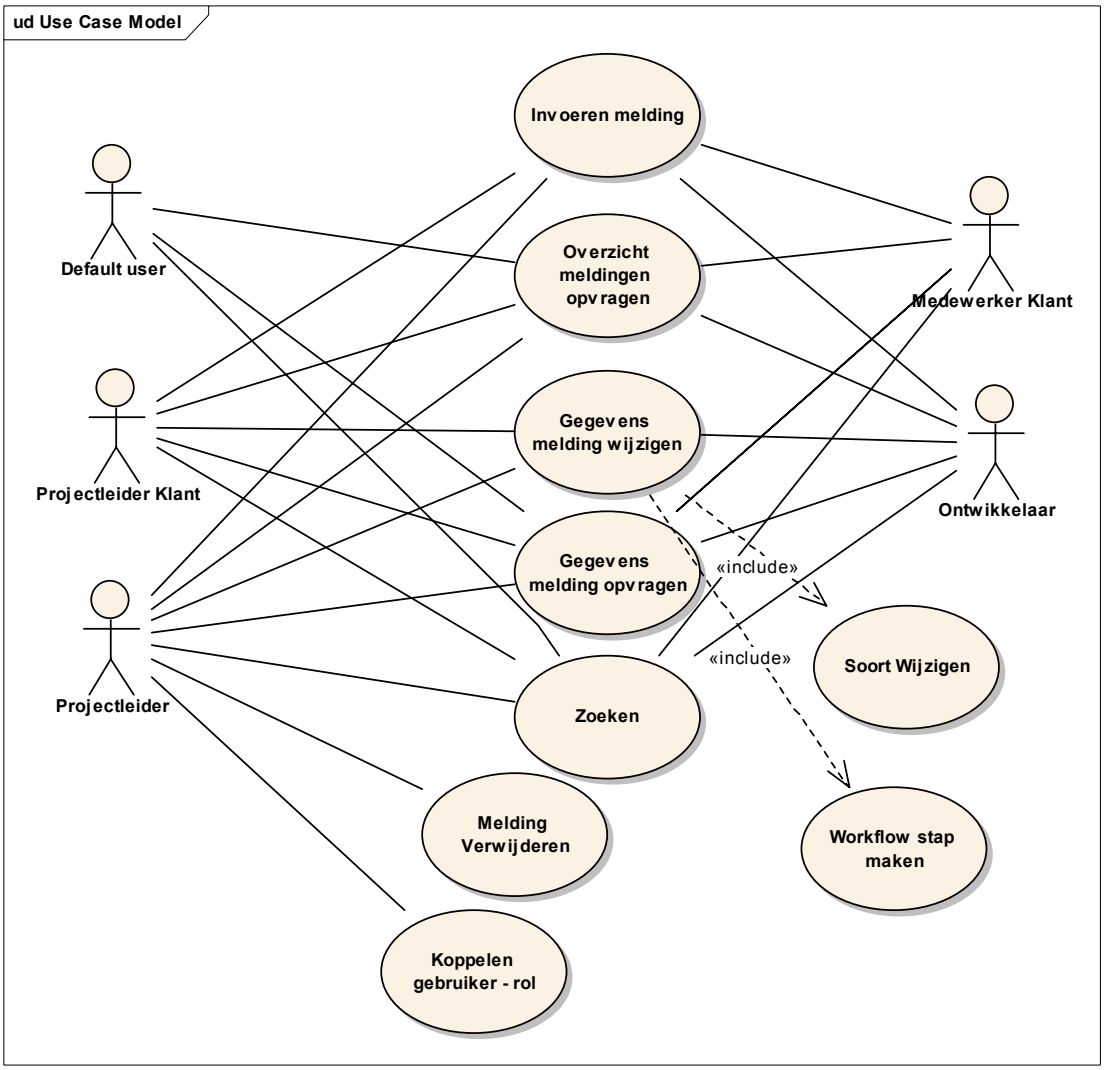
3 Pilot 2

In de tweede pilot zal de applicatie uitgebreid worden met een rollen en rechtenstructuur. De use-case om gebruikers te koppelen aan een rol en om te zoeken zullen geïmplementeerd worden.

Bestaande use-cases zullen worden aangepast, zodat enkel de juiste persoon een bepaalde stap in de workflow mag maken en bepaalde gegevens op mag vragen.

3.1 Pilotdelen

Tijdens de tweede pilot zullen er twee nieuwe use-cases geïmplementeerd worden (zoeken en koppelen gebruiker- rol). Daarnaast zullen er verschillende vormen van autorisatie geïmplementeerd worden.



Use-cases van pilot 2

Volgorde van ontwikkeling:
- Use-case: Koppelen gebruiker – rol
- Taak autorisatie
  Heeft betrekking op alle use-cases.
- Workflow autorisatie
  Heeft betrekking op de use-case: workflow stap maken.
- Gegevens autorisatie
  Heeft betrekking op de use-cases: gegevens melding wijzigen en gegevens melding opvragen.
- Use-case: overzicht meldingen opvragen
  Aanpassen van de verschillende overzichten
- Use-case: zoeken
- Kwaliteitsverbetering

3.2 Bouwstenen

De use-cases zijn opgebouwd uit de volgende bouwstenen:

- User Interface
- DataSet
- DataAdapter
- BusinessLayer

Elke use-case zal één of meerdere DataSets gebruiken. DataSets gebruiken op hun beurt DataAdapters. Deze DataAdapters worden aangesproken door de klassen.

Ik verwijs naar de functionele structuur pilot voor een uiteenzetting van de te ontwikkelen DataSets/DataAdapters en Klassen in de BusinessLayer.

3.2.1 Taak autorisatie

Taak autorisatie houdt in dat er gecontroleerd wordt of een bepaalde gebruiker een pagina mag opvragen en welke elementen op de pagina hij mag zien.

De bouwstenen voor dit pilotdeel bestaan uit:
- Verschillende taken onderscheiden
- Functie bouwen die de taak goedkeurt of afkeurt
- Functie implementeren op alle pagina's

3.2.2 Workflow autorisatie

Workflow autorisatie houdt in dat er gecontroleerd wordt of een bepaalde gebruiker een stap in de workflow mag maken.

De bouwstenen voor dit pilotdeel bestaan uit:
- Aanpassen opvragen workflow keuzes

3.2.3 Gegevens autorisatie

Gegevens autorisatie houdt in dat er gecontroleerd wordt of een bepaalde gebruiker een gegeven mag zien en/of wijzigen.

De bouwstenen voor dit pilotdeel bestaan uit:
- Aanpassen opvragen verschillende velden
- Aanpassen DataList voor de opbouw van de gegevens bekijken / wijzigen

3.2.4 Kwaliteitsverbetering

Tijdens de gehele pilot zal er rekening gehouden worden met de volgende kwaliteitspunten. Er is tevens één dag uitgetrokken om deze punten met terugwerkende kracht te verbeteren:

- Een typed dataset niet untyped aanspreken.
- Het gebruik van vaste waardes zoals 1 of 2 vermijden.
- Betere implementatie van de error afhandeling

3.3 Beoordelen en testen

Na afronding van een pilotdeel zal dit worden beoordeeld en getest door Michiel Bruins. Na afloop van deze pilot zal het testplan, dat is opgesteld aan de hand van de use-cases, uitgevoerd worden om deze pilot te testen.

3.4 Invoering

Deze pilot zal ingevoerd worden. Hiervoor zal een installatie handleiding worden geschreven. De installatie procedure zal eerst worden getest.

Er zullen een aantal stored procedures gemaakt worden die de database vullen met een standaard set gegevens, zoals de velden en de workflow. Zodat het systeem operationeel kan draaien.

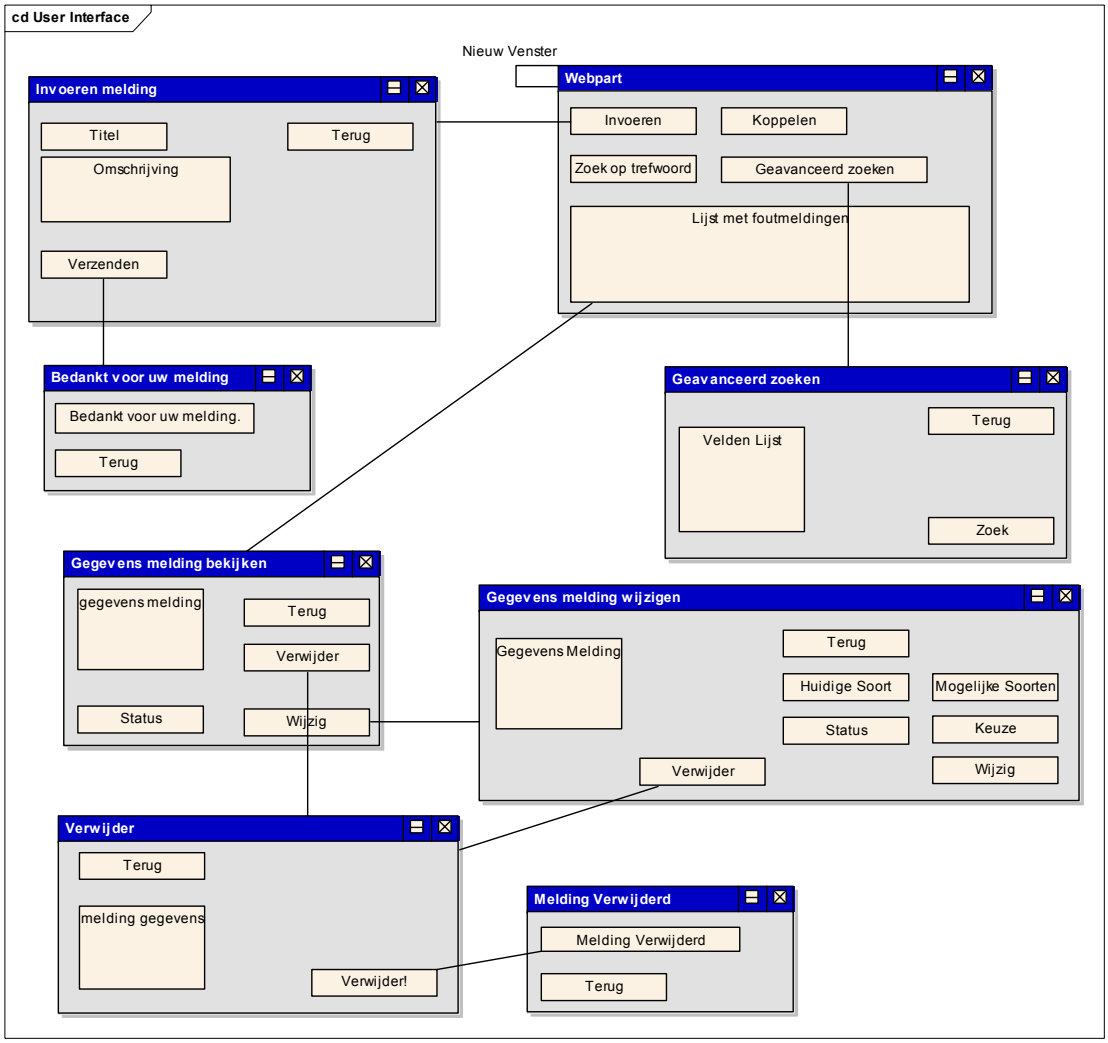
6. Functionele structuur

1 Inleiding

De globaal functionele structuur specificeert de functionele inhoud van de pilot. In dit document staat hoe de vastgestelde use-cases door het systeem afgehandeld worden.
In dit document is tevens de doorloopvolgorde van de schermen weergegeven.

2 Gebruikersinterface

In onderstaand schema is de opbouw van de user interface te zien. Het Webpart scherm zal het begin scherm zijn. Vanuit daar zijn de overige schermen bereikbaar, dit is weergegeven door een verbindingslijn tussen knop en nieuw scherm.



Schermem en hun doorloopmogelijkheden

Hierboven is de interactie tussen de verschillende schermen weergegeven.

3 User Controls

Er worden twee user-controls gebruikt. Het doel van deze controls wordt hier besproken. Het “standaard gegevens melding” user-control kan makkelijk hergebruikt worden.

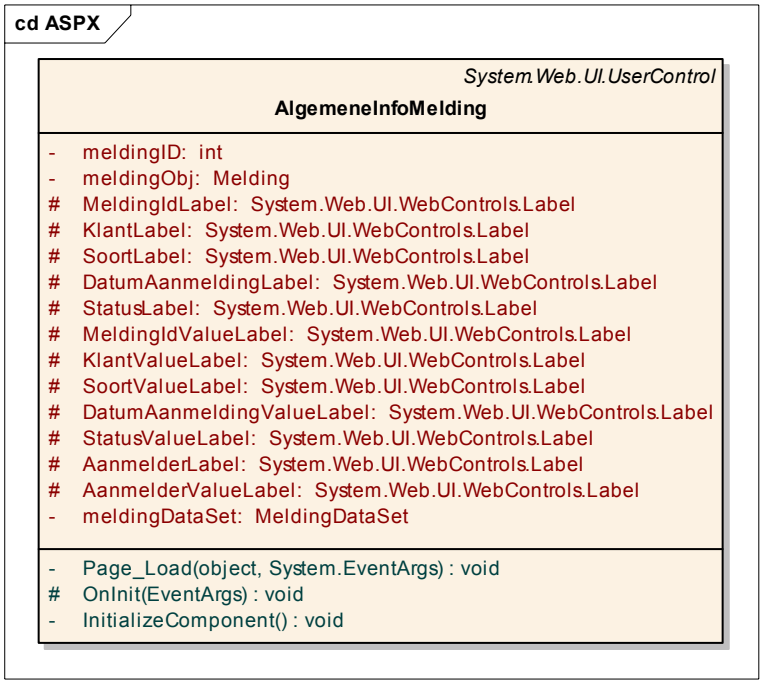
3.1.1 Webpart

Deze user control kan gekoppeld worden met de smartpart waardoor de user control wordt weergegeven als webpart. Dit is de reden geweest om dit user control aan te maken. Het onderstaande overzicht geeft alle objecten en operaties in dit user-control weer.

Attributen en operaties van de user control WorkFlowPart

3.1.2 Standaard gegevens melding

Het volgende user control laat de standaard gegevens over een melding zien. Deze worden weergegeven op meerdere pagina's. Om de code hiervan maar eenmalig te hoeven schrijven is er gekozen om dit in een user control onder te brengen. De schermen die de gegevens over een melding willen laten zien hoeven enkel dit user control te importeren.

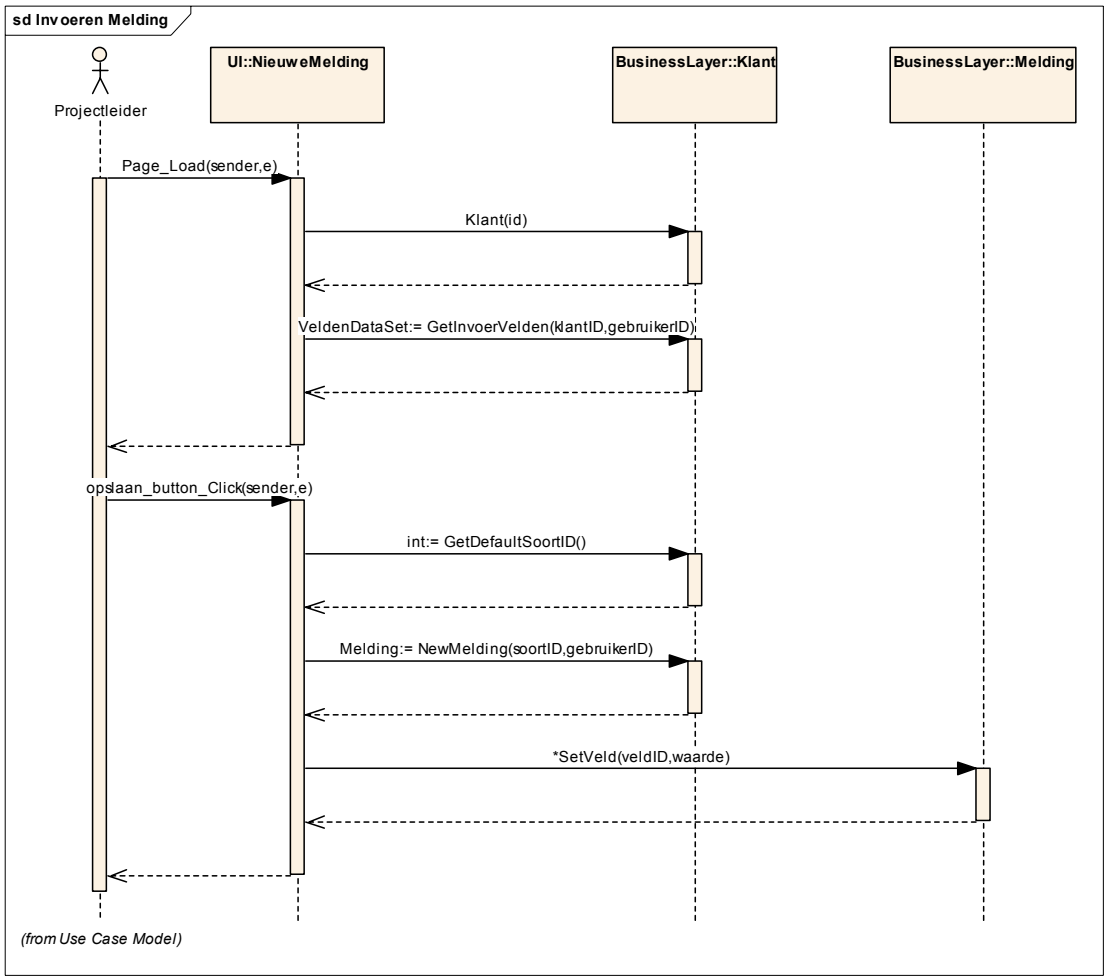


Attributen en operaties van de user control AlgemeneInfoMelding

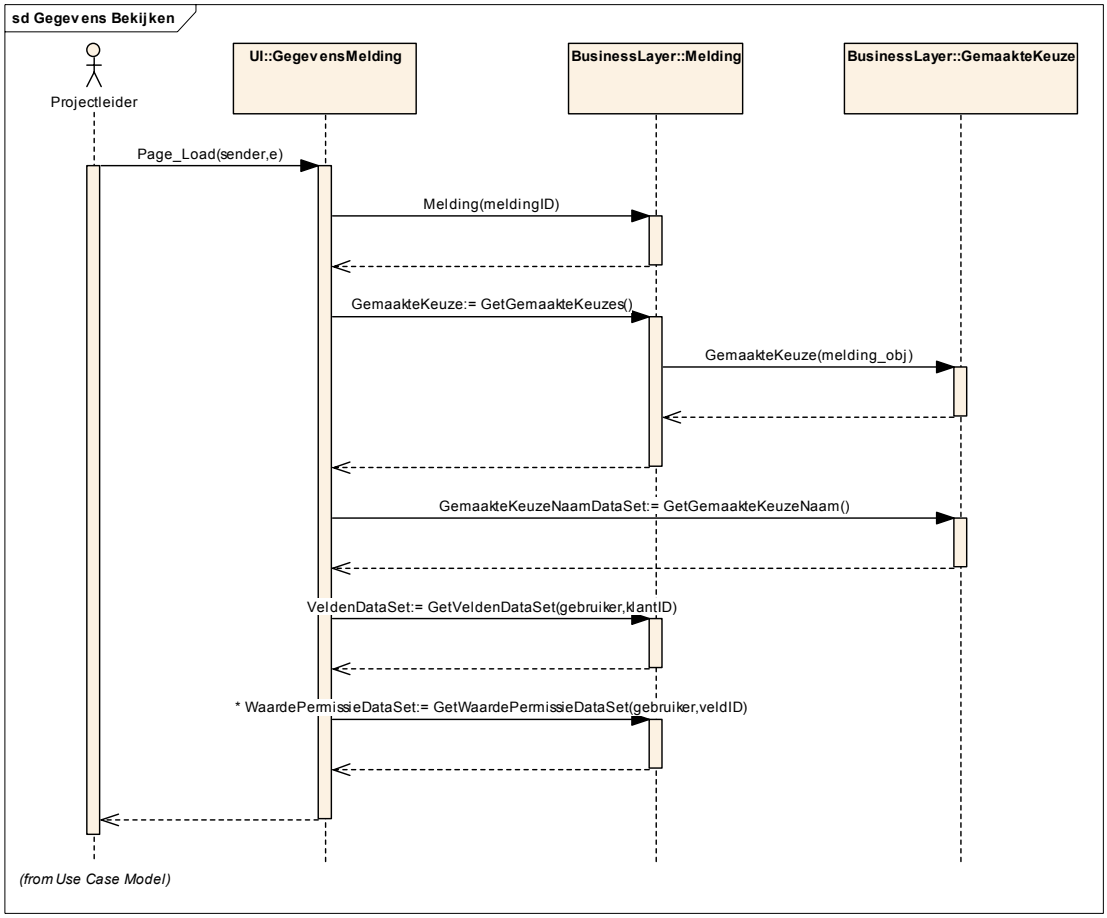
4 Sequentie Diagrammen

In de volgende sequentie diagrammen zijn de use-cases uitgewerkt. Voor de inhoud van de use-cases verwijs ik naar het systeem concept.

4.1 Invoeren melding

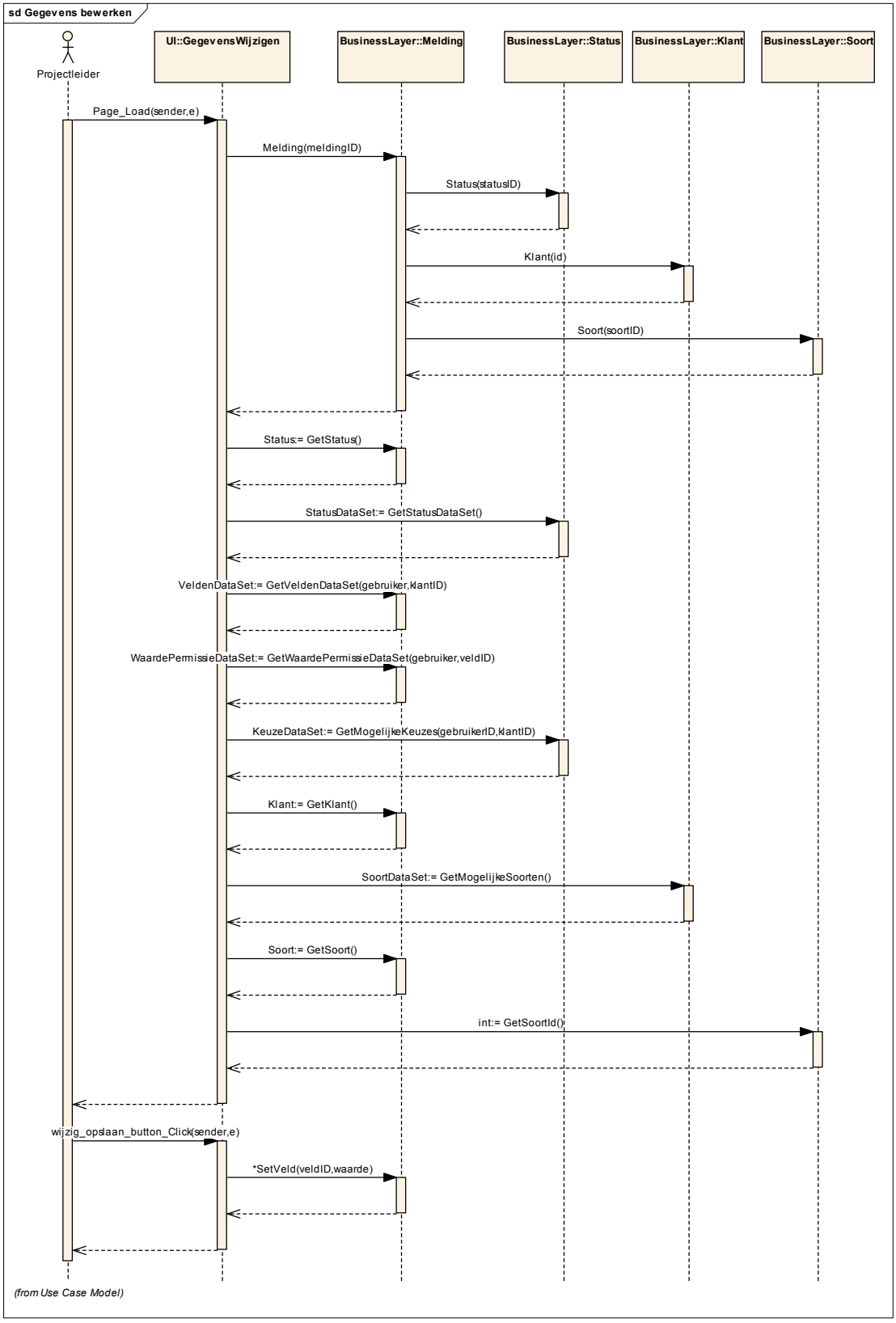


4.2 Gegevens melding opvragen



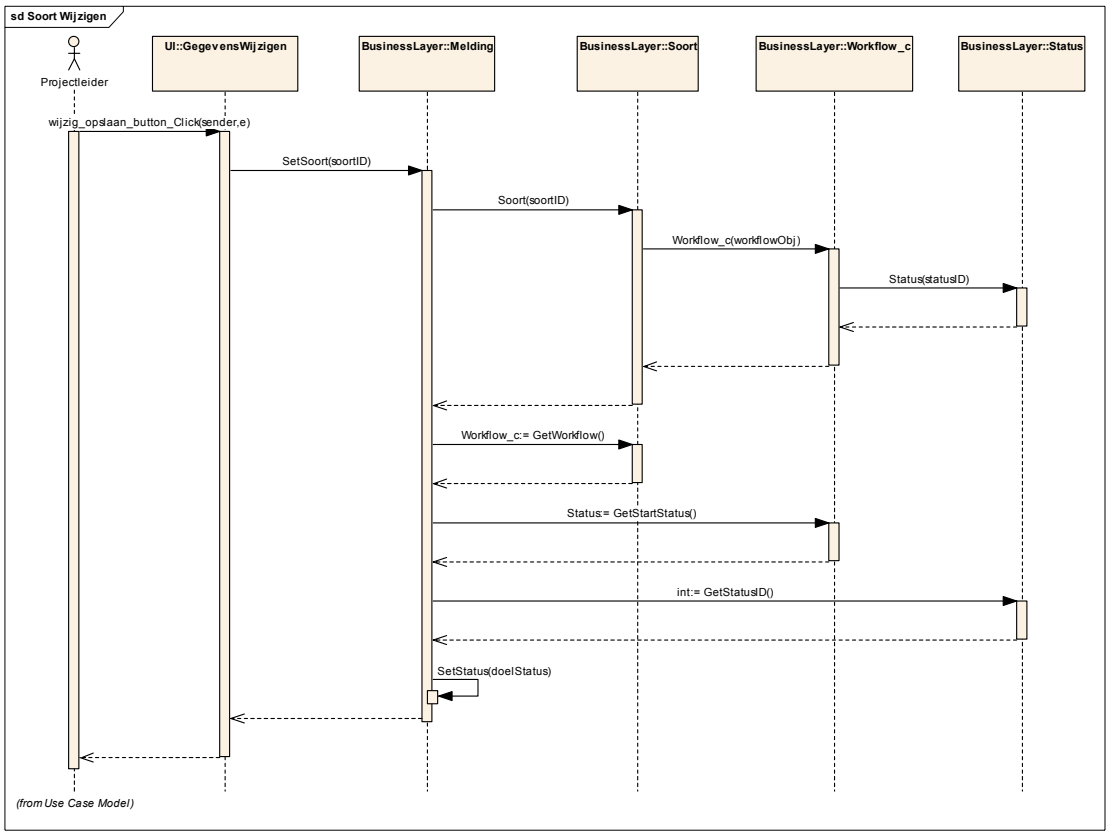
Zie het hoofdstuk “autorisatie” over de afhandeling van de gegevens autorisatie bij dit onderdeel.

4.3 Gegevens melding wijzigen

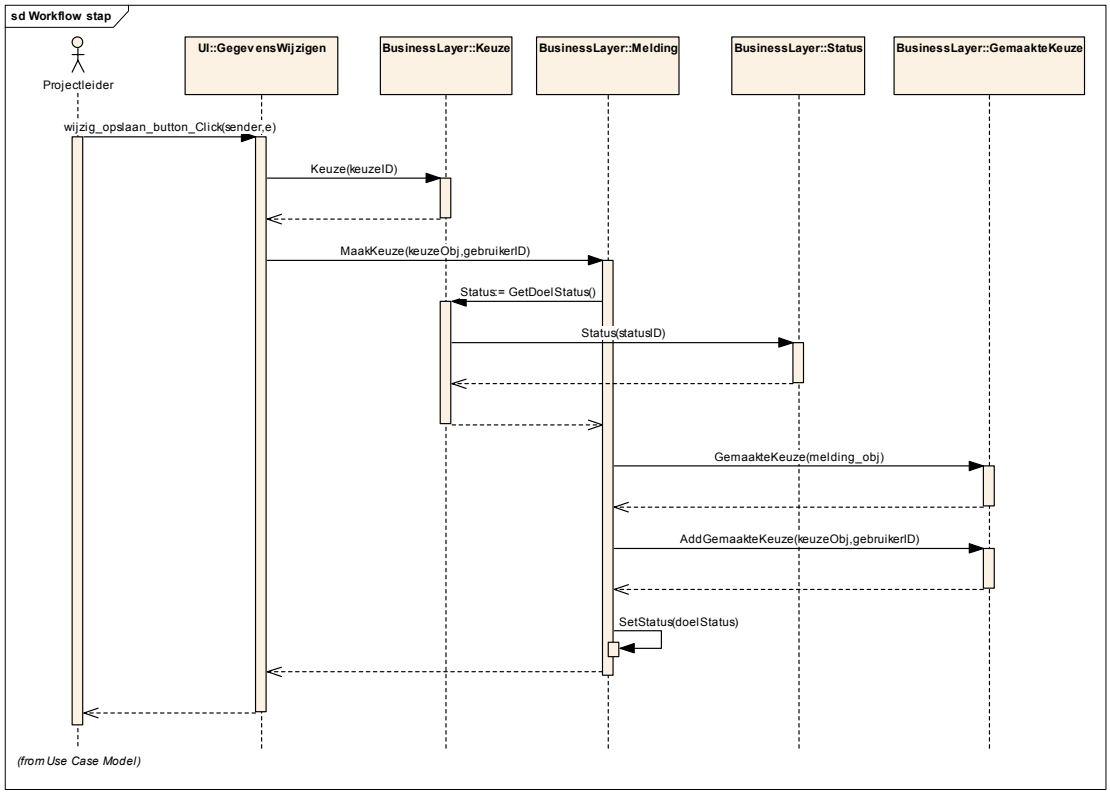


Zie het hoofdstuk “autorisatie” over de afhandeling van de gegevens autorisatie bij dit onderdeel.

4.3.1 Soort Wijzigen

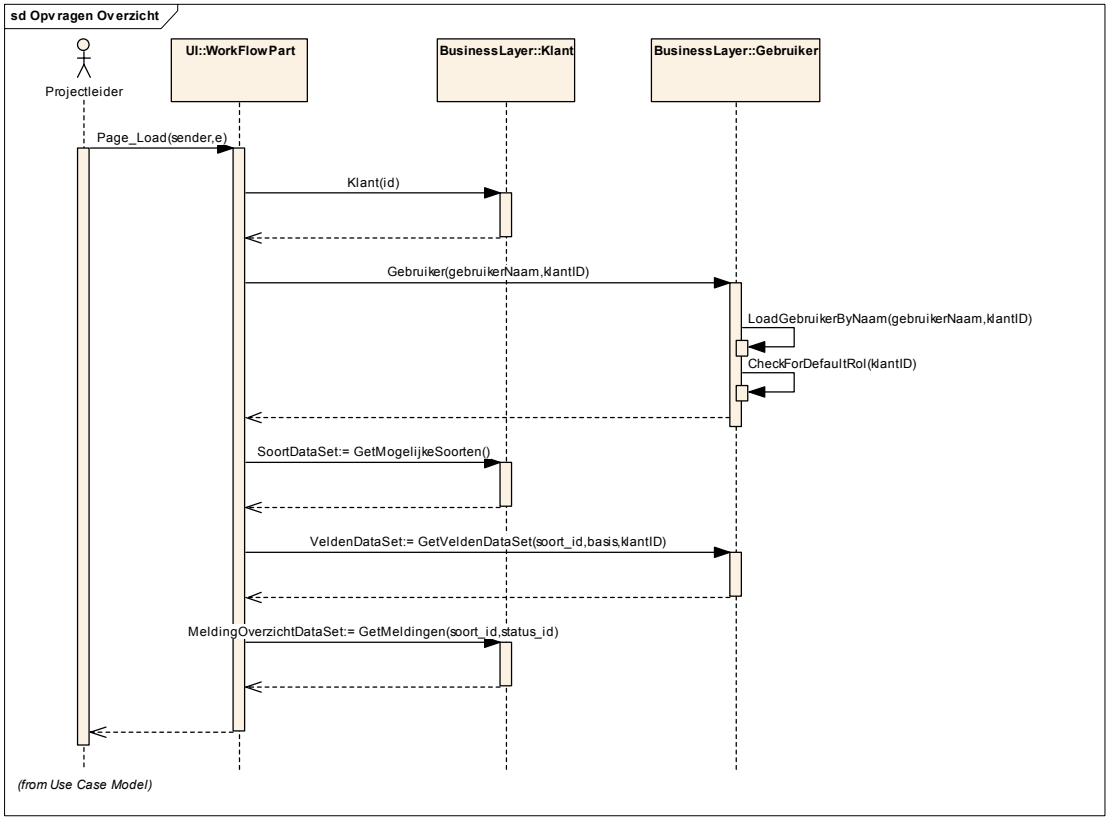


4.3.2 Workflow stap maken



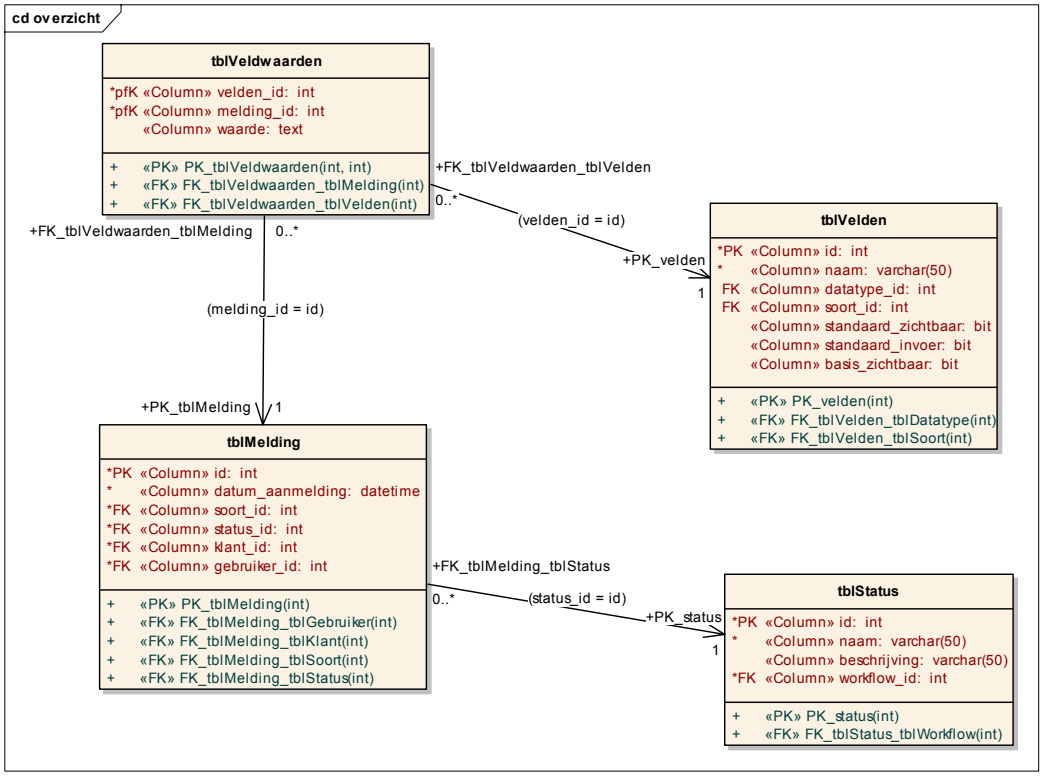
Zie het hoofdstuk “autorisatie” over de afhandeling van de workflow autorisatie.

4.4 Overzicht alle meldingen opvragen



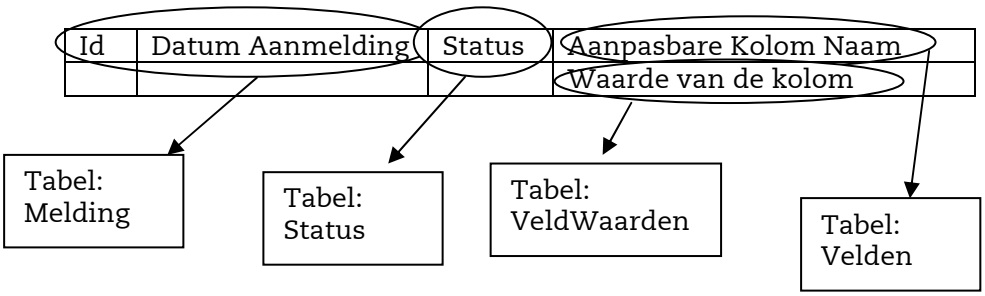
4.4.1 Stappenplan Opbouwen Overzicht Meldingen

Het overzicht met meldingen wordt opgebouwd uit drie tabellen:



De drie tabellen die nodig zijn voor het opbouwen van het overzicht met meldingen.

Om tot het uiteindelijke overzicht te komen, heb ik deze eerst uitgetekend en aangegeven welke gegevens waar vandaan komen. Dit is hieronder te zien.



De naam van de aanpasbare kolomnaam komt uit de tabel Velden.
De volgende velden worden getoond:

- Normaal
  Alle velden die niet bij een soort horen of waarvan het soort bij de klant hoort waarvan het overzicht bekeken wordt. De bit standaard_zichtbaar moet op 1 staan bij deze velden.
- Per soort
  Alle velden van het soort dat bekeken wordt. De bit standaard_zichtbaar moet op 1 staan bij deze velden.
- Basis overzicht
  Alle velden die niet bij een soort horen of waarvan het soort bij de klant hoort waarvan het overzicht bekeken wordt. De bit basis_overzicht moet op 1 staan bij deze velden.

Het overzicht wordt als volgt opgebouwd:

1) Haal alle meldingen van deze klant op uit tblMelding, koppel de tabel aan de status tabel (zodat de Status Naam getoond kan worden in plaats van status_id) en zet deze in een DataSet.

OverzichtDataSet:

Melding_id	Datum_aanmelding	StatusNaam
1	01-01-2005	Aangemeld
2	01-01-2005	In behandeling

2) Haal de naam op van alle velden in tblVelden waarvan het soort_id null is en standaard_zichtbaar true.

VeldenDataSet

id	Naam	soort	Standaard_zichtbaar	Basis_overzicht
1	Naam		1	1
2	Omschrijving		1	0

3) Voor elke naam die is opgehaald uit tblVelden:
    a. Voeg een kolom toe aan de DataSet met de huidige naam

OverzichtDataSet

Melding_id	Datum_aanmelding	StatusNaam	Naam
1	01-01-2005	Aangemeld	
2	01-01-2005	In behandeling	

    b. Haal alle waardes op waarvan het veld_id overeenkomt met huidige veld en waar het melding_id hoort bij de huidige klant.

TijdelijkeDataSet

Velden_id	Melding_id	waarde
1	1	Testnaam
1	2	Melding 2

    c. Voeg per veldwaarde de waarde toe aan de DataSet bij de corresponderende melding.

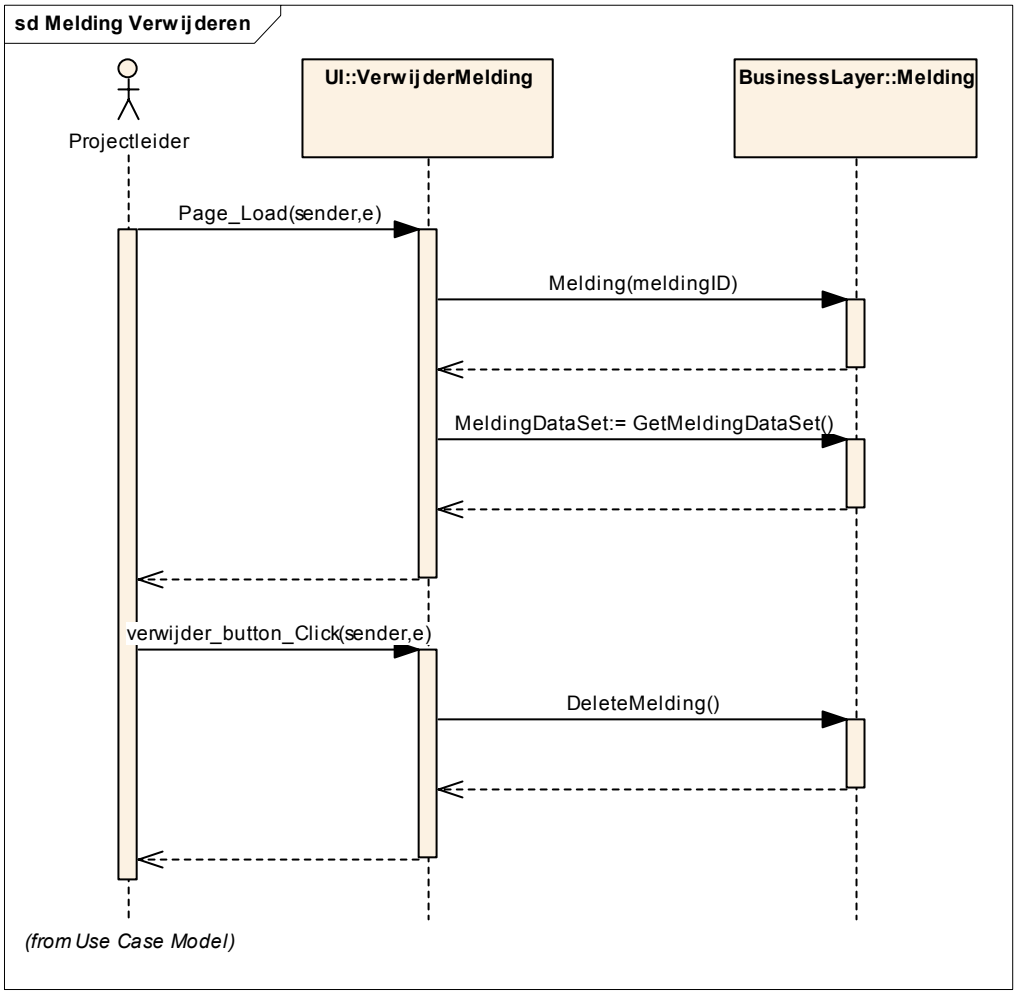
Melding_id	Datum_aanmelding	StatusNaam	Naam
1	01-01-2005	Aangemeld	Testnaam
2	01-01-2005	In behandeling	Melding 2

4) Toon de DataSet.

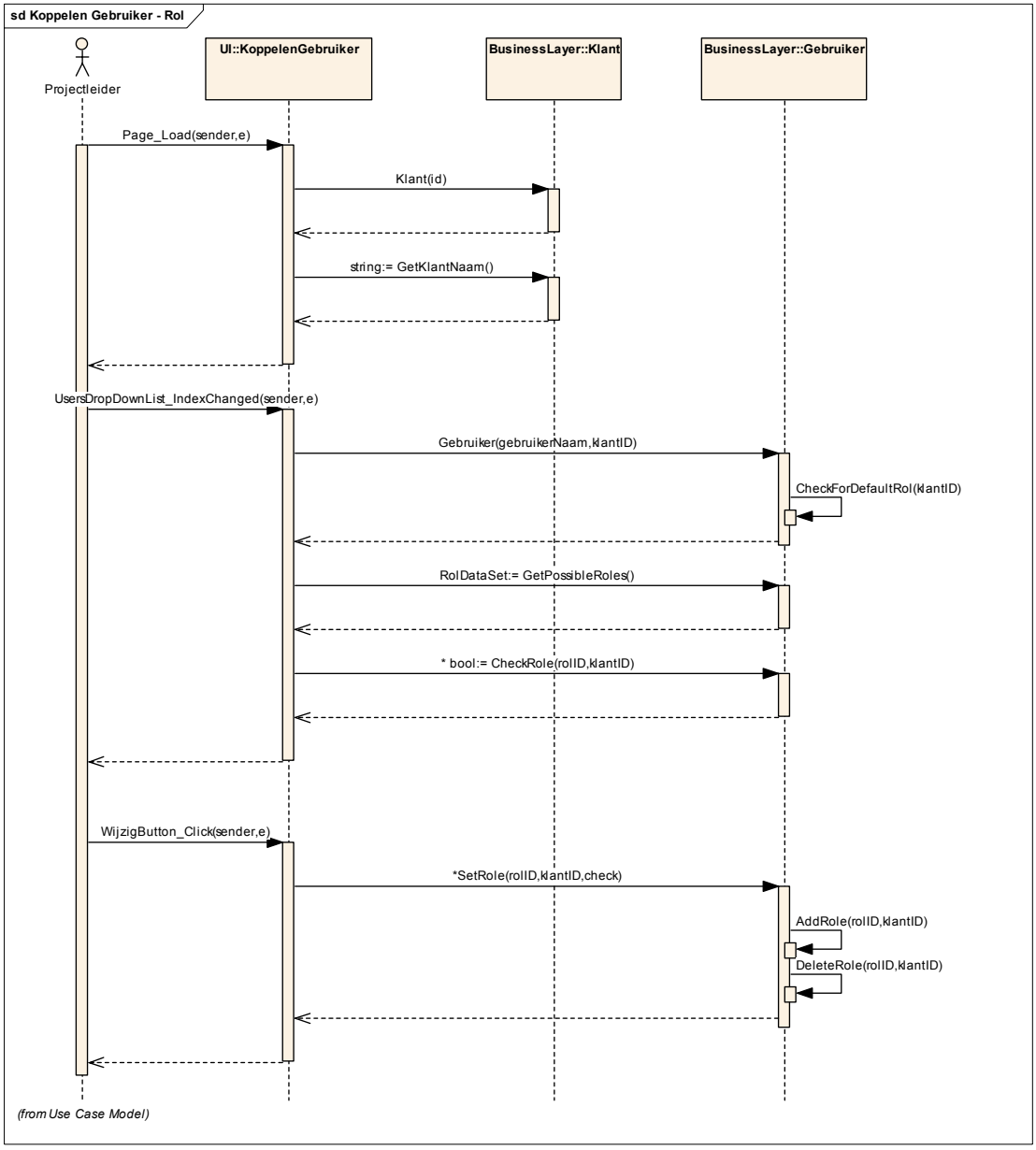
OverzichtDataSet na 2 loops:

Melding_id	Datum_aanmelding	StatusNaam	Naam	Omschrijving
1	01-01-2005	Aangemeld	Testnaam	Test 1
2	01-01-2005	In behandeling	Melding 2	Test 2

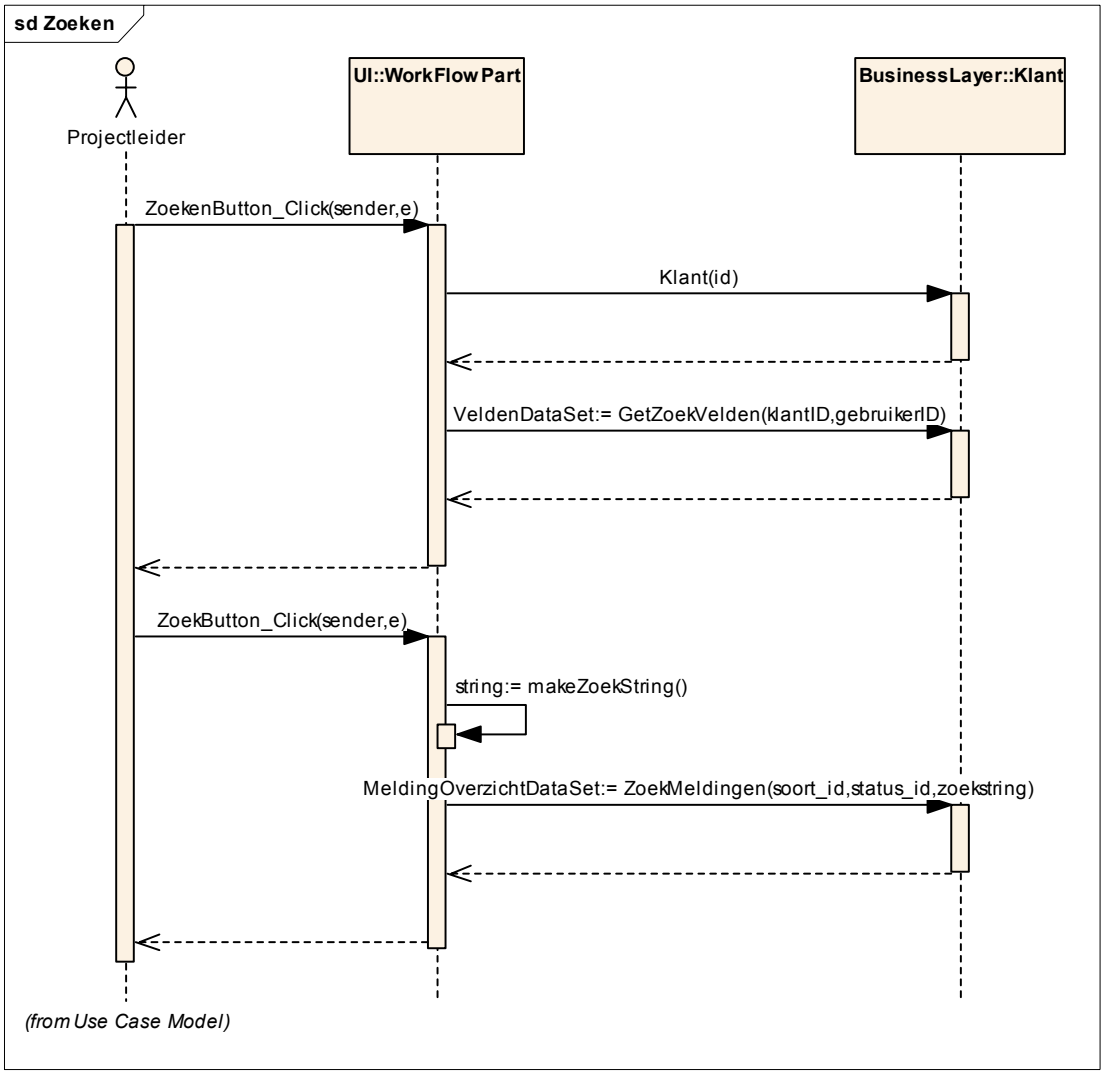
4.5 Melding Verwijderen



4.6 Koppelen Gebruiker – Rol



4.7 Zoeken

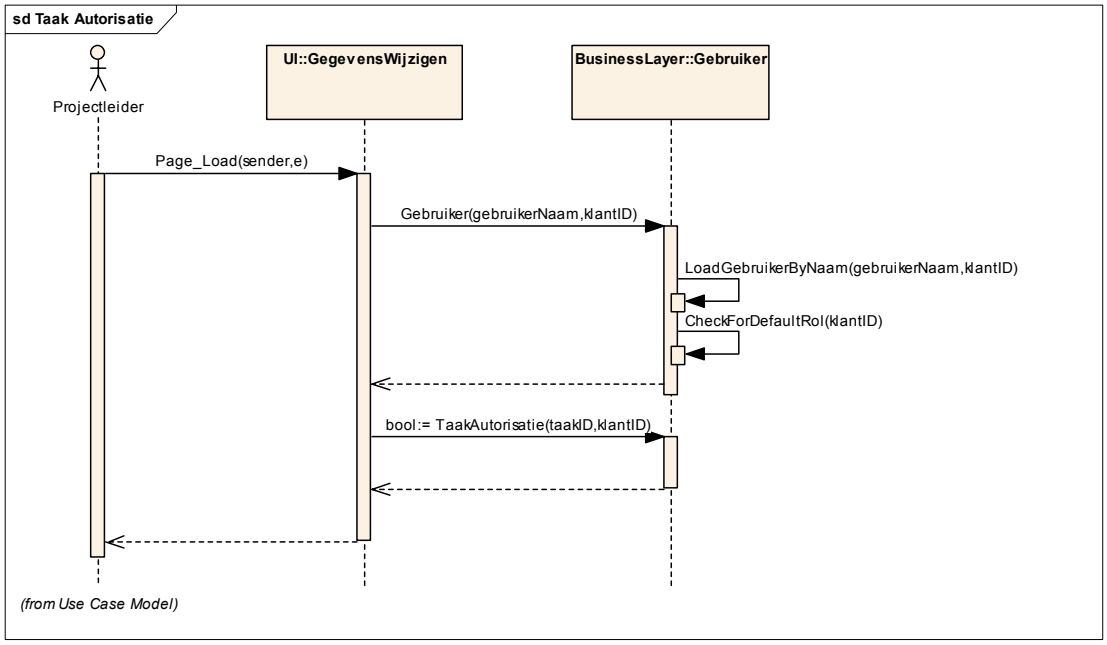

sd Zoeken
Projectleider
UI::WorkFlowPart
BusinessLayer::Klant
ZoekenButton_Click(sender,e)
Klant(id)
VeldenDataSet:= GetZoekVelden(klantID,gebruikerID)
ZoekButton_Click(sender,e)
string:= makeZoekString()
MeldingOverzichtDataSet:= ZoekMeldingen(soort_id,status_id,zoekstring)
(from Use Case Model)

5 Autorisatie

Dit hoofdstuk bespreekt de drie vormen van autorisatie.

5.1 Taak Autorisatie

De uitvoering van de taak autorisatie gebeurt als volgt:



5.2 Workflow Autorisatie

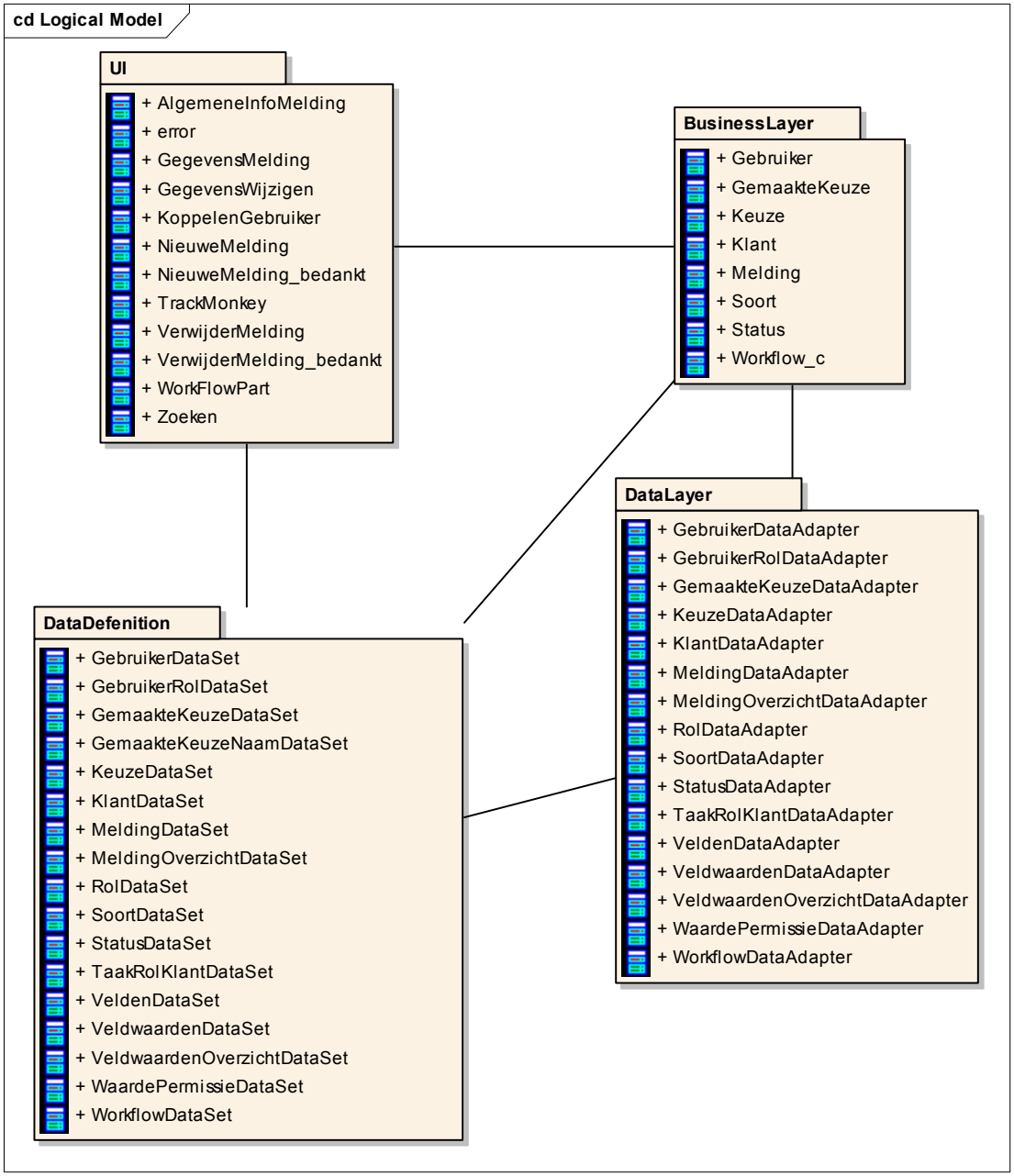
Gebeurt via de SQL query die de mogelijke keuzes opvraagt.
Zie de technische structuur.

5.3 Gegevens autorisatie

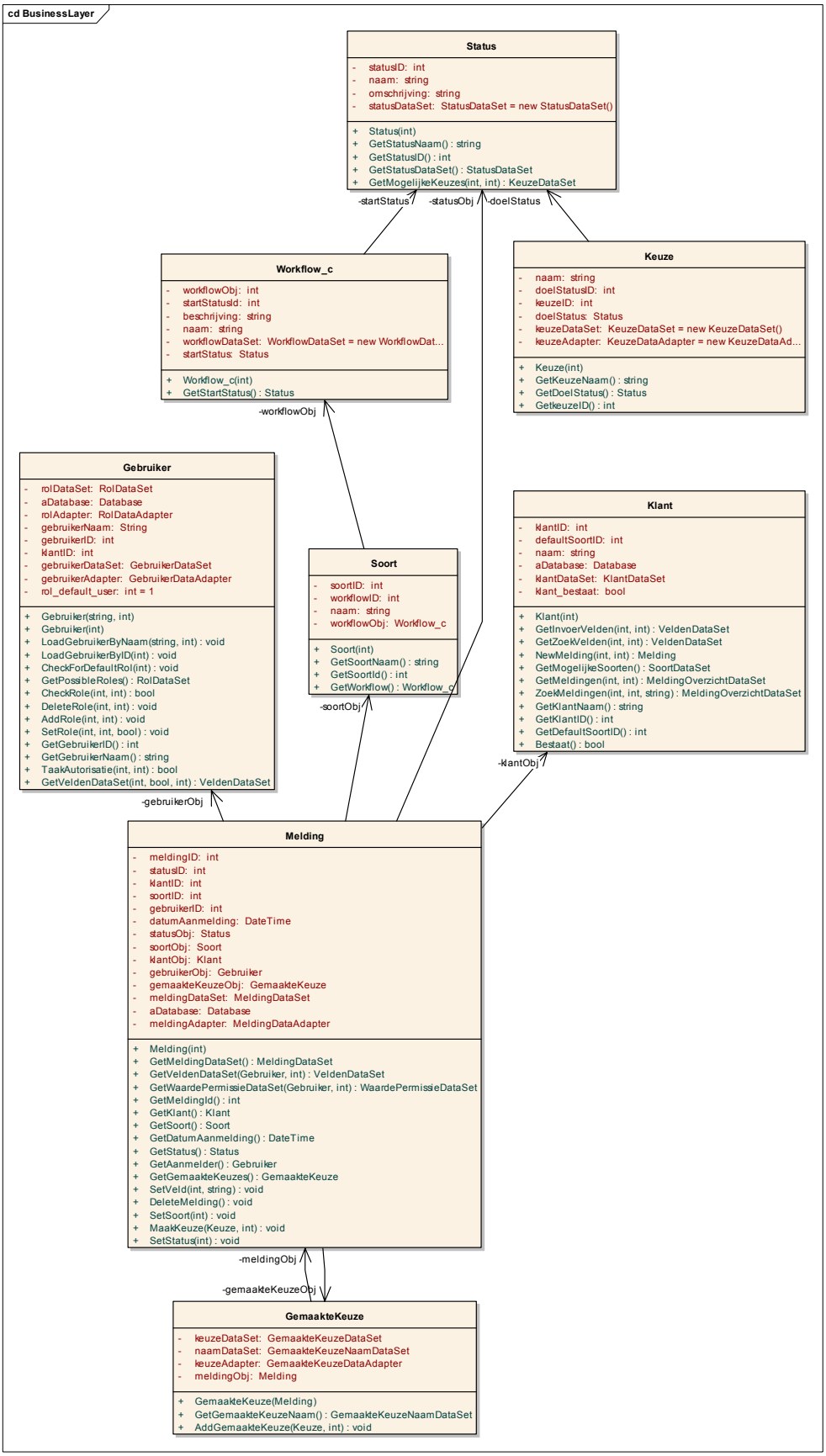
Gebeurt via de sql query die alle velden en veldwaarden ophaalt.
Zie de technische structuur.

6 Klassen

In onderstaand overzicht zijn de relaties tussen de verschillende lagen te zien. De klassen in de BusinessLayer zijn in het tweede overzicht weergegeven. Alle attributen en methodes van de klassen zijn tevens zichtbaar. Zo is er duidelijk te zien welke methodes de klasse Melding bijvoorbeeld heeft.



Vier lagen model



Relaties tussen de klassen

7 Datasets

Om in ASP.NET data uit de database te halen, is het gebruikelijk deze data in een dataset te laden. Om dit proces gemakkelijk en overzichtelijk te houden wordt het databasecomponent van ETX gebruikt. Dit component handelt de connectie met de database af, het dient als doorgeefluik naar de database.

De naam van een dataset komt overeen met de tabel uit het datamodel die er in wordt opgeslagen. DataSets worden gevuld door bijbehorende DataAdapters.

De volgende datasets zijn gemaakt:
- MeldingenDataSet
- GebruikerDataSet
- GebruikerRolDataSet
- KlantDataSet
- RolDataSet
- TaakRolKlantDataSet
- StatusDataSet
- KeuzeDataSet
- GemaakteKeuzeDataSet
- WorkFlowDataSet
- SoortDataSet
- VeldenDataSet
- VeldwaardenDataSet
- WaardePermissieDataSet

Om overzichten te kunnen maken, zijn er gegevens nodig uit verschillende tabellen. Om zo'n overzicht makkelijk te kunnen genereren zijn er een aantal DataSets specifiek voor een overzicht, dit zijn:

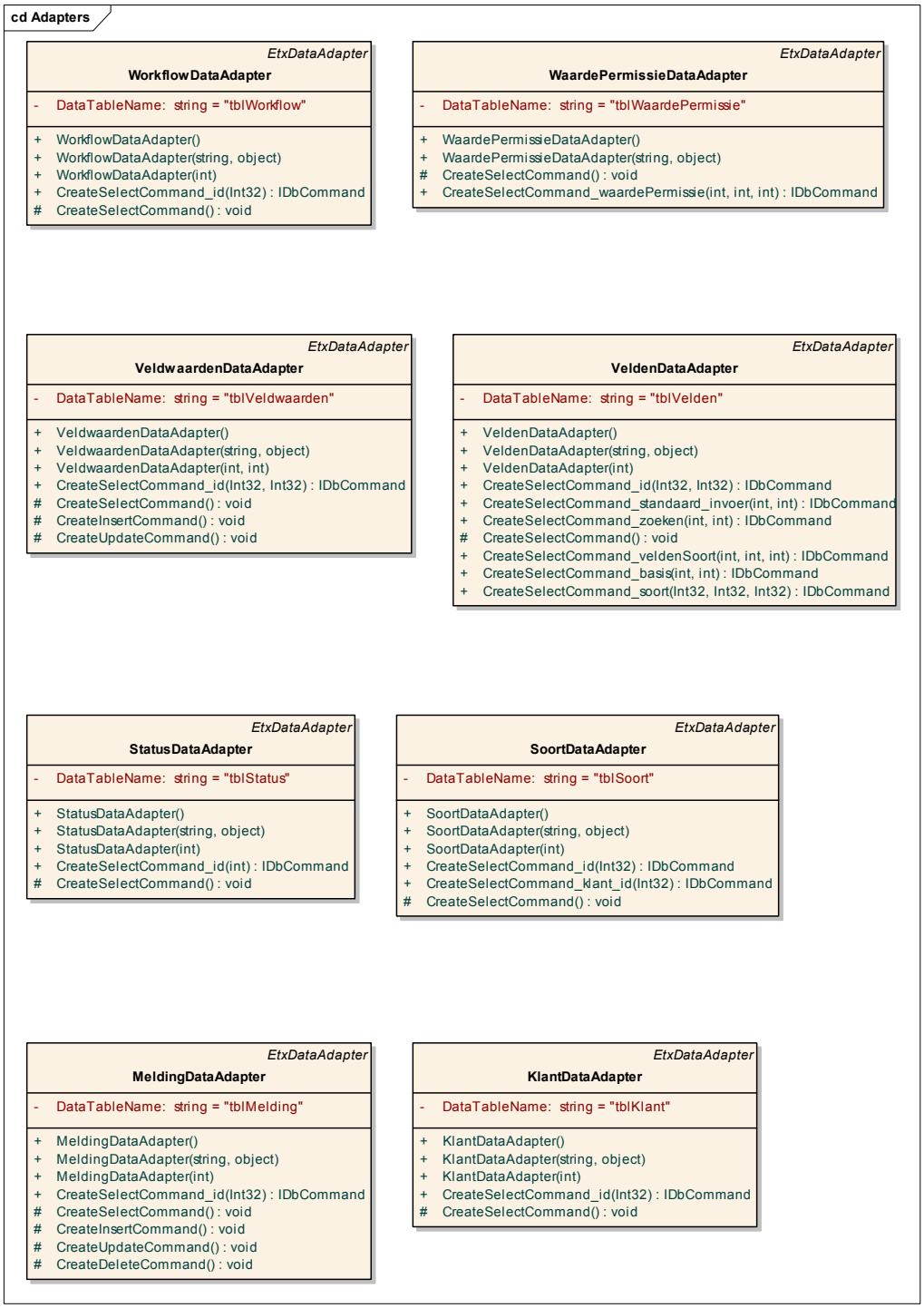
- MeldingOverzichtDataSet
  Nodig voor het overzicht van alle meldingen met daarin de status naam en de soort naam.
- VeldwaardenOverzichtDataSet
  Nodig voor het overzicht van alle meldingen. In deze dataset staan de veldwaarden van alle meldingen van één veld (bijvoorbeeld alleen veld "naam"). In deze DataSet is melding_id primary key gemaakt om bij het generen van het overzicht de juiste waarde bij de juiste melding te kunnen matchen.
- GemaakteKeuzeNaamDataSet
  Nodig voor het tonen van een overzicht van alle gemaakte keuzes bij een melding met de naam en omschrijving van deze keuze. Deze dataset bevat dus gegevens van zowel de tabel GemaakteKeuze, Keuze en Gebruiker.

8 Data Adapters

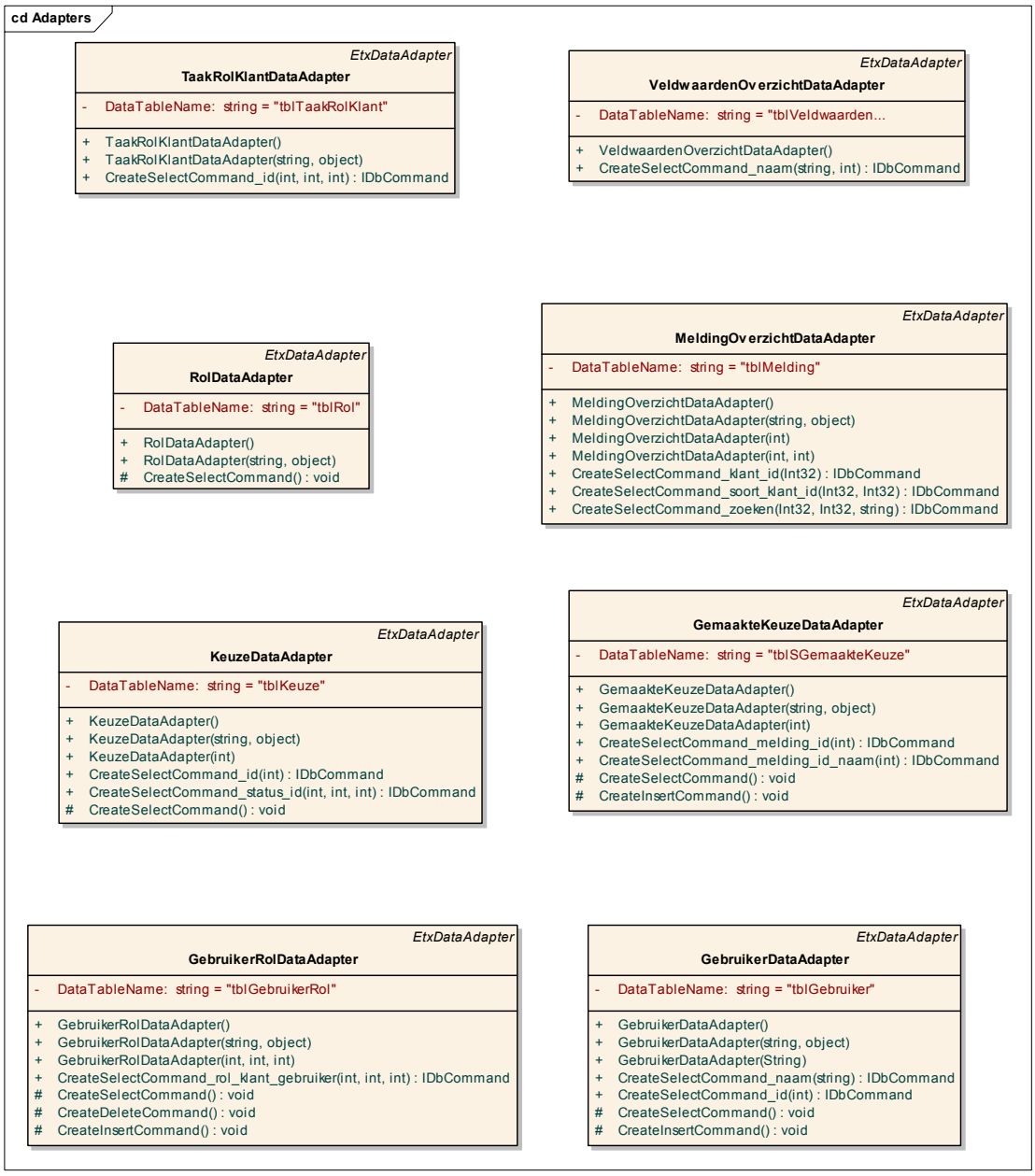
Een dataset wordt gevuld vanuit een DataAdapter. Hieronder volgen de Data Adapters met hun attributen en operaties.

Het attribuut “DataTableName: string = “ geeft steeds weer bij welke tabel de Data Adapter hoort.

Elke adapter heeft minstens één select command. Sommige adapters hebben er meerdere om verschillende selects uit te kunnen voeren.
Waar nodig hebben de adapters ook een update of delete command.



Deel één van de adapters met hun attributen en operaties.



Deel twee van de adapters met hun attributen en operaties.

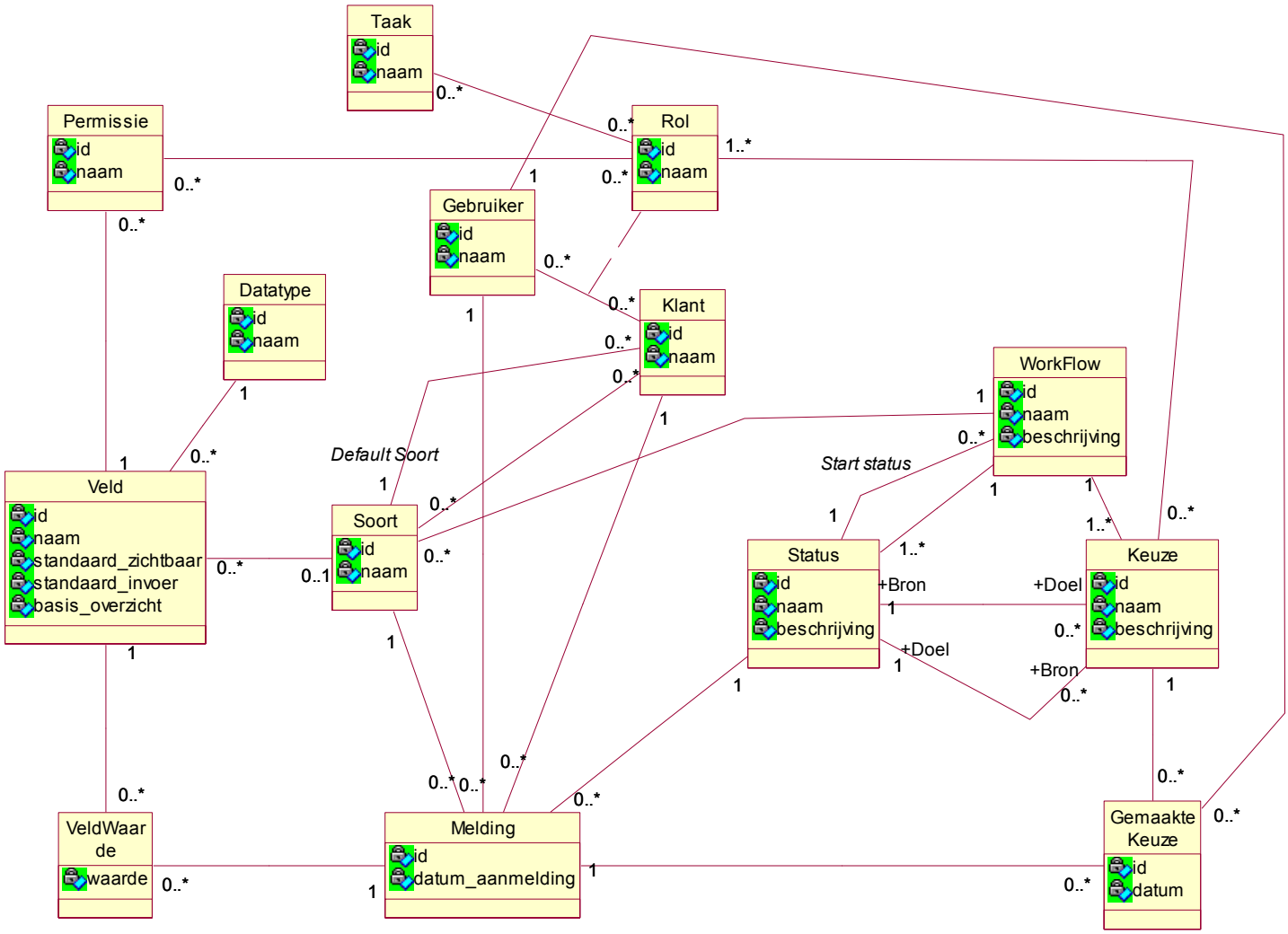
7. Technische structuur

1 Inleiding

In dit document komt de technische structuur van de tweede pilot ter sprake. Als eerste wordt het datamodel besproken, daarna komt de configuratie van de Webpart binnen Sharepoint ter sprake. Tot slot wordt kort besproken hoe de autorisatie in zijn werk gaat.

2 Datamodel

In het volgende overzicht is het datamodel weergegevens met de attributen en relaties tussen de klassen.



Hieronder volgt een uitleg van dit model.

Workflow:
In bovenstaand model kunnen verschillende *workflows* opgeslagen worden. Elke *workflow* kent een aantal *statussen*, en elke *status* kent mogelijke *keuzes*. Elke *keuze* kent weer een *status* waar je naar toe gaat als je die keuze maakt.

Er is gekozen voor de klasse *soort* omdat de workflow per soort melding (fout, wijzigingsverzoek etc) wordt opgeslagen. Elke *soort* kan een andere *workflow*, hebben, maar meerdere soorten kunnen ook verwijzen naar dezelfde workflow. Een *klant* kan weer verschillende *soorten* hebben. Het is niet wenselijk dat voor elk *klant* weer de gehele *workflow* aangemaakt moet worden. Door de klant te koppelen met een bestaand *soort*, wordt een reeds bestaande *workflow* gevolgd.
Mocht je toch een aparte *workflow* willen voor de *klant*, dien je een nieuw *soort* aan te maken en dit te koppelen met de *klant*. Voor dit *soort* kan je dan een andere *workflow* invoeren.

Er wordt door de koppeling van *melding* en *klant* bijgehouden welke melding bij welke klant hoort. Op deze manier is nog alle meldingen van een klant op te vragen.

De historie van gemaakte keuzes van een *melding* wordt bijgehouden bij *GemaakteKeuze*.

Melding:
Een *melding* heeft een naam, omschrijving, etc. Deze worden echter niet onder *melding* opgeslagen. Een *melding* is namelijk van een bepaald *soort*, bij een bepaald *soort* horen *velden*, zoals naam, omschrijving, prijs etc. Deze *velden* worden opgeslagen in *velden*.
Er is een standaard set met *velden* die voor elke *melding* zullen gelden. Extra *velden* kunnen vervolgens makkelijk toegevoegd worden.

Bij het invoeren van een *melding* bij een bepaalde *klant*, zal de *melding* van het *soort* worden dat het *default soort* is van de *klant*.

De gegevens van zo'n *veld*, dus de werkelijke naam, omschrijving etc worden opgeslagen bij *Veldwaarden*. Door de koppeling met *velden* en *melding* is bekend wat de waarde representeert (welk *veld*), en bij welke *melding* het hoort.
Van een *veld* wordt het *datatype* opgeslagen.

Er is gekozen voor deze oplossing omdat op deze manier er via de database extra *velden* kunnen worden toegevoegd aan *meldingen* van een bepaald *soort*. Dit was een eis vanuit de opdrachtgever.
Ook wilde de opdrachtgever per *veld* kunnen aangeven wat de rechten zijn van een bepaalde *rol*, ook dat is op deze manier makkelijk op te slaan door de koppeling met *permissie*.

Een melding hoort altijd bij een *klant*. Een *klant* kan verschillende *soorten* hebben.

Gebruikers:
Van *gebruikers* wordt de naam bijgehouden. Ook hebben *gebruikers* per *klant* een *rol*. Aan de *rol* zitten *permissies* gekoppeld, een *permissie* bepaald wat je rechten zijn op een bepaald *veld*. De mogelijke permissies zijn, niet zichtbaar, zichtbaar en zichtbaar en wijzigbaar.
Een rol heeft tevens taken, dit zijn de taken die hij mag uitvoeren in het systeem.

Met betrekking tot de workflow mogen bepaalde keuzes enkel uitgevoerd worden door sommige rollen. Van de gemaakte keuze wordt de gebruiker bijgehouden.

3 Relationeel Representatiemodel

Hieronder is het relationele representatiemodel te vinden, dit is afgeleid van het klassendiagram.

Workflow (**id**, Naam, beschrijving, *start_status_id*)

- Start_status_id is FK, verwijst naar id in status, null is niet toegestaan

Status (**id**, *workflow_id*, naam, beschrijving)

- workflow_id is FK, verwijst naar id in workflow, null is toegestaan

Keuze (**id**, *workflow_id*, naam, beschrijving, *status_id*, *doel_status_id*)

- workflow_id is FK, verwijst naar id in workflow, null is niet toegestaan
- status_id is FK, verwijst naar id in status, null is niet toegestaan
- doel_status_id is FK, verwijst naar id in status, null is niet toegestaan

Melding (**id**, datum_aanmelding, *soort_id*, *status_id, klant_id, gebruiker_id, project_id*)

- soort_id is FK, verwijst naar id in soort, null is niet toegestaan
- status_id is FK, verwijst naar id in status, null is niet toegestaan
- klant_id is FK, verwijst naar id in klant, null is niet toegestaan
- gebruiker_id is FK, verwijst naar id in gebruiker, null is niet toegestaan
- project_id is FK, verwijst naar id in project, null is toegestaan.

GemaakteKeuze (**id**, datum, *melding_id*, *keuze_id, gebruiker_id*)

- melding_id is FK, verwijst naar id in melding, null is niet toegestaan
- keuze_id is FK, verwijst naar id in keuze, null is toegestaan
- gebruiker_id is FK, verwijst naar id in gebruiker, null is niet toegestaan

Soort (**id**, naam, *workflow_id*)

- workflow_id is FK, verwijst naar id in workflow, null is niet toegestaan

SoortKlant (***klant_id, soort_id***)

- klant_id is FK, verwijst naar id in klant, null is niet toegestaan
- soort_id is FK, verwijst naar id in soort, null is niet toegestaan

Klant (**id**, naam, beschrijving, *default_soort_id*)
- default_soort_id is FK, verwijst naar id in soort, null is niet toegestaan

Project (**id**, naam, *klant_id*)
- klant_id is FK, verwijst naar id in klant, null is niet toegestaan

Rol (**id**, naam)

Taak (**id**, naam)

RolTaak (***rol_id***, ***taak_id***)
- rol_id is FK, verwijst naar id in rol, null is niet toegestaan
- taak_id is FK, verwijst naar id in taak, null is niet toegestaan

Gebruiker (**id**, naam)

GebruikerRol (***rol_id***, ***klant_id, gebruiker_id***)
- rol_id is FK, verwijst naar id in rol, null is niet toegestaan
- klant_id is FK, verwijst naar id in klant, null is niet toegestaan
- gebruiker_id is FK, verwijst naar id in gebruiker, null is niet toegestaan

KeuzeRol(***rol_id***, ***keuze_id***)
- rol_id is FK, verwijst naar id in rol, null is niet toegestaan
- keuze_id is FK, verwijst naar id in keuze, null is niet toegestaan

RolPermissie (***veld_id***, ***rol_id***, *permissie_id*)
- veld_id is FK, verwijst naar id in velden, null is toegestaan
- rol_id is FK, verwijst naar id in rol, null is toegestaan

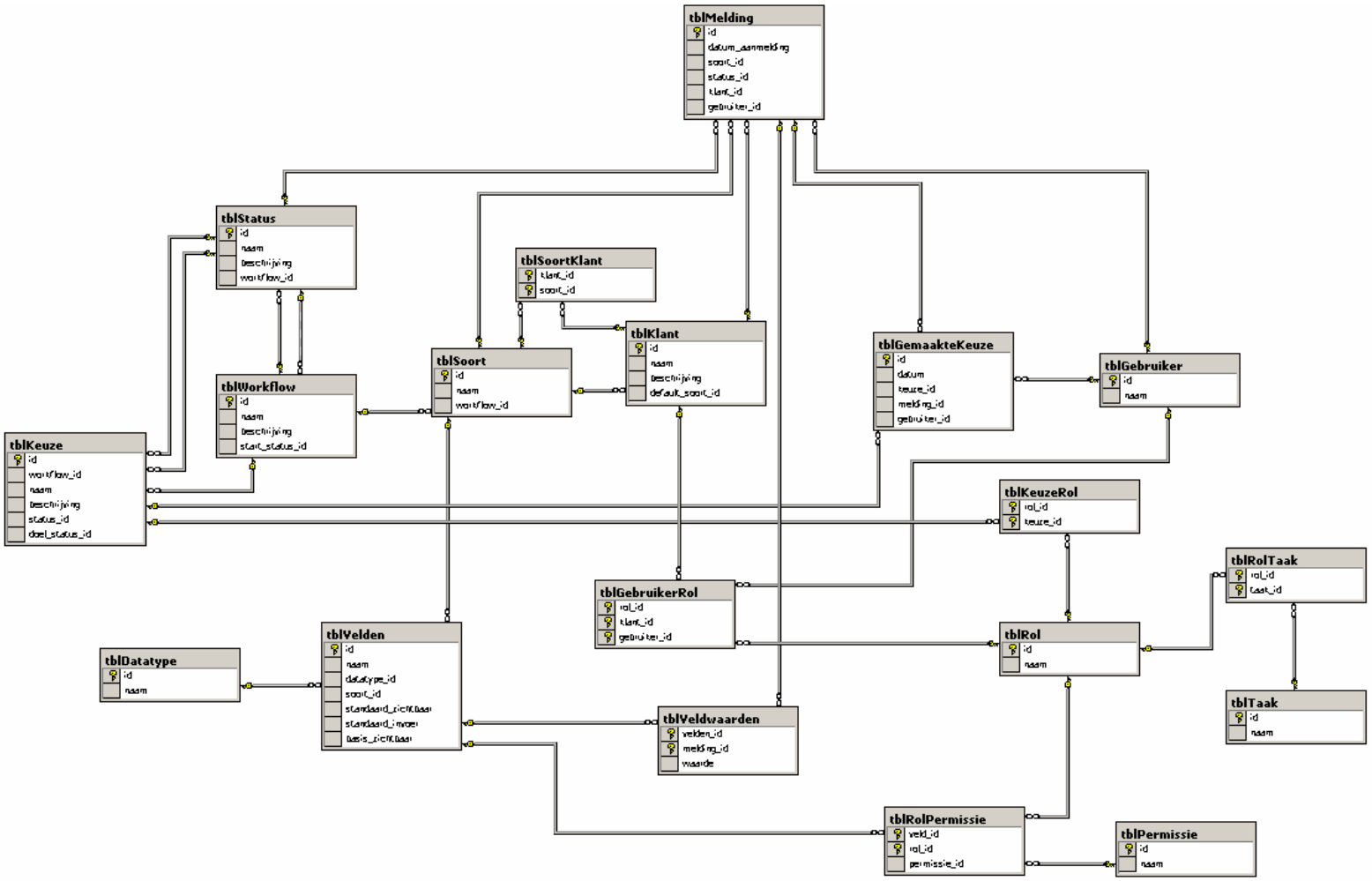
Permissie (**id**, naam)

Velden (**id**, naam, datatype, *soort_id*, standaard_zichtbaar, standaard_invoer, basis_overzicht)
- soort_id is FK, verwijst naar id in soort, null is toegestaan

Veldwaarden (***velden_id***, ***melding_id***, waarde)
- velden_id is FK, verwijst naar id in velden, null is niet toegestaan
- melding_id is FK, verwijst naar id in workflow, null is niet toegestaan

4 Tabellen

Het relationele representatie model resulteert na implementatie in de volgende tabellen.



4.1 Tabel Permissies

De volgende permissies zijn mogelijk:

1) Niet zien
2) Zien
3) Zien en wijzigen

4.2 Tabel Rollen

Standaard zijn er de volgende rollen:

1. Default User
2. Ontwikkelaar
3. Projectleider
4. Medewerker klant
5. Projectleider klant

4.3 Tabel Datatypes

1) Text
   Normale tekst, komt in een wijzig veld van 1 regel.
   Geen controle op de invoer.

2) Longtext
   Grote tekst, komt in een wijzig veld van meerdere regels.
   Geen controle op de invoer.

3) Mandatory
   Normale tekst, komt in een wijzig veld van 1 regel.
   Invoer van dit veld is verplicht.

4) Mandatorylongtext
   Grote tekst, komt in een wijzig veld van meerdere regels.
   Invoer van dit veld is verplicht.

5) Int
   Getal, komt in een wijzig veld van 1 regel.
   Invoer van dit veld moet een getal zijn.

4.4 Tabel Taken

De volgende taken staan standaard in het systeem:

1) Meldingen overzicht opvragen
2) Nieuwe melding invoeren
3) Gegevens melding opvragen
4) Gegevens melding wijzigen
5) Melding verwijderen
6) Trefwoord zoeken
7) Geavanceerd zoeken
8) Soort wijzigen
9) Workflow stap maken
10) Koppelen gebruiker – rol
11) Historie bekijken

5 Sharepoint / Webpart

Het systeem draait als webpart binnen de Sharepoint server. Er waren een aantal implementatie mogelijkheden:

- Pure webpart, nadeel: ontwerp van GUI is tijdrovend. Visual Studio.Net kent geen GUI ontwerp hulpmiddelen voor webparts.
- Smartpart: webpart is een web user control, GUI ontwikkeling gaat nu veel sneller.
- Smartpart Webpart maken van lijst met alle meldingen, met zoek/sorteer/filter optie, de andere schermen als externe pagina (web applicatie).
- Page Viewer Webpart, dan kan je een gewone webpage maken en via dit webpart kan je het dan bekijken als webpart.

Er is gekozen voor de 3e optie. Op die manier krijg je wel een echt webpart, dat geïntegreerd kan worden in de Sharepoint portal. De webpart zal een lijst laten zien van alle meldingen.
In de webpart zitten hyperlinks die toegang geven tot de andere functies. Deze andere functies zijn ondergebracht in andere schermen. Deze draaien als webapplicatie op de Sharepoint portal en zullen openen in een nieuw venster. Het voordeel hiervan is dat zowel de webpart als de webapplicatie gebouwd kan worden met behulp van Visual Studio.NET en de Visual Studio.NET GUI hulpmiddelen.

De webpart maakt gebruik van het Smartpart Webpart. Door middel van dit webpart is het mogelijk om een webpart te bouwen als Web User Control. De GUI van een web user control kan, in tegenstelling tot bij een webpart, worden gemaakt in Visual Studio.NET. Dit levert dus tijdwinst op.

In de bijlage is de licentie overeenkomst van Smartpart te vinden. Het is vrij voor commercieel gebruik.

5.1 Custom Properties

Het is bij een webpart mogelijk om custom properties aan te geven. Deze kunnen aangepast worden via de Sharepoint Portal. Op die manier is het mogelijk dat elke klant een eigen versie van de webpart krijgt. Bij het custom propertie van de webpart geef je dan het klant_id mee, zodat enkel meldingen van die klant in de webpart komen.

Hieronder volgt de code die gebruikt is voor het aanmaken van een custom property:

```
[System.ComponentModel.Browsable(true),
System.ComponentModel.Description("klant_id van de klant!")]
            public string klant_id_sharepoint
            {
                        get { return _klant_id_sharepoint; }
                  set { _klant_id_sharepoint = value; }
            }
```

6 Autorisatie

Binnen de applicatie zijn verschillende autorisatie onderdelen. Deze worden in dit hoofdstuk behandeld.

6.1 Taak Autorisatie

Onder taak autorisatie valt het autoriseren van het uitvoeren van bepaalde taken. Dit kan het openen van een pagina zijn of het laten zien van een bepaald element op een pagina.

De taak autorisatie is als volgt geïmplementeerd:
Gebruikers zijn gekoppeld aan rollen (per klant), rollen mogen taken uitvoeren.
Er is een functie geschreven die een true (toegang) of false (geen toegang) terug geeft. De invoer van deze functie is het taaknummer en klantID.
Aan de hand van de uitvoer van deze functie (true of false) wordt bepaald of de taak mag worden uitgevoerd door de gebruiker.

New gebruiker("bci2000\michielp", klantID)
Gebruiker.TaakAutorisatie(1, klantID);

In de ASPX pagina's zullen de taaknummers hard gecodeerd worden.

Het klantnummer zal meegegeven worden aan de constructor van het gebruiker object. Hierdoor is het mogelijk dat elke klant standaard default user is. De gebruiker zal namelijk aan de rol default user gekoppeld worden voor deze klant.

6.2 Gegevens Autorisatie

Onder gegevens autorisatie valt het autoriseren van het bekijken of wijzigen van gegevens. Er zijn drie mogelijkheden:

1. Niet zien
2. Zien
3. Zien en wijzigen

- De gegevens staan in de tabel Veldwaarden.
- De velden staan in de tabel Velden
- De mogelijke permissies (zie boven) staan in de tabel Permissie
- De permissie die een rol heeft op een veld staan in RolPermissie

Met behulp van de gebruikersnaam is de rol op te vragen. Via de rol zijn de permissies op te vragen.

In de ASPX pagina en de stored procedures zijn de permissies (1, 2 en 3) hard gecodeerd worden.

6.3 Workflow Autorisatie

De workflow kent statussen en keuzes. Niet iedereen mag elke keuze maken.
Een keuze is gekoppeld aan een rol.

Aan de SQL query die de mogelijke keuzes opvraagt wordt de gebruikersnaam meegegeven. De query zorgt er voor dat er enkel de keuzes die je mag uitvoeren zichtbaar worden.

De stored procedure "spKeuze_sel_status_id" is verantwoordelijk voor de workflow autorisatie.

7 Bijlage: Smartpart License

SmartPart for SharePoint: License
(bron: http://www.gotdotnet.com/workspaces/License.aspx?id=6cfaabc8-db4d-41c3-8a88-3f974a7d0abe)

Shared Source License for SmartPart for SharePoint

This license governs use of the accompanying software ('Software'), and your use of the Software constitutes acceptance of this license.

You may use the Software for any commercial or noncommercial purpose, including distributing derivative works.

In return, we simply require that you agree:
1. Not to remove any copyright or other notices from the Software.
2. That if you distribute the Software in source code form you do so only under this license (i.e. you must include a complete copy of this license with your distribution), and if you distribute the Software solely in object form you only do so under a license that complies with this license.
3. That the Software comes "as is", with no warranties. None whatsoever. This means no express, implied or statutory warranty, including without limitation, warranties of merchantability or fitness for a particular purpose or any warranty of title or non-infringement. Also, you must pass this disclaimer on whenever you distribute the Software or derivative works.
4. That no contributor to the Software will be liable for any of those types of damages known as indirect, special, consequential, or incidental related to the Software or this license, to the maximum extent the law permits, no matter what legal theory it’s based on. Also, you must pass this limitation of liability on whenever you distribute the Software or derivative works.
5. That if you sue anyone over patents that you think may apply to the Software for a person's use of the Software, your license to the Software ends automatically.
6. That the patent rights, if any, granted in this license only apply to the Software, not to any derivative works you make.
7. That the Software is subject to U.S. export jurisdiction at the time it is licensed to you, and it may be subject to additional export or import laws in other places. You agree to comply with all such laws and regulations that may apply to the Software after delivery of the software to you.
8. That if you are an agency of the U.S. Government, (i) Software provided pursuant to a solicitation issued on or after December 1, 1995, is provided with the commercial license rights set forth in this license, and (ii) Software provided pursuant to a solicitation issued prior to December 1, 1995, is provided with “Restricted Rights” as set forth in FAR, 48 C.F.R. 52.227-14 (June 1987) or DFAR, 48 C.F.R. 252.227-7013 (Oct 1988), as applicable.
9. That your rights under this License end automatically if you breach it in any way.
10. That all rights not expressly granted to you in this license are reserved.

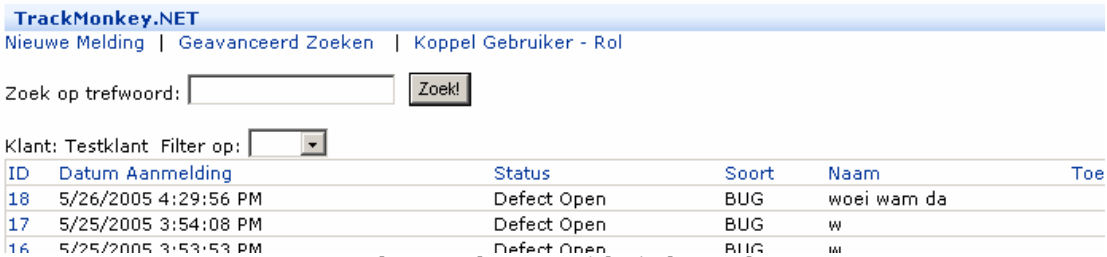
8. Gebruikers handleiding

1 Inleiding

Dit is de gebruikers handleiding die hoort bij pilot twee. Deze handleiding is bedoeld voor gebruikers die meer te weten willen komen over de functionaliteit van de webpart.

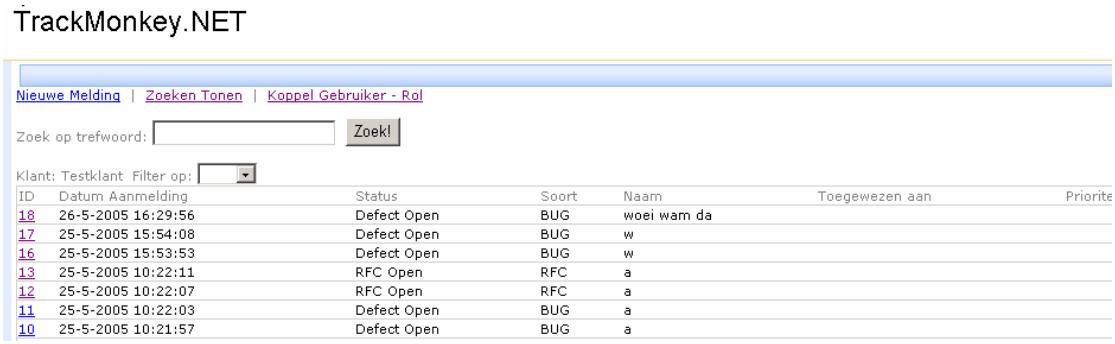
Aan de hand van screenshots wordt de functionaliteit op elke pagina besproken.

2 Overzicht



Screenshot van het overzicht in het webpart

Door op de titel van het webpart te klikken (TrackMonkey.NET) kan de webpart in een nieuw scherm worden geopend.



Screenshot van het overzicht in een volledige pagina

Dit is het openingsscherm van de applicatie.

Afhankelijk van uw rol in het systeem zijn er een aantal knoppen beschikbaar aan de bovenkant van het scherm:

- Nieuwe melding
  Verwijst naar een pagina waar een nieuwe melding in te voeren is.
- Geavanceerd Zoeken
  Opent het zoekscherm
- Koppelen – Gebruiker Rol
  Opent het scherm waar de beheerder gebruikers aan rollen kan koppelen.

In het overzicht zijn alle meldingen te vinden. De meldingen zijn te sorteren door op een kolomnaam te klikken. Door nogmaals op de kolomnaam te klikken worden ze andersom gesorteerd.
Als er meer dan 10 meldingen in het overzicht staan, zijn deze verdeeld over meerdere pagina's. Door op een pagina nummer te klikken onder het overzicht, kan de volgende pagina bekeken worden.

Wanneer er meerdere soorten meldingen beschikbaar zijn is het mogelijk om meldingen te filteren. Achter "Filter op: " staat een drop down box. Wanneer hierin een soort wordt geselecteerd, worden enkel meldingen van dat soort getoond.

Na het klikken op het nummer in de ID kolom van een melding, kunnen de gegevens van de melding bekeken worden.

3 Gegevens melding bekijken

Screenshot van de pagina "gegevens melding bekijken"

Deze pagina wordt zichtbaar nadat er op het nummer van een melding is geklikt in het overzicht.

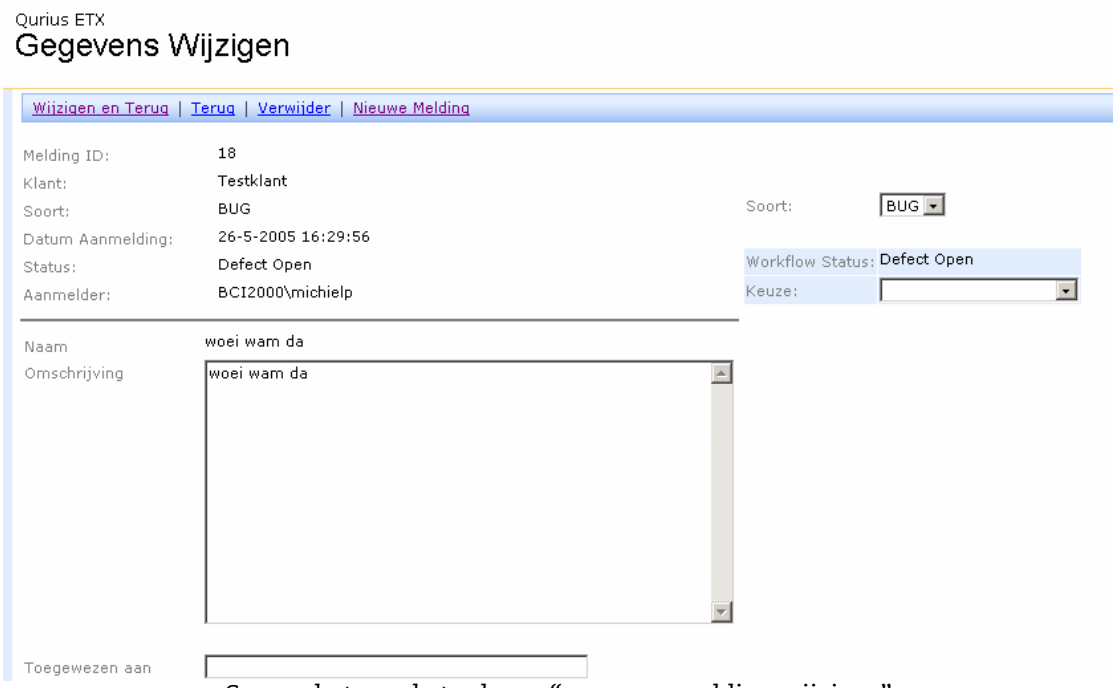
Afhankelijk van uw rol in het systeem zijn er een aantal knoppen beschikbaar aan de bovenkant van het scherm:

- Wijzig
  Door te klikken op deze knop kunnen de gegevens gewijzigd worden.
- Terug
  Terug naar het overzicht.
- Verwijder
  Deze melding verwijderen (er volgt eerst een bevestig scherm).
- Nieuwe Melding
  Een nieuwe melding invoeren

Onder deze knoppen zijn de gegevens van de melding te vinden. Eerst volgen enkele algemene gegevens, zoals het melding nummer, de status en de aanmelder die de melding heeft aangemeld.
Na de horizontale streep volgen gegevens zoals de naam en de omschrijving van de melding. De hoeveelheid gegevens die hier zichtbaar zijn hangt af van uw rol in het systeem.

Aan de rechterkant van het scherm is de historie te zien. Dit is de historie van stappen in de workflow. In het overzicht staat welke stap is genomen, wanneer en door wie.

4 Gegevens melding wijzigen



Screenshot van het scherm "gegevens melding wijzigen"

Dit scherm wordt getoond nadat er op een "wijzig" knop wordt geklikt bij het bekijken van gegevens over een melding.

Afhankelijk van uw rol in het systeem zijn er een aantal knoppen beschikbaar aan de bovenkant van het scherm:

- Wijzigen en Terug
  Bevestig de wijzigingen die gemaakt zijn in het scherm en sla ze op.
- Terug
  Terug naar het overzicht.
- Verwijder
  Deze melding verwijderen (er volgt eerst een bevestig scherm).
- Nieuwe Melding
  Een nieuwe melding invoeren

De algemene gegevens over een melding worden onder de knoppen getoond. Dit zijn het melding nummer, de datum van aanmelding en de aanmelder. Deze zijn niet wijzigbaar.

Onder de horizontale streep zijn de gegevens van de melding wijzigbaar. Afhankelijk van de rol in het systeem zullen velden wijzigbaar of niet wijzigbaar zijn. In bovenstaand screenshot is de "naam" niet wijzigbaar, maar de omschrijving wel.

Bij sommige gegevens mogen enkel getallen worden ingevoerd. Wanneer dit niet gebeurt zal er een melding worden getoond. In dat geval dient er een getal ingevoerd te worden, dit mag een negatief of positief getal zijn. Er mag één punt of komma in voorkomen.

Nadat er veranderingen zijn gemaakt en op de "Wijzigen en Opslaan" knop is gedrukt, zullen de wijzigingen worden opgeslagen.

4.1 Soort wijzigen

Afhankelijk van de rol in het systeem, is het soort van de melding te wijzigen.
Aan de rechterkant van het scherm is een drop down box te zien met daarin het huidige soort. In de drop down box staan de andere beschikbare soorten. Deze kunnen geselecteerd worden.
Pas nadat er op de “Wijzigen en Terug” knop is gedrukt, zal het nieuwe soort worden opgeslagen.

Als het soort wordt gewijzigd zal de melding automatisch in de start status van een nieuwe workflow komen. Een stap in de workflow maken heeft dus geen effect als het soort wordt gewijzigd.

4.2 Workflow stap maken

Afhankelijk van de rol in het systeem, kan er een stap worden gemaakt in de workflow.

Aan de rechterkant, onder de dropdown box met het huidige soort, staat de huidige status van de melding. Daaronder staan de keuzes die beschikbaar zijn. Afhankelijk van de rol in het systeem zijn er meer of minder keuzes beschikbaar.

Er kan één keuze geselecteerd worden.

Pas nadat er op de “Wijzigen en Terug” knop is gedrukt, zal de keuze worden gemaakt en worden opgeslagen.
Van elke keuze wordt opgeslagen wie en wanneer de keuze is gemaakt.

5 Melding verwijderen

Screenshot van de pagina waar meldingen verwijderd kunnen worden.

Deze pagina wordt getoond als er op een “Verwijder” knop is geklikt.

Afhankelijk van uw rol in het systeem zijn er een aantal knoppen beschikbaar aan de bovenkant van het scherm:

- Verwijder!
  Verwijder de melding definitief.
- Terug
  Terug naar het overzicht met de gegevens over de melding.

Op deze pagina wordt u om bevestiging van de verwijdering gevraagd. Voor de juistheid worden de gegevens van de melding op het scherm getoond.

Na het klikken op de “Verwijder!” knop zal de melding definitief worden verwijderd. Dit zal bevestigd worden met een melding.

6 Nieuwe melding

Invoeren nieuwe melding

Dit scherm wordt getoond als er een nieuwe melding ingevoerd mag worden.

Afhankelijk van uw rol in het systeem zijn er een aantal knoppen beschikbaar aan de bovenkant van het scherm:

- Opslaan en Sluiten
  Na invoering worden van de gegevens kunnen deze door middel van deze knop worden opgeslagen.
- Terug naar webpart
  Deze link leidt terug naar het webpart.

Bij het invoeren van een nieuwe melding dient er een naam en omschrijving meegegeven te worden. Deze velden zijn verplicht. Als ze niet ingevuld worden, verschijnt er een melding dat er iets ingevuld dient te worden.

Na het klikken op de knop “Opslaan en Sluiten” wordt de melding verstuurd.

7 Zoeken

Er kan gezocht worden op trefwoord of op elk veld afzonderlijk.

7.1 Op trefwoord

Vanuit het overzicht kan er meteen op trefwoord worden gezocht.

Zoeken op trefwoord

Na het intikken van een trefwoord en het klikken op Zoek! Zal er in alle velden worden gezocht naar deze waarde, dus zowel in tekstvelden als in getal velden.

De resultaten kunnen gefilterd worden op soort en gesorteerd worden op elke kolom.

7.2 Geavanceerd Zoeken

Screenshot van het zoekscherm

Na het klikken op “Geavanceerd zoeken” zal het zoekscherm zichtbaar worden. Als het geavanceerde zoekscherm open staat, zal het zoeken op trefwoord worden verborgen. Het zoeken op trefwoord kan zichtbaar worden gemaakt door het geavanceerde zoeken te verbergen. Dit kan door op “Zoeken Verbergen” te klikken.

In het zoekscherm kan gezocht worden op verschillende velden. Afhankelijk van uw rol binnen het systeem zijn deze velden beschikbaar.

De volgende velden zijn altijd beschikbaar:

- Nummer
- Status
- Datum

Onder de velden waarop gezocht kan worden staat de knop “Zoek!”. Na het klikken op deze knop, wordt er gezocht op alle velden die ingevuld zijn.

9. Installatie handleiding

1 Inleiding

Dit is de installatie handleiding die hoort bij pilot twee. Deze handleiding is bedoeld voor de installatie van de webpart op de Sharepoint Portal.
Eerst wordt de installatie op Sharepoint besproken, daarna wordt de database setup behandeld. Enige ervaring met Sharepoint is gewenst.

2 Installatie

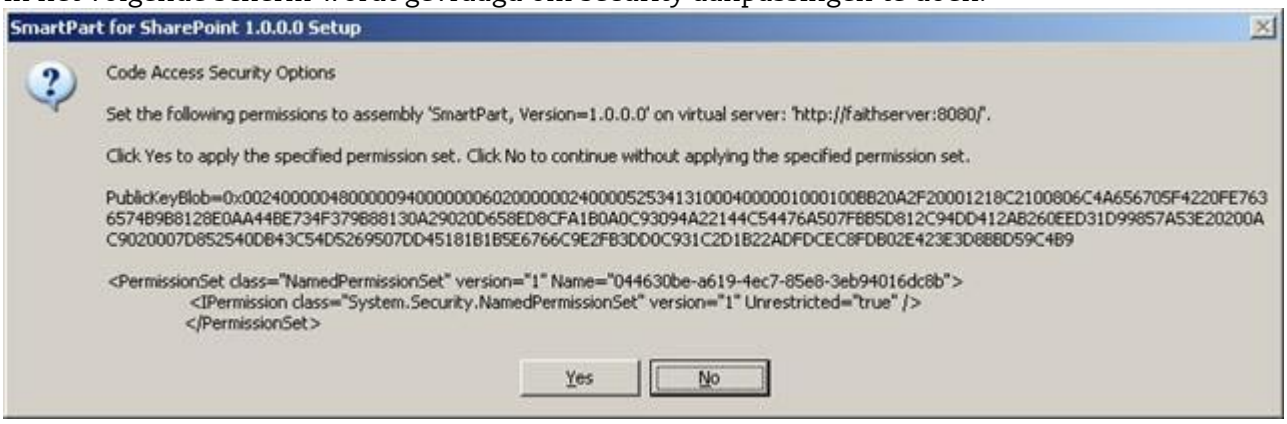
De installatie bestaat uit drie onderdelen. Eerst wordt het Smartpart geïnstalleerd. Vervolgens het webpart in het smartpart en ten slotte de webapplicatie.

2.1 Smartpart installatie

Installeer de Smartpart via “SmartPart for SharePoint 1.0.0.0.msi”.

Er wordt gevraagd of de SmartPart.dll in de GAC mag, antwoord ja.

In het volgende scherm wordt gevraagd om security aanpassingen te doen.



Antwoord NO.

Maak de volgende directories aan, als deze nog niet bestaan:

- Inetpub\wwwroot\bin
- Inetpub\wwwroot\UserControls

Kopieer het bestand “WorkFlowPart.ascx” naar de UserControls directory.
Kopieer alle bestanden uit de installatie\trackmonkey\bin\ directory naar de zojuist aangemaakte bin directory op de webserver.

Open inetpub\wwwroot\Web.config
Voeg het volgende toe voor de database toegang:

Binnen <configSections>:

```
      <sectionGroup name="ETX">
         <section name="Database"
type="System.Configuration.NameValueSectionHandler,system,
Version=1.0.3300.0, Culture=neutral, PublicKeyToken=b77a5c561934e089,
Custom=null" />
      </sectionGroup>
```

Na </configSections>:

```
      <ETX>
      <Database>
         <add key="ServerName" value="DATABASE_SERVER" />
         <add key="Catalog" value="DATABSE_NAAM" />
      </Database>
      </ETX>
```

Vervolgens dient het volgende aangepast te worden in Web.config:

De regel “<trust level="WSS_Medium" originUrl="" />” moet op Medium staan, niet op Minimal!

Om database toegang te laten werken moet het security bestand aangepast worden. Zoek de volgende regel:
<trustLevel name="WSS_Medium" policyFile="D:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\60\config\wss_mediumtrust.config" />

Zoek het bestand waar naar verwezen wordt op, in dit geval dus:
“Program Files\Common Files\Microsoft Shared\Web Server Extensions\60\config\ wss_mediumtrust.config”

Verander de volgende regel in het bestand:

```
<CodeGroup
        class="FirstMatchCodeGroup"
        version="1"
        PermissionSetName="FullTrust">
    <IMembershipCondition
            class="AllMembershipCondition"
            version="1"
    />
```

Ga vervolgens verder met de webpart installatie hieronder.

2.2 Webpart installatie

Open internet explorer en ga naar een Sharepoint pagina waar de smartpart dient komen te staan.

Vervolgens kan het smartpart op de Sharepoint pagina worden geplaatst door het smartpart te slepen uit de Virtual Server Gallery.

Klik op: modify page
Er verschijnt een drop down list, kies daaruit Add Web Parts > Browse

In de virtual server gallery zijn twee webparts beschikbaar, sleep de SmartPart List 1.0.0.0 naar de gewenste plek op de pagina. De smartpart zal getoond worden op de pagina. Klik op “open the tool pane” in de webpart.

Er verschijnt een configuratie scherm aan de rechterkant. Kies bovenaan, in de drop down list voor de WorkFlowPart. Klik op ok.

De configuratie van het webpart staat in het hoofdstuk “Webpart Configuratie”.

2.3 Webapplicatie installatie

- Ga naar de “Sharepoint Portal Server Central Administration”
- Klik op “Configure virtual server settings from the Virtual Server List page” onder “Portal Site and Virtual Server Configuration”.
- Kies de juiste virtual server.
- Klik op “Define Managed Paths”
- Voeg het path “/trackmonkey” toe aan de EXCLUDED PATHS

- Maak de map inetpub/wwwroot/trackmonkey aan
- Plaats alle bestanden uit de installatie directory in deze directory.

- Ga naar de IIS Manager bij de Administrative Tools van het Configuratie Scherm.
- Ga naar deze server > Web Sites.
- Kies vervolgens de juiste web server (Default Web Server)
- Maak een virtual directory aan genaamd trackmonkey en laat deze verwijzen naar de map inetpub/wwwroot/trackmonkey die je zojuist hebt aangemaakt.

2.4 Database Component

Het ETX Database component dient toegevoegd te worden aan de GAC, dit kan door het volgende uit te voeren voor de dlls:

Etx.Data.Database.dll

Registreren van ETX DLLs voor global assembly cache:
D:\WIN2003\Microsoft.NET\Framework\v1.1.4322>gacutil /i Etx.Security.Cryptography.IDataProtector.dll

3 Database Setup

Een algemene uitleg over het database-ontwerp is te vinden in het technisch ontwerp.

3.1 Tabellen structuur inladen

Met behulp van de SQL Project explorer kan het TrackMonkey.spe project geladen worden. In het TrackMonkey project staat het laatste sql bestand voor de tabellen structuur.
Na uitvoering van dit bestand zullen alle tabellen, relaties en stored procedures in de aangegeven database geladen zijn.

3.2 Tabellen Setup

Om de juiste gegevens in de tabellen te laden zijn er enkele setup stored procedures. Er is een verplicht en optioneel gedeelte dat uitgevoerd kan worden.

3.2.1 Verplicht

Dit gedeelte is verplicht, het zal o.a. de taken en de rollen in de tabellen laden.

Voer met behulp van de SQL Query Analyzer de volgende code uit:

```
EXEC spSetup_create_standaard_datatype
EXEC spSetup_create_standaard_rollen
EXEC spSetup_create_standaard_permissies
EXEC spSetup_create_standaard_taken_plus_rechten
```

3.2.2 Optioneel

Het optionele gedeelte laadt o.a. de standaard workflow voor een RFC en BUG in en zal de velden met rechten aanmaken.

```
EXEC spSetup_create_standaard_velden_ALGEMEEN_plus_rechten null
EXEC spSetup_create_soort_BUG_workflow_velden
EXEC spSetup_create_soort_RFC_workflow_velden
```

Vervolgens kunnen er klanten ingevoerd worden en gekoppeld worden met soorten. Zie hiervoor de beheer handleiding.

4 Webpart Configuratie

Bij de properties van de webpart kan het volgende aangegeven worden:

Appearance
Title: Gewenste title, bijvoorbeeld: TrackMonkey
Hoogte en breedte kunnen desgewenst ook nog aangepast worden.

Advanced
Detail Link:
/trackmonkey/TrackMonkey.aspx?klant_id=KLANTNR&soort_id=SOORTNR&basis_overzicht=*(1 voor ja en -1 voor nee)*&zoeken=*(1 voor automatisch zoekscherm tonen en -1 voor normaal venster)*

Custom properties
Er kunnen aan de webpart custom properties worden meegegeven om de webpart specifiek alleen meldingen te laten tonen van één klant.
De volgende custom properties kunnen worden meegegeven:

- Klant_id
  Kan veranderd worden in het klant nummer waarvoor de meldingen getoond moeten worden. Het nummer moet overeenkomen met het klant_id uit de database.
- Soort_id
  Standaard: -1
  Kan veranderd worden in een soort_id dat bij de klant hoort om enkel één soort melding te tonen in de webpart. Het nummer moet overeenkomen met soort_id uit de database tabel Soort en moet gekoppeld zijn met de klant in SoortKlant.
- Basis_overzicht
  Standaard: -1
  Wanneer deze op 1 staat zullen enkel de basis kolommen in het overzicht worden getoond. Welke basis kolommen worden getoond kan worden geconfigureerd in de database.

5 Web Applicatie Verplaatsen

De webapplicatie hoort in de directory:
Inetpub/wwwroot/trackmonkey

Dit vertaalt zich op de internetserver naar:
http://localhost/trackmonkey

Dit kan aangepast worden, maar er zullen dan ook wijzigingen in de code van de webpart plaats moeten vinden. Deze verwijst namelijk naar deze directory.

Uit de HTML code van WorkFlowPart.aspx:

```
<asp:HyperLinkColumn Target="_self" DataNavigateUrlField="id"
DataNavigateUrlFormatString="/trackmonkey/GegevensMelding.aspx?meldin
g_id={0}" DataTextField="id" SortExpression="id" HeaderText="ID">
```

Het vetgedrukte gedeelte zal aangepast moeten worden.

De volgende constante in WorkFlowPart.aspx moet ook aangepast worden:

```
private const string baseURL = "/trackmonkey/";
```

5.1 CSS

Voor de opmaak van de pagina's is een CSS bestand gebruikt dat standaard op de sharepoint server aanwezig is.

Alle pagina's verwijzen naar het CSS bestand op de volgende locatie:
http://server/_layouts/1033/styles/ows.css

10. Beheer handleiding

1 Inleiding

Dit is de beheer handleiding die hoort bij pilot twee. Deze handleiding is bedoeld voor de beheerder van de applicatie TrackMonkey.NET.

Doordat er bij de pilot zeer beperkte beheermogelijkheden zijn gemaakt, zal het beheren van de webpart voornamelijk gebeuren via de database. Daar kunnen bijvoorbeeld klanten en soorten worden aangemaakt.

Voor elke beheertaak zijn stored procedures gemaakt. Deze stored procedures kunnen gebruikt worden om de beheerstaken later makkelijk in een user interface te laten uitvoeren.

2 Beheer

Het beheer van de webpart gebeurt via de database, omdat er geen beheerinterface is. Voor gemakkelijk toegang tot de database kan bijvoorbeeld het programma Microsoft SQL Enterprise Manager / SQL Query analyzer gebruikt worden.

Een algemene uitleg over het database-ontwerp is te vinden in het technische structuur document.

Er worden hieronder een aantal veel voorkomende beheer taken besproken.

2.1 Klanten toevoegen of verwijderen

Een webpart wordt altijd geïnstalleerd per klant. Voor elke klant kunnen meldingen ingevoerd worden.
Een klant kan meerdere soorten meldingen hebben, die een eigen workflow volgen.
De koppeling hiertussen staat in de tabel SoortKlant.
De soort die standaard voor deze klant zal worden gebruikt wordt aangegeven door de waarde van “default_soort_id”.

2.1.1 Toevoegen nieuwe klant

Een nieuwe klant kan toegevoegd worden via de stored procedure: spKlant_ins
Deze stored procedure heeft de volgende gegevens nodig:

- Naam
  Naam van de klant, wordt weergegeven in de webpart
- Beschrijving
  Beschrijving van de klant
- Default_soort_id
  Het soort dat een melding standaard zal worden als er een melding wordt ingevoerd voor deze klant. Zie “soort toevoegen of verwijderen” voor het toevoegen en verwijderen van soorten.

De output van deze stored procedure is het klant id, dit kan ingevuld worden bij de configuratie van de webpart.

Vervolgens dient de klant aan bepaalde soorten gekoppeld te worden.
Ook dienen gebruikers rollen te krijgen voor deze nieuwe klant.

LET OP: er moet tenminste één gebruiker als projectleider worden aangemaakt via de database. Deze kan vervolgens andere mensen rollen geven.

2.1.2 Koppelen klant aan soorten

De soorten waar een klant aan gekoppeld is, bepaalt welke soorten meldingen er voor een klant afgehandeld worden. Het standaard soort staat bij klant aangegeven in default_soort_id.

LET OP: de klant moet ook gekoppeld zijn aan het default_soort_id in de tabel tblSoortKlant anders zijn de ingevoerde meldingen niet opvraagbaar. Dit gebeurt automatisch bij het toevoegen van een klant!

Het toevoegen van een koppeling gaat met de volgende stored procedure:

spSoortKlant_ins (klant_id, soort_id)

2.1.3 Verwijderen klant

Het verwijderen van een klant kan via de stored procedure: spKlant_del

LET OP: bij het verwijderen van een klant worden ook alle meldingen van de klant verwijderd. Ook alle koppelingen tussen de klant en de soort worden verwijderd. Van alle melding worden ook alle waarden verwijderd.

2.2 Velden bij een melding toevoegen of verwijderen

Voor het invoeren van de standaard velden, zoals opgeleverd bij het systeem, verwijs ik naar de installatie handleiding.

2.2.1 Invoeren nieuw Veld

Maak een nieuwe rij in de tabel Velden met de volgende gegevens:

- Id
  Unieke id
- Naam
  De naam die weergegeven zal worden in de overzichten.
- datatype_id
  Het datatype van het veld, zie de paragraaf datatype.
- soort_id
  Het soort waar het veld bij hoort.
  Als dit veld null is zal het zichtbaar zijn voor elke melding van elk soort.
  Voor het toevoegen of verwijderen van soorten zie de paragraaf “Soorten toevoegen of verwijderen”.
- standaard_zichtbaar
  Deze boolean (0 of 1) bepaalt of het veld standaard zichtbaar mag zijn in de overzichten met alle meldingen.
- standaard_invoer
  Deze boolean (0 of 1) bepaalt of dit veld zichtbaar mag zijn bij het invoeren van nieuwe meldingen.
  LET OP: deze kolom heeft enkel invloed op velden met het soort_id null.
- Basis_overzicht
  Deze boolean (0 of 1) bepaalt of dit veld zichtbaar mag zijn in het basisoverzicht (zie de configuratie van de webpart voor het basisoverzicht).

Deze gegevens kunnen ook als parameters worden meegegeven aan de stored procedure: spVelden_ins

2.2.2 Datatype

De volgende datatypes kunnen aangegeven worden

1) Text
   Normale tekst, komt in een wijzig veld van 1 regel.
   Geen controle op de invoer.

2) Longtext
   Grote tekst, komt in een wijzig veld van meerdere regels.
   Geen controle op de invoer.

3) Mandatory
   Normale tekst, komt in een wijzig veld van 1 regel.
   Invoer van dit veld is verplicht.

4) Mandatorylongtext
   Grote tekst, komt in een wijzig veld van meerdere regels.
   Invoer van dit veld is verplicht.

5) Int
   Getal, komt in een wijzig veld van 1 regel.
   Invoer van dit veld moet een getal zijn.

2.2.3 Velden verwijderen

Een veld verwijderen kan handmatig gebeuren of via de stored procedure: spVelden_del.

LET OP: bij het verwijderen van een veld zullen alle waardes die opgeslagen zijn bij meldingen over dat veld ook verwijderd worden! Ook alle permissies die op dit veld gelden zullen verwijderd worden.

2.3 Permissies toevoegen of verwijderen

Permissies gelden voor een bepaalde rol op een veld.

2.3.1 Toevoegen permissie

Een permissie toevoegen kan met de stored procedure: spRolPermissie_ins

Hieraan dient meegegeven te worden:

- veld_id
  Nummer van het veld dat is toegevoegd bij de velden.
- rol_id
  Nummer van de rol (zie onder)
- permissie_id
  Nummer van de permissie (zie onder)

2.3.2 Verwijderen permissie

Het verwijderen van een permissie zal automatisch gebeuren bij het verwijderen van een veld.

2.3.3 Permissies

De volgende permissies zijn mogelijk:

1) Niet zien
2) Zien
3) Zien en wijzigen

2.3.4 Rollen

Standaard zijn er de volgende rollen:

1. Default User
2. Ontwikkelaar
3. Projectleider
4. Medewerker klant
5. Projectleider klant

2.4 Soorten toevoegen of verwijderen

Een soort is altijd gekoppeld met een bepaalde workflow.
Meerdere klanten kunnen hetzelfde soort gebruiken.

2.4.1 Soort toevoegen

Een soort kan toegevoegd worden met de stored procedure: spSoort_ins

De volgende gegevens zijn nodig:

- Naam
  Naam van het soort
- Workflow_id
  Workflow die de soort moet volgen, verwijst naar het id in tblWorkflow

2.4.2 Soort verwijderen

Een soort kan verwijderd worden met de stored procedure: spSoort_del

LET OP: als het soort wordt verwijderd worden ook alle velden die bij dit soort horen verwijderd.
Een soort kan niet verwijderd worden als een klant nog verwijst naar dit soort.

2.5 Workflow

Een workflow hoort bij een soort melding.
De beginstatus van een workflow geeft aan wat de eerste status van een ingevoerde melding moet zijn.

De workflow kent statussen en keuzes. Keuzes mogen enkel gemaakt worden door bepaalde rollen die worden aangegeven in tblKeuzeRol.

De workflow kan het beste ingevoerd worden door een stored procedure. Een nieuwe workflow kan het beste worden ingevoerd door een bestaande stored procedure aan te passen aan de nieuwe workflow.

2.5.1 Invoeren standaard workflow

Er zijn twee stored procedures beschikbaar om een workflow in te voeren:

- `spSetup_create_standaard_bug_workflow_plus_rechten`
- `spSetup_create_standaard_rfc_workflow_plus_rechten`

Beide stored procedures geven het workflow_id als output. Dit kan gebruikt worden om een soort aan een workflow te koppelen.

De invoer voor beide stored procedures is:

- Naam
- omschrijving

2.5.2 Verwijderen workflow

Bij het verwijderen van een workflow worden ook alle statussen en keuzes verwijderd.

LET OP: een workflow kan niet verwijderd worden als er nog meldingen in een status in de workflow zitten.

3 Webpart

Custom properties
Er kunnen aan de webpart custom properties worden meegegeven om de webpart specifiek alleen meldingen te laten tonen van één klant.
De volgende custom properties kunnen worden meegegeven:

- Klant_id
  Standaard: -1
  Kan veranderd worden in het klantnummer waarvoor de meldingen getoond moeten worden. Het nummer moet overeenkomen met het klant_id uit de database.
- Soort_id
  Standaard: -1
  Kan veranderd worden in een soort_id dat bij de klant hoort om enkel één soort melding te tonen in de webpart. Het nummer moet overeenkomen met soort_id uit de database tabel Soort en moet gekoppeld zijn met de klant in SoortKlant.
- Basisoverzicht
  Standaard: -1
  Als deze aan staat (1) dan wordt in het overzicht enkel de basis kolommen weergegeven. Dit zijn:
  - ID
  - Datum aanmelding
  - Status
  - Soort
  - De rest van de basis kolommen wordt bepaald door de velden die op basis_overzicht (1) staan (zie het toevoegen van een nieuw veld).

4 Web Applicatie

De webapplicatie hoort in de directory:
Inetpub/wwwroot/trackmonkey

Dit vertaalt zich op de internetserver naar:
http://localhost/trackmonkey

Dit kan aangepast worden, maar er zullen dan ook wijzigingen in de code van de webpart plaats moeten vinden. Deze verwijst namelijk naar deze directory.

Uit de HTML code van WorkFlowPart.aspx:

```
<asp:HyperLinkColumn Target="_self" DataNavigateUrlField="id"
DataNavigateUrlFormatString="/trackmonkey/GegevensMelding.aspx?meldin
g_id={0}" DataTextField="id" SortExpression="id" HeaderText="ID">
```

Het vetgedrukte gedeelte zal aangepast moeten worden.

De volgende constante in WorkFlowPart.aspx moet ook aangepast worden:

```
private const string baseURL = "/trackmonkey/";
```

4.1 CSS

Voor de opmaak van de pagina's is een CSS bestand gebruikt dat standaard op de Sharepoint server aanwezig is.

Alle pagina's verwijzen naar het CSS bestand op de volgende locatie:
http://server/_layouts/1033/styles/ows.css

11. Testplan

1 Inleiding

Dit is het testplan van de tweede pilot. Dit testplan maakt het mogelijk om de tweede pilot gestructureerd te doorlopen en te testen. Eventuele opmerkingen kunnen aangegeven worden na elke stap.

Het testplan is bedoeld voor mijzelf en voor de opdrachtgever. Beiden moeten aan de hand van dit testplan de applicatie kunnen testen en controleren of het voldoet aan de functionele eisen die omschreven zijn in het systeem concept.

2 Testplan

Dit testplan is opgebouwd aan de hand van de use-cases. Aan de hand van opgestelde scenario's wordt alle functionaliteit van het systeem doorlopen.

2.1 Testdata

De volgende testdata zal worden gebruikt tijdens het uitvoeren van de test. Deze moeten ingevuld zijn als custom property bij de webpart. Zie hiervoor de paragraaf "Webpart configuratie".

Naam	Data	Beschrijving
Klant	Klant = 1	De acties worden getest voor klant nummer 1.
Project	Project_id = -1	Er wordt geen project gespecificeerd.
Soort	Soort_id = -1	Er wordt geen soort gespecificeerd.

2.2 Actoren

De volgende actoren hebben toegang tot het systeem:

1. Default user
2. Ontwikkelaar
3. Projectleider
4. Medewerker Klant
5. Projectleider Klant

Via de use-case "Koppelen gebruiker - rol" kan een gebruiker gekoppeld worden aan een rol. Hij zal dan de rol van die actor hebben en de bijbehorende use-cases kunnen uitvoeren.
Welke use-cases een actor mag uitvoeren staat in het systeem concept. In dit testplan zal elke use-case voor maximaal twee actoren worden getest, om het testplan bruikbaar te houden.

2.3 Overzicht meldingen opvragen

2.3.1 Scenario overzicht meldingen opvragen

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Ga naar de URL: http://ETXPC048/MySite/default.aspx	Overzicht van de eerste 10 meldingen.		
Klik op de "2".	Het tweede scherm met de meldingen.		
Klik op de "1"	Het eerste scherm met de eerste 10 meldingen.		
Klik op de "1"	Er gebeurt niks.		

2.3.2 Scenario overzicht meldingen opvragen per soort en sorteren

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Ga naar de URL: http://ETXPC048/MySite/default.aspx - Selecteer het soort “BUG” uit de drop down box.	Overzicht van de eerste 10 meldingen van het soort “BUG”. Er mogen geen andere soorten getoond worden.		
Sorteer de meldingen per naam door op de kolom “naam” te klikken.	Meldingen gesorteerd per naam. Alleen BUGS worden getoond.		
Klik op de “2”.	De volgende 10 resultaten, nog steeds gesorteerd per naam. Alleen BUGS worden getoond.		
Klik op de datum.	De meldingen zijn nu op datum geselecteerd. Je bent weer op de eerste pagina met resultaten. Alleen BUGS worden getoond.		

2.3.3 Scenario Controle Kolommen

Vooraf: zorg dat je de rol “projectleider” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Ga naar de URL: http://ETXPC048/MySite/default.aspx	De kolommen die getoond worden: • ID • Datum aanmelding • Status • Soort • Naam • Toegewezen • Prioriteit Enkel de kolommen met standaard_zichtbaar = 1, zichtbaar voor de projectleider en kolommen die horen bij een soort in het overzicht.		
Selecteer het soort “BUG” uit de drop down box.	De kolommen die getoond worden zijn: • ID • Datum aanmelding • Status • Soort • Naam • Toegewezen • Prioriteit Enkel de kolommen met standaard_zichtbaar = 1, zichtbaar voor de projectleider en kolommen die horen bij de soort BUG.		

2.3.4 Scenario Controle Kolommen (default user)

Vooraf: zorg dat je de rol “default user” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Ga naar de URL: http://ETXPC048/MySite/default.aspx	De kolommen die getoond worden: • ID • Datum aanmelding • Status • Soort • Naam Enkel de kolommen met standaard_zichtbaar = 1, zichtbaar voor de default user en kolommen die horen bij een soort in het overzicht.		
Selecteer het soort “BUG” uit de drop down box.	De kolommen die getoond worden zijn: • ID • Datum aanmelding • Status • Soort • Naam Enkel de kolommen met standaard_zichtbaar = 1, zichtbaar voor de projectleider en kolommen die horen bij de soort BUG.		

2.4 Koppelen Gebruiker – Rol

2.4.1 Scenario projectleider

Vooraf: zorg dat je de rol “projectleider” hebt.

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Klik op de link “Koppelen gebruiker - rol” in het webpart scherm.	Er wordt een scherm getoond met daarop de klantnaam, en een drop down box met alle gebruikers.		
Selecteer de naam bci2000\michielp uit de drop down box met gebruikers.	De mogelijke rollen worden weergegeven. • Default User • Ontwikkelaar • Projectleider • Medewerker klant • Projectleider Klant De rollen “Default User” en “Projectleider” zijn aangevinkt.		
Selecteer de rol ontwikkelaar en de-selecteer de rol Default user. Klik op Opslaan.	Het scherm wordt herladen.		
Klik op een willekeurige andere naam. Klik vervolgens weer op de naam bci2000\michielp	De mogelijke rollen worden weergegeven. De rollen Default User, Ontwikkelaar en Projectleider zijn aangevinkt.		

2.4.2 Scenario default user

Vooraf: zorg dat je de rol “default user” hebt.

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Ga naar de URL: http://ETXPC048/MySite/default.aspx	De link “koppelen gebruiker - rol” is niet zichtbaar		

2.5 Zoeken

2.5.1 Scenario Trefwoord Zoeken

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Ga naar de URL: http://ETXPC048/MySite/default.aspx	De webpart wordt getoond, met daarin een tekstvak en zoekbutton ernaast.		
Toets het trefwoord “test” in en klik op zoek.	De meldingen met het woord test in de naam of omschrijving worden getoond.		

2.5.2 Scenario Geavanceerd zoeken

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Ga naar de URL: http://ETXPC048/MySite/default.aspx - Klik op geavanceerd zoeken.	De zoek pagina wordt getoond. De velden waarop gezocht kan worden, worden opgebouwd aan de hand van de velden in de database die voor jou zichtbaar en/of wijzigbaar zijn.		
Toets het zoekwoord “test” in bij het veld naam.	De meldingen met het woord test in de naam of omschrijving worden getoond.		

2.6 Gegevens melding opvragen

2.6.1 Scenario gegevens melding opvragen

Vooraf: zorg dat je de rol “projectleider” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Klik op nummer 1 uit de lijst met meldingen.	De volgende algemene gegevens over een melding worden getoond: - Melding Id - Klant naam - Soort - Datum Aanmelding - Status - Aanmelder Onder de lijn worden de volgende velden met gegevens over de melding getoond: - naam - omschrijving - prijs - toegewezen - prioriteit De link “gegevens wijzigen” is zichtbaar. Het overzicht met de historie van workflow stappen is zichtbaar.		

2.6.2 Scenario gegevens melding opvragen (default user)

Vooraf: zorg dat je de rol “default user” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Klik op nummer 1 uit de lijst met meldingen.	De volgende algemene gegevens over een melding worden getoond: - Melding Id - Klant naam - Soort - Datum Aanmelding - Status - Aanmelder Onder de lijn worden de volgende velden getoond: - Naam - Omschrijving - Prijs De link “gegevens wijzigen” is niet zichtbaar. Het overzicht met de historie van workflow stappen is niet zichtbaar.		

2.7 Gegevens melding wijzigen

2.7.1 Scenario gegevens melding wijzigen

Vooraf: zorg dat je de rol “projectleider” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 1.	Toont de pagina met de gegevens in een wijzigbare vorm. De volgende velden zijn wijzigbaar: - naam - omschrijving - prijs - toegewezen - prioriteit De algemene gegevens boven de lijn zijn niet wijzigbaar.		
Wijzig de gegevens en druk op Opslaan en Verzenden.	Een overzicht van de nieuwe gegevens van de melding wordt getoond.		

2.7.2 Scenario gegevens melding wijzigen (ontwikkelaar)

Vooraf: zorg dat je de rol “ontwikkelaar” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 1.	Toont de gegevens op de pagina in een wijzigbare vorm. De ontwikkelaar heeft beperkte wijzigrechten. De andere velden worden getoond in normale vorm.		

2.7.3 Scenario geen invoer

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 61. - Maak alle velden leeg en klik op Wijzigen en Verzenden	Er wordt een melding getoond dat de titel en omschrijving van een melding niet leeg mogen zijn en dat er bij de prijs een getal ingevoerd moet worden.		

2.7.4 Scenario vreemde tekens invoer

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 61. - Schrijf bij de inhoud van een veld een aantal vreemde tekens zoals < ? <% %> “ ‘ / en \	Een overzicht van de nieuwe gegevens van de melding wordt getoond. Alle velden zijn correct geüpdate.		

2.8 Workflow stap maken

2.8.1 Scenario workflow stap maken

Vooraf: zorg dat je de rol “projectleider” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 1.	Naast de gegevens over de melding is staat de status: Defect Open. Daaronder is een drop down box te vinden met daarin de mogelijke keuzes (zie hoofdstuk workflow voor mogelijke keuzes).		
Maak een keuze in de workflow door op “Toekennen” te klikken en druk op opslaan en verzenden.	De gegevens van de melding worden getoond. De melding is nu in de status “In behandeling”.		

2.8.2 Scenario workflow stap maken (projectleider klant)

Vooraf: zorg dat je de rol “projectleider klant” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 1.	Naast de gegevens over de melding is staat de status: In behandeling. Daaronder is een drop down box te vinden met daarin de mogelijke keuzes (zie hoofdstuk workflow voor mogelijke keuzes). Deze is nu leeg.		

2.9 Soort Wijzigen

2.9.1 Scenario soort wijzigen

Vooraf: zorg dat je de rol “projectleider” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 1.	Naast de gegevens over een melding staat een drop down box met daarin 2 mogelijke soorten: • BUG • RFC De BUG staat standaard geselecteerd bij deze melding.		
Wijzig het soort van de melding in RFC en druk op opslaan en verzenden.	Een overzicht van de gegevens van de melding wordt getoond, het soort is veranderd in RFC ook de status is veranderd in “Start”. De velden die horen bij een RFC zijn zichtbaar geworden.		

2.9.2 Scenario gegevens melding en soort wijzigen

Vooraf: zorg dat je de rol “projectleider” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 1. - Wijzig het soort van de melding in BUG en druk op opslaan en verzenden. - Wijzig de gegevens van de melding	Een overzicht van de gegevens van de melding wordt getoond, het soort is veranderd in BUG ook de status is veranderd in “Defect Open”. Het extra veld “extra naam” is zichtbaar geworden in het overzicht.		

2.9.3 Scenario soort wijzigen (ontwikkelaar)

Vooraf: zorg dat je de rol “ontwikkelaar” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Klik op Wijzig Gegevens bij het bekijken van de gegevens over melding 1.	Naast de gegevens over een melding is geen drop down box zichtbaar waarin het soort gewijzigd kan worden.		

2.10 Invoeren melding

2.10.1 Scenario invoeren lege en gevulde melding

Vooraf: zorg dat je de rol “ontwikkelaar” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Ga naar de URL: http://localhost/MySite/default.aspx - Klik op invoeren melding.	Het scherm om een nieuwe melding in te voeren wordt getoond.		
Voer niks in en druk op verzenden.	Er wordt op de pagina getoond dat er iets ingevuld dient te worden.		
Voer een titel en beschrijving in van een nieuwe melding en druk op verzenden.	Er wordt een pagina getoond dat de melding goed is ontvangen.		
Klik op “Terug”.	Je keert terug naar de overzicht pagina.		

2.10.2 Scenario invoeren melding (default user)

Vooraf: zorg dat je de rol “default user” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Ga naar de URL: http://localhost/MySite/default.aspx	De link “Invoeren melding” is niet aanwezig op de pagina.		

2.11 Verwijderen melding

2.11.1Scenario verwijderen melding

Vooraf: zorg dat je de rol “projectleider” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
- Klik op het nummer van een melding om de gegevens van deze melding te bekijken. - Druk op Verwijderen bij het tonen van de gegevens van een melding.	De algemene gegevens over de melding worden getoond met de vraag of de melding verwijderd mag worden.		
Klik op Verwijder.	Er wordt een scherm getoond dat de melding verwijderd zal worden.		
	De gegevens van de melding zijn uit de database verwijderd.		
Klik op “Terug”	Je keert terug naar de overzicht pagina.		

2.11.2Scenario verwijderen melding (default user)

Vooraf: zorg dat je de rol “default user” hebt (zie hiervoor het testplan “Koppelen gebruiker - rol”).

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Klik op het nummer van een melding om de gegevens van deze melding te bekijken.	Er is geen link “Verwijderen”		

2.12 Webpart Configuratie

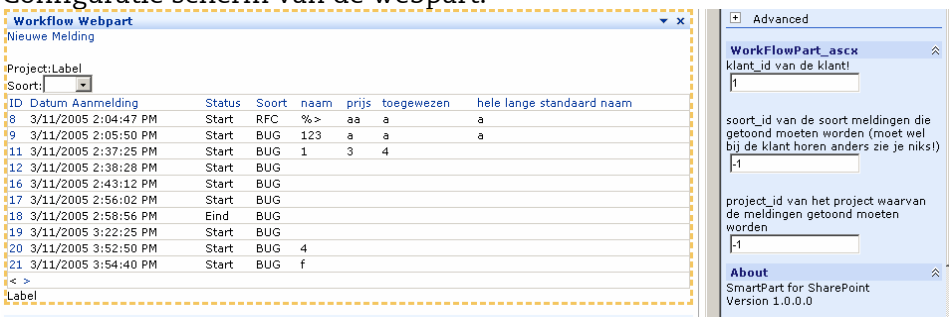
2.12.1Scenario webpart installatie per klant

	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
Ga naar http://localhost/MySite/default.aspx en klik op de ⋁ van de Workflow Webpart.	Configuratiescherm van de webpart (zie screenshot onder dit testplan).		
- Verander in het configuratiescherm het klant_id in 2. - Klik op OK.	De webpart toont enkel meldingen van klant nummer 2.		

2.12.2Scenario webpart installatie per soort en klant

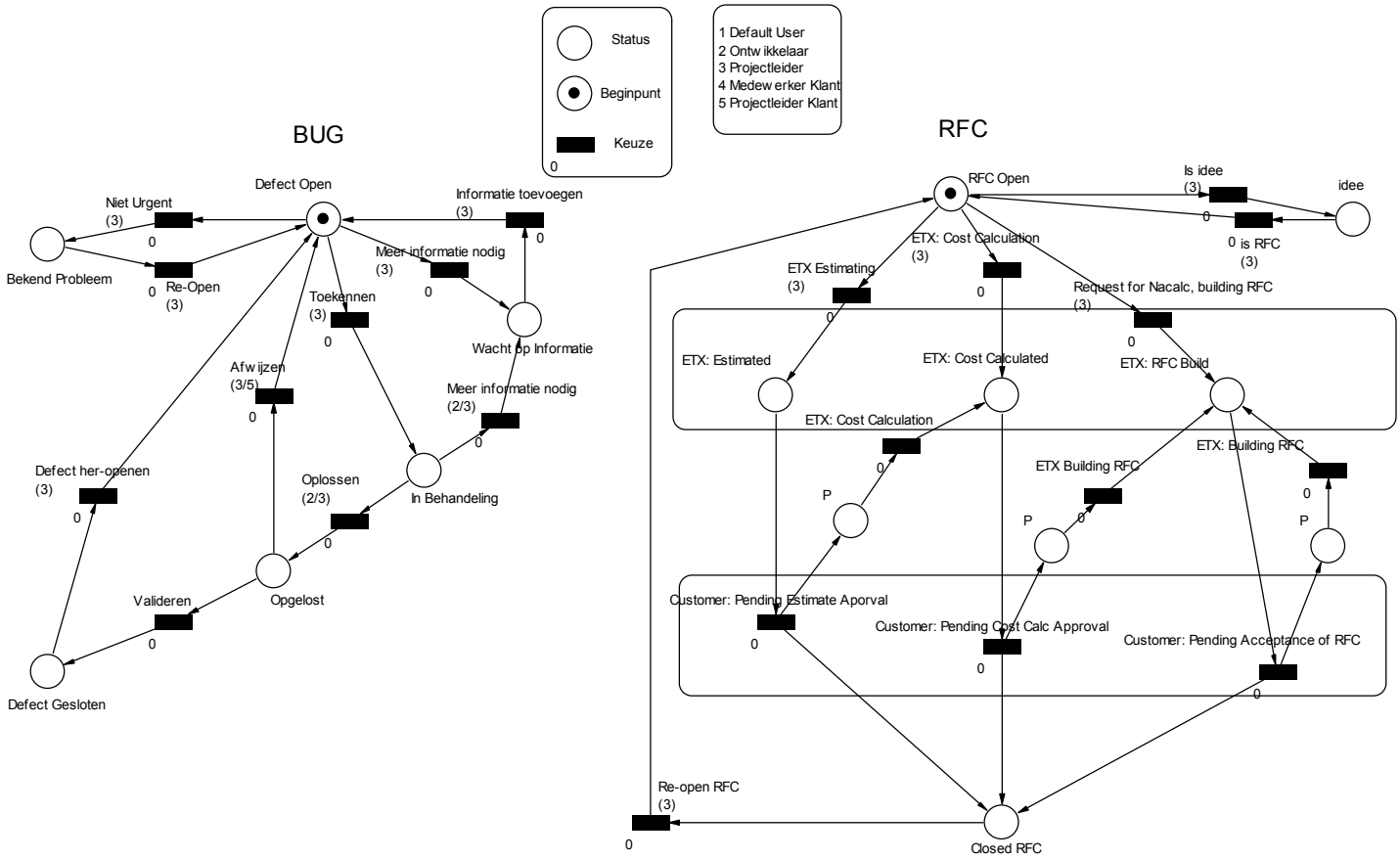
	Datum:		**Getest door:**
Actie	**Verwacht resultaat:**	**Ok**	**Opmerkingen**
-Ga naar http://localhost/MySite/default.aspx en klik op de ⋁ van de Workflow Webpart. - Verander in het configuratiescherm het soort_id in 2. - Klik op OK.	De webpart toont enkel meldingen van het soort RFC.		
- Klik op het ⋁ van de Workflow Webpart. - Verander in het configuratiescherm het klant_id in 2. - Klik op OK.	De webpart toont enkel meldingen van het soort RFC, van klant nummer 2.		

Configuratie scherm van de webpart:



3 Workflow

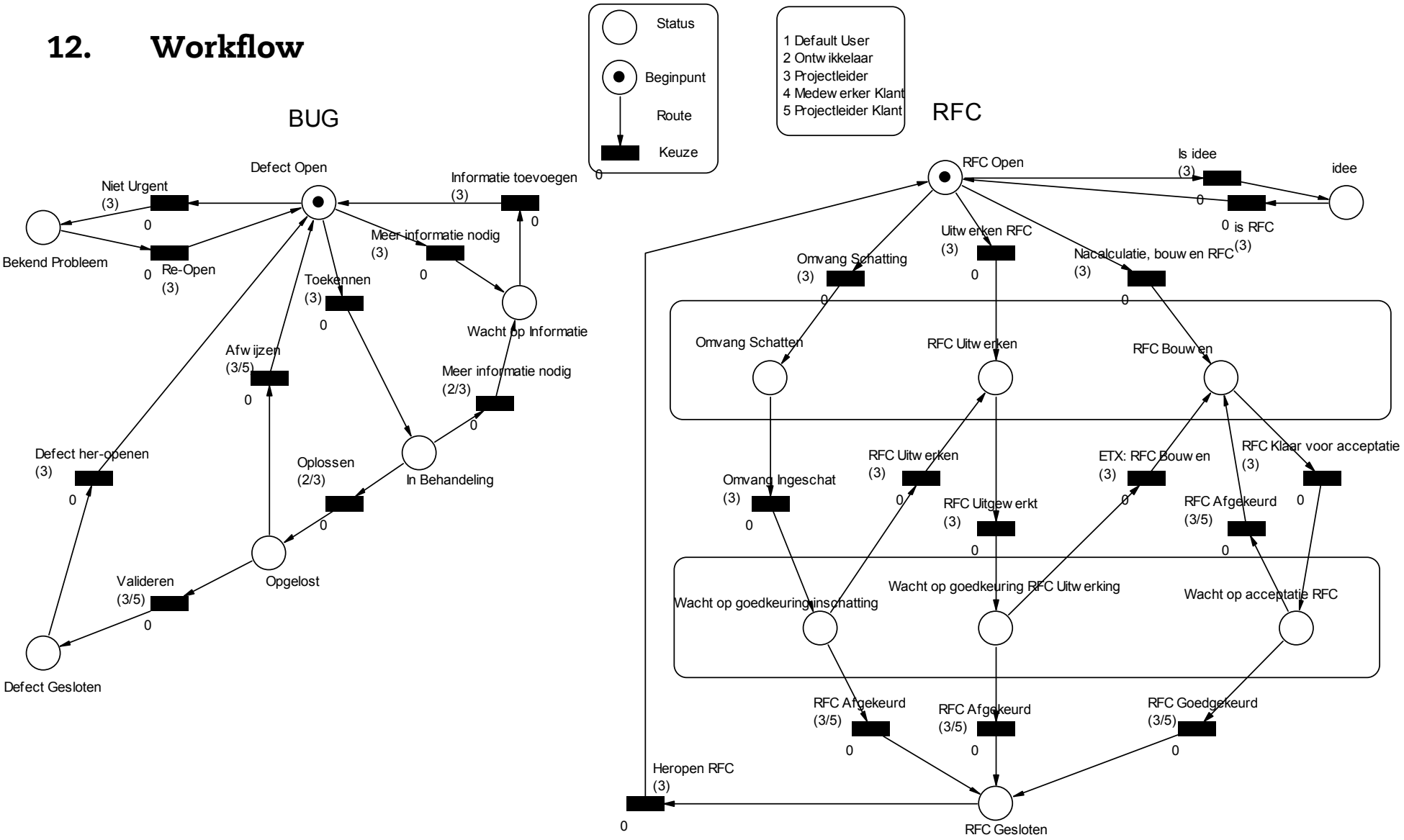
De workflow voor een BUG en een RFC. Deze workflow kan het systeem volgen.



4 Overige opmerkingen

Hieronder kunnen opmerkingen worden geplaatst die niet bij een specifiek scenario horen.

12. Workflow


Status
Beginpunt
Route
Keuze
1 Default User
2 Ontwikkelaar
3 Projectleider
4 Medewerker Klant
5 Projectleider Klant
BUG
Defect Open
Niet Urgent
(3)
Informatie toevoegen
(3)
Bekend Probleem
Re-Open
(3)
Meer informatie nodig
(3)
Toekennen
(3)
Wacht op Informatie
Afwijzen
(3/5)
Meer informatie nodig
(2/3)
Defect her-openen
(3)
Oplossen
(2/3)
In Behandeling
Valideren
(3/5)
Opgelost
Defect Gesloten
RFC
RFC Open
Is idee
(3)
idee
is RFC
(3)
Uitwerken RFC
(3)
Omvang Schatting
(3)
Nacalculatie, bouwen RFC
(3)
Omvang Schatten
RFC Uitwerken
RFC Bouwen
Omvang Ingeschat
(3)
RFC Uitwerken
(3)
RFC Uitgewerkt
(3)
ETX: RFC Bouwen
(3)
RFC Klaar voor acceptatie
(3)
RFC Afgekeurd
(3/5)
Wacht op goedkeuring inschatting
Wacht op goedkeuring RFC Uitwerking
Wacht op acceptatie RFC
RFC Afgekeurd
(3/5)
RFC Afgekeurd
(3/5)
RFC Goedgekeurd
(3/5)
Heropen RFC
(3)
RFC Gesloten