Afstudeer verslag

Weld Analysis System

ARPLAS SYSTEMS

HOGESCHOOL UTRECHT

Auteur: Martijn Brands 1554106

1e examinator: Gerald Ovink

Inhoudsopgave

Managementsamenvatting

In dit verslag wordt het ontwikkeltraject beschreven van het “Weld Analysis System”.
De opdracht is uitgevoerd bij het bedrijf AL-S Technology, een fabrikant van lassystemen voor de auto industrie. De klanten bestaan uit de bekende autofabrikanten zoals Audi, Daimler, General Motors, BMW enz. AL-S Technology opereert wereldwijd vanuit de hoofdvestiging in Amersfoort. De belangrijkste markten op dit moment zijn Europa, West en Zuid Amerika en Azië.

De lassystemen van AL-S Technology worden aangestuurd door een eigen ontwikkelde lascontroller. Deze lascontroller verzamelt veel data. Elke gemaakte las wordt gecontroleerd en de data wordt opgeslagen. Om deze data op een efficiënte manier te analyseren is er behoefte aan een geautomatiseerd systeem. Op dit moment worden de gegevens statistisch geanalyseerd met Excel. Dit is niet efficiënt en kost veel tijd.

Het doel van dit project is om een systeem te ontwerpen / ontwikkelen dat op een efficiënte wijze logfiles beheert en analyseert. Het systeem moet zo bijdragen aan het verbeteren van de prestaties van de lassystemen van AL-S Technology. Het systeem moet webbased zijn met een centrale data opslag (database).

Het systeem is ontwikkeld met de RUP methode.
In de inception fase is er een plan van aanpak opgesteld met de eisen en voorwaarden. Verder is er in deze fase een Use Case Model gemaakt met de te verwachten use cases. In deze fase is er ook al nagedacht over de mogelijke architectuur, frameworks en technieken die mogelijk gebruikt zouden kunnen worden.

In de elaboration fase is de architectuur verder uitgewerkt, zijn de technieken definitief gekozen en is de ontwikkelomgeving ingericht. Er is een use case specification document gemaakt waar alle uses cases gedetailleerd zijn met scherm ontwerpen. De ontwikkelomgeving is getest door een zeer eenvoudige webapplicatie te maken, met alle geselecteerde technieken en frameworks.
Onderstaande technieken zijn gebruikt:

- Unified Modeling Language (UML): Voor het ontwerpen van de architectuur modellen en de use cases.
- JAVA EE: voor het programmeren van de applicatie aan de server zijde.
- JSF 2.0: voor het programmeren van de web pagina’s
- PrimeFaces: voor de componenten op de web pagina’s
- Hibernate: voor een abstracte tussenlaag tussen de database en de webapplicatie.
- MySQL: voor de database.
- GlassFish applicatieserver: Voor het laten draaien van de applicatie in de ontwikkelomgeving.
- Netbeans IDE voor de ontwikkelomgeving
- MySQL Workbench
- StAX parser (XML)

De construction fase is in twee iteraties uitgevoerd. De eerste fase bevat alle CRUD use cases. De tweede iteratie bevat de use case die meer logica bevatten. Zoals het uploaden van logfiles, parsen van XML file en het analyseren van datasets. Bij het analyseren van datasets worden er grafieken gegenereerd, namelijk lijn grafiek en histogram. De applicatie draait in een webbrowser.

1. Inleiding

Dit verslag is geschreven naar aanleiding van de afstudeeropdracht “Weld Analysis System”.
Het project is uitgevoerd bij AL-S Technology, een leverancier van lassystemen voor de auto industrie.
In dit verslag wordt beschreven hoe het project tot stand is gekomen en welke keuzes zijn gemaakt.

1.1 Leeswijzer

Het Verslag is als volgt opgebouwd:

In hoofdstuk twee, Bedrijfsorganisatie, wordt beschreven wat het bedrijf AL-S Technology uitvoert en wat voor producten het produceert.

In hoofdstuk drie, Afstudeerproject, wordt de probleemstelling en doelstelling beschreven.

In hoofdstuk vier, Functionele analyse, wordt beschreven hoe de requirements en eisen tot stand zijn gekomen.

In hoofdstuk vijf, Inception fase, wordt beschreven hoe de inception fase samenhangt met het project.

In hoofdstuk zes, technieken en tools, wordt beschreven welke technieken, tools en frameworks gebruikt zijn.

In hoofdstuk zeven, Design van Database iteratie 1, wordt beschreven hoe het ontwerp van het businessmodel en de database is verlopen.

In hoofdstuk acht, Usability, wordt beschreven hoe er voor gezorgd is dat de applicatie gebruiksvriendelijk is.

In hoofdstuk negen, Elaboration fase, wordt beschreven hoe de elaboration fase samenhangt met het project.

In hoofdstuk tien, Ontwikkelomgeving, wordt beschreven hoe de ontwikkelomgeving to stand is gekomen.

In hoofdstuk elf, Architectuur, wordt besproken hoe de architectuur in elkaar zit. Aan de hand van de use case “location beheren” wordt dit verder uitgelegd.

In hoofdstuk twaalf, Design van Applicatie iteratie 2, wordt beschreven hoe het ontwerp en de architectuur van iteratie twee is verlopen.

In hoofdstuk dertien, Construction fase, wordt beschreven hoe de construction fase samenhangt met het project.

In hoofdstuk veertien, Conclusie en aanbevelingen, wordt beschreven wat de conclusie en aanbevelingen zijn van het ontwikkeltraject.

In hoofdstuk vijftien, Evaluatie, wordt het project geëvalueerd.

2 Bedrijfsorganisatie

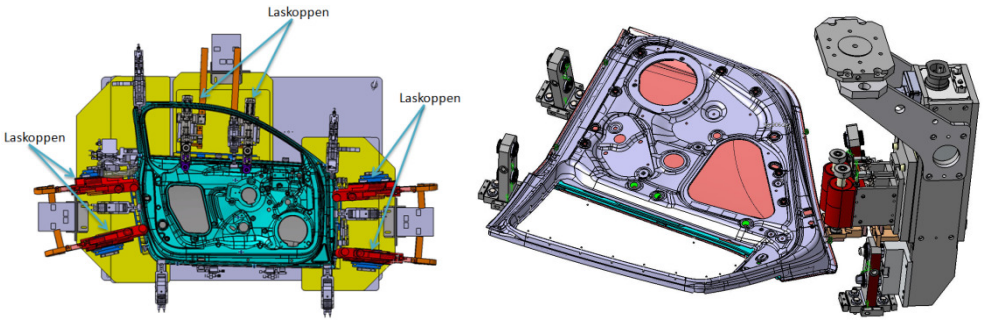
2.1 De Organisatie

De afstudeer opdracht wordt uitgevoerd bij AL-S Technology in Amersfoort.
AL-S Technology is tevens mijn werkgever, ik ben hier ruim 5 jaar werkzaam als project engineer en de laatste anderhalf jaar ook als software engineer.
AL-S Technology heeft als core business het vervaardigen en verkopen van lassystemen voor de auto industrie. Onze klanten bestaan uit de bekende autofabrikanten zoals Audi, Daimler, General Motors, BMW enz.

AL-S Technology opereert wereldwijd vanuit de hoofdvestiging in Amersfoort. De belangrijkste markten op dit moment zijn Europa, West en Zuid Amerika en Azië. AL-S Technology heeft wel wereldwijd partners die de buitenlandse activiteiten ondersteunen. Met name de lokale service partners zijn erg belangrijk voor AL-S Technology. De automotive industrie vraagt een zeer hoge beschikbaarheid, het is daarom van groot belang dat er lokaal snel gereageerd kan worden. De engineerings activiteiten vinden plaats vanuit Amersfoort. De fabricage en eind assemblage vinden zowel in Europa, west Amerika en zuid Korea plaats.
De lassystemen zijn compleet door AL-S Technology ontwikkeld. 95 % van alle maak onderdelen worden door toeleveranciers vervaardigd. De eindassemblage en eindtesten worden door AL-S Technology zelf uitgevoerd.

2.2 Het lassysteem

Het las systeem van AL-S Technology is een gepatenteerd systeem. Dit systeem kan in zeer korte tijd met hoge stromen lassen maken. De lastijd bedraagt +/- 3 tot 6 ms, met stromen tot 25kA.
AL-S Technology onderscheidt zich in het feit dat traditionele puntlas systemen veel langer lassen, veel warmte inbrengen en zichtbare laspunten achterlaten. Bij het systeem van AL-S Technology is dit niet het geval, zo kan er gelast worden op zeer smalle flenzen. Er kunnen min of meer onzichtbare directe lassen gemaakt worden. Bij indirecte lassen (serielassen) garandeert AL-S Technology een volledige onzichtbaarheid aan de buitenzijde.
De lassystemen van AL-S Technology worden meestal ingezet op moeilik te lassen plaatsen of op secundaire zichtpunten zoals deuren, motorklep, enz. AL-S Technology levert zowel vaste installaties als robot applicaties.



Figuur 1, links lassysteem vaste opstelling, rechts robot applicatie

2.3 Project betrokkenen

- Karel Pieterman eigenaar en begeleider bij AL-S Technology
- Gerald Ovink begeleider Hoge School Utrecht

Karel Pieterman

Karel Pieterman is gepromoveerd aan de universiteit van Delft, hier heeft hij vervolgens 5 jaar gewerkt als wetenschappelijk medewerker fysica.
De laatste 25 jaar als zelfstandig ondernemer / eigenaar van AL-S Technology.
Binnen AL-S Technology is Karel Pieterman verantwoordelijk voor de visie en strategische doelstellingen op technologisch gebied. Hij is mijn opdrachtgever en tevens begeleider voor dit project. Alle beslissingen omtrent het project zijn in samenspraak genomen. We hebben ongeveer twee wekelijks contact over de voortgang, technische en functionele aspecten van de opdracht.

Gerald Ovink

Gerald Ovink is mijn begeleider vanuit de Hogeschool Utrecht. Hij is tevens mijn eerste examinator. Tijdens het project is hij een paar keer bij AL-S Technology langs geweest om de voortgang en kwaliteit van de scriptie te bespreken.

3 Afstudeerproject

3.1 Aanleiding

De lassystemen van AL-S Technology worden wereldwijd ingezet. Het aantal installaties en klanten blijft groeien. Het is op dit moment erg moeilijk om te beoordelen hoe de verschillende systemen presteren.
De lascontrollers verzamelen erg veel data, tot wel 15.000 lassen per dag. Voor elke las wordt er een record bewaard op de lascontroller. Er is daarom een grote behoefte aan een tool die deze data op een efficiënte manier kan bewerken en analyseren.

In eerste instantie is deze applicatie voor de engineers en medewerkers van AL-S Technology bedoeld. Door op de gegevens statistische analyse los te laten krijgt men een goed inzicht in hoe een systeem presteert. De applicatie moet helpen om systemen beter te kunnen fine tunen en fouten vroegtijdig op te kunnen sporen.

Er is nagedacht over hoe de applicatie er uit zou moeten zien, een desktop applicatie of een web-based applicatie. Er is gekozen voor een webbased applicatie met een database. Dit is besloten omdat zo de lasdata op een centrale plaats en op gestructureerde wijze bewaard wordt. Iedereen over de gehele wereld die met het systeem werkt beschikt over dezelfde gegevens.

3.2 Probleemstelling

Er is geen applicatie beschikbaar die op een snelle en efficiënte wijze grote hoeveelheden lasdata kan analyseren. De werkwijze die op dit moment gevolgd wordt is tijdrovend en verre van efficiënt. Verder is er een probleem dat de engineers overal logfiles hebben op laptop, desktop, externe harde schijven enz. Deze logfiles worden rond gemaild als iemand ze nodig heeft. Vaak is het niet exact bekend bij welke installatie de files horen. De applicatie moet helpen om de logfiles op een gestructureerde wijze op te slaan, en deze moet vervolgens op een gebruiksvriendelijke wijze weer beschikbaar zijn voor verdere analyse.

Arplas Qcs - Log History - C:\Users\m.brands\Desktop\Audi_22-09-2011.xml

System View Admin Profiles Log Counters Extra

Profiles | Log File | Counter | Terminal | Audi_22-09-2011 | QCS Ignition Protoc

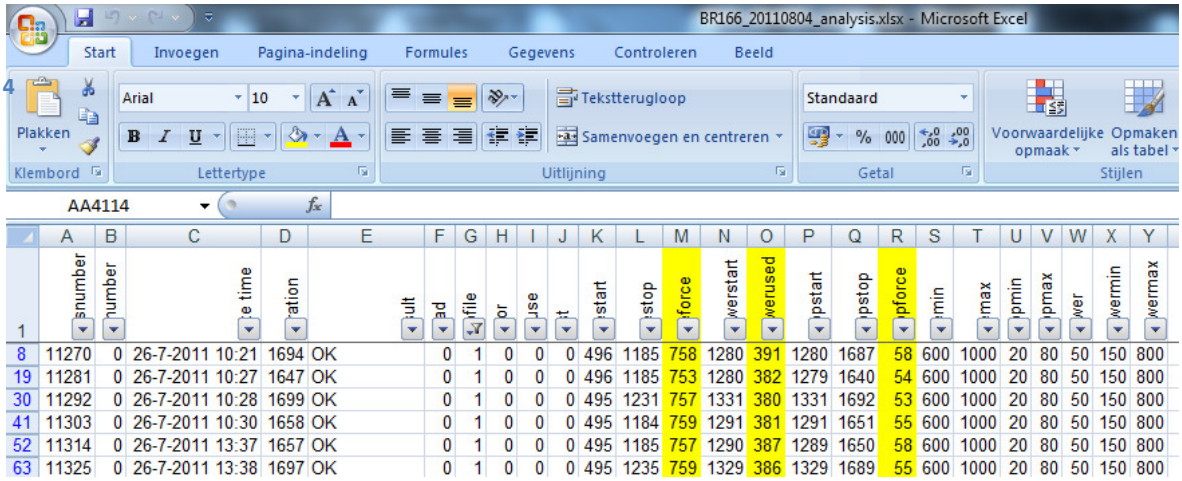
qcs	date	prof	preforce	postforce	dropforce	fire	power	duration	graph	result
0000002158	23-9-2011 12:12	002	0726	0715	0011	02445	0274	2881	☑	OK
0000002157	23-9-2011 12:11	002	0726	0713	0013	02446	0264	2845	☑	OK
0000002156	23-9-2011 12:00	002	0728	0714	0014	02444	0253	2843	☑	OK
0000002155	23-9-2011 11:59	002	0727	0711	0016	02445	0253	2845	☑	OK
0000002154	23-9-2011 11:58	002	0729	0710	0019	02446	0254	2882	☑	OK
0000002153	23-9-2011 11:58	002	0724	0709	0015	02484	0253	2884	☑	OK
0000002152	23-9-2011 11:57	002	0725	0709	0016	02518	0260	2918	☑	OK
0000002151	23-9-2011 11:56	002	0725	0712	0013	02441	0252	2877	☑	OK
0000002150	23-9-2011 11:52	002	0727	0709	0018	02447	0252	2847	☑	OK
0000002146	23-9-2011 11:41	002	0734	0713	0021	02443	0246	2844	☑	OK
0000002145	23-9-2011 11:41	002	0730	0711	0019	02442	0246	2842	☑	OK
0000002144	23-9-2011 11:40	002	0733	0714	0019	02410	0246	2810	☑	OK
0000002143	23-9-2011 11:40	002	0726	0709	0017	02410	0246	2810	☑	OK
0000002142	23-9-2011 11:40	002	0727	0710	0017	02410	0246	2810	☑	OK
0000002141	23-9-2011 11:39	002	0724	0707	0017	02452	0248	2853	☑	OK
0000002140	23-9-2011 11:39	002	0724	0709	0015	02415	0246	2815	☑	OK
0000002139	23-9-2011 11:38	002	0722	0706	0016	02478	0247	2877	☑	OK
0000002138	23-9-2011 11:38	002	0722	0707	0015	02485	0246	2884	☑	OK
0000002137	23-9-2011 11:37	002	0722	0706	0016	02451	0246	2851	☑	OK
0000002136	23-9-2011 11:36	002	0724	0706	0018	02480	0246	2880	☑	OK

Figuur 2, Voorbeeld van de logrecords per laspunt, screenshot van het PC programma wat met de QCS Weldcontroller communiceert.

In figuur 2 is het PC programma te zien wat gebruikt wordt om de lascontroller in te stellen en logfiles te downloaden. In figuur 2 zijn de log records te zien per laspunt. De kolommen pre-force, post-force en power zijn de kolommen die gebruikt worden voor analyse. Hier is te zien dat het lastig is om over getoonde gegevens een statistische conclusie te trekken. Dit PC programma is er dan ook voor gemaakt de lassen per record te beoordelen.

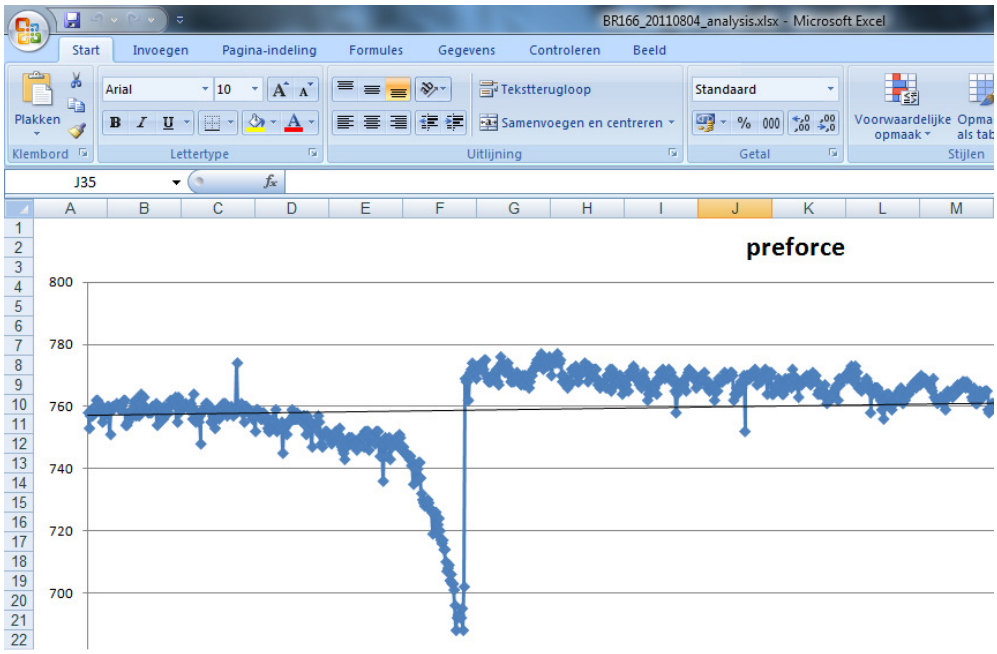
Pre-force, post-force en power

Het AL-S Technology lassysteem is opgebouwd rond een gevoelig krachtmeetsysteem. Dit krachtmeetsysteem meet de kracht tussen de elektroden voor de las (pre-force). Als deze pre-force binnen de ingestelde grenzen ligt dan wordt er een las gemaakt. Vervolgens wordt de kracht gemeten na de las (post-force). Tijdens de las wordt de lasstroom (power) gemeten. Deze 3 factoren kunnen nauwkeurig bepalen of de gemaakte las een goede of slechte las was. De lascontroller kan deze conclusie alleen maar trekken als de meetgrenzen voor deze 3 parameters optimaal zijn ingesteld. Anders is het mogelijk dat goede lassen afgekeurd worden en slechte lassen goedgekeurd worden. Daarom is het belangrijk om de statistische gegevens van deze waarden inzichtelijk te maken. In figuur 4 is al iets merkwaardigs te zien, de pre-force zou een stabiele waarde moeten hebben voor elke las, maar zoals te zien is loopt die terug om vervolgens in één keer weer omhoog te schieten. Daarom is het te ontwikkelen “Weld Analysis System” zo belangrijk, om dit soort feiten op te sporen en te verklaren.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	snumber	number	e time	ation	ult	ad	file	or	se	t	start	stop	force	werstart	werused	pstart	pstop	pforce	min	max	pmin	pmax	wer	wermin	wermax
8	11270	0	26-7-2011 10:21	1694	OK	0	1	0	0	0	496	1185	758	1280	391	1280	1687	58	600	1000	20	80	50	150	800
19	11281	0	26-7-2011 10:27	1647	OK	0	1	0	0	0	496	1185	753	1280	382	1279	1640	54	600	1000	20	80	50	150	800
30	11292	0	26-7-2011 10:28	1699	OK	0	1	0	0	0	495	1231	757	1331	380	1331	1692	53	600	1000	20	80	50	150	800
41	11303	0	26-7-2011 10:30	1658	OK	0	1	0	0	0	495	1184	759	1291	381	1291	1651	55	600	1000	20	80	50	150	800
52	11314	0	26-7-2011 13:37	1657	OK	0	1	0	0	0	495	1185	757	1290	387	1289	1650	58	600	1000	20	80	50	150	800
63	11325	0	26-7-2011 13:38	1697	OK	0	1	0	0	0	495	1235	759	1329	386	1329	1689	55	600	1000	20	80	50	150	800

Figuur 3, Excel werkblad met analyse van logfile



Figuur 4, Excel werkblad met pre-force grafiek, elk punt is een las

Zoals te zien is (figuur 3 en 4), is het een tijdrovende klus om alle data in Excel te krijgen, zeker als het een project betreft met 12 lascontrollers. Het overzicht is snel zoek en het zijn allemaal separate bestanden die geen duidelijke naam hebben. Daarom is het van groot belang dat in het nieuwe systeem geüploade logfiles een eenduidige relatie hebben met een geselecteerde lascontroller. Alleen op deze manier is het mogelijk om op de gegevens te vertrouwen en de juiste conclusies te kunnen trekken.

3.3 Doelstelling

Het doel van dit project is om een systeem te ontwerpen / ontwikkelen dat op een efficiënte wijze logfiles beheert en analyseert. Het systeem moet zo bijdragen aan het verbeteren van de prestaties van de lassystemen van AL-S Technology. Door logfiles te analyseren van 5000 records of groter kunnen er fouten opgespoord worden, kunnen controle grenzen optimaal ingesteld worden enz. Door duidelijke grafieken te tonen moet het voor een gebruiker eenvoudig zijn om de gewenste informatie te vinden.

De volgende functies moeten beschikbaar zijn :

1. Service monteurs en medewerkers van AL-S Technology moeten in kunnen loggen zodat bekend is wie welke files uploadt.
2. Een of meerdere beheerders moeten het systeem kunnen onderhouden, bijv. klanten beheren, log files beheren enz.
3. Gebruikers moeten files kunnen uploaden.
4. Gebruikers moeten op een dataset (geüploade logfiles) een analyse kunnen maken. Hierbij moet een zeer gebruiksvriendelijke user interface voor handen zijn. De gebruiker kan d.m.v. het genereren van grafieken conclusies trekken over het gedrag / prestatie van het lassysteem.
5. De user interface moet in een webbrowser draaien, de data is centraal opgeslagen in een database.

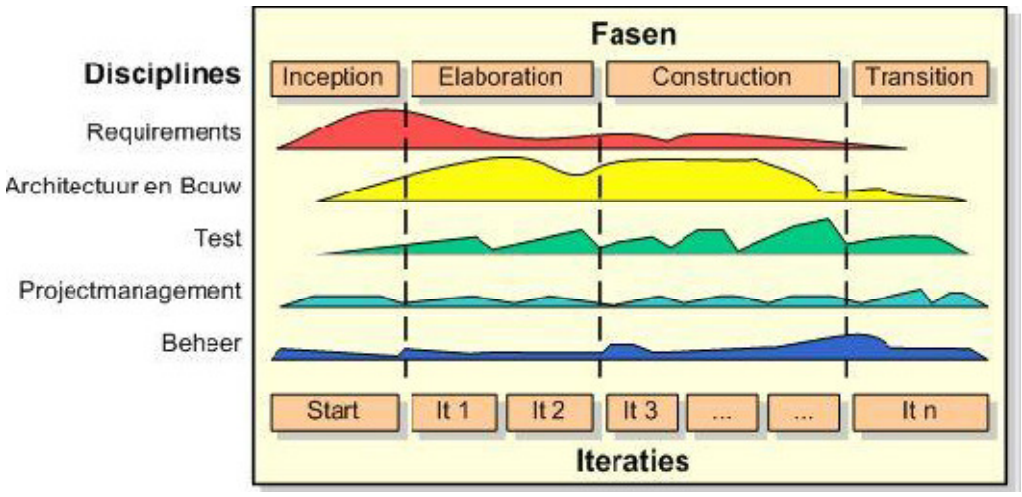
De applicatie is werkend te zien in de ontwikkelomgeving. Deployment op een server zal in een later stadium plaatsvinden. De applicatie is wel zo ontworpen dat die geschikt is om op een server te installeren en voor meerdere gebruikers beschikbaar is.

3.4 Op te leveren producten

- Use Case Model Document
- Use Case Specification Document / functioneel ontwerp
- ERD / technisch ontwerp
- Data analyse (welke data uit de logfile gaan we gebruiken)
- Web applicatie met database

3.5 Project Aanpak en planning

Het afstudeerproject betreft een software ontwikkelingstraject. Ik heb daarom gekozen voor de RUP methode. Deze methode wordt ook op de Hogeschool Utrecht toegepast bij software ontwikkel projecten. De RUP methode bestaat uit 4 fasen Inception fase, Elaboration fase, Construction fase en Transition fase.



Figuur 5, Fasen binnen RUP

In de **Inception fase** wordt de basis gelegd voor het project. Hier wordt bekeken of het project haalbaar is en of de kosten opwegen tegen de baten. In overleg met de opdrachtgever heb ik gekeken naar de wensen en eisen die er op dat moment waren. We hebben toen besloten dat het project haalbaar was mits er voldoende tijd beschikbaar zou zijn.

In de inception fase wordt het plan van aanpak[1] geschreven en goedgekeurd. Het schrijven van het plan van aanpak verliep zonder problemen. Door het directe contact met de opdrachtgever was er snel een goede afbakening van de functionaliteiten die de applicatie moest bevatten.

Naast het plan van aanpak is ook het use case model[2] opgesteld. Hierin zijn alle functionaliteiten opgesteld. Het use case document is uitvoerig met opdrachtgever besproken zodat het voor beide partijen duidelijk was hoe de applicatie er uit moest zien en welke functionaliteiten die moest bevatten. Na goedkeuring van beide documenten was de basis gelegd voor het project en hadden alle partijen overeenstemming.

In de inception fase is er ook gekeken naar welke technieken er gebruikt zouden gaan worden. Nadat besloten was welke technieken er gebruikt werden is er begonnen met het inrichten van de ontwikkel omgeving.

De **Elaboration Fase** bestaat uit het detailleren van het project. De use cases worden gedefinieerd met bijbehorende business rules. De gebruikte technieken worden verder gedetailleerd. Doordat samen met de opdrachtgever besloten is een web applicatie te ontwikkelen i.p.v. een desktop applicatie zijn er andere technieken nodig. Hier is dus de nodige aandacht aan besteed.

In de **Construction fase** wordt het project daadwerkelijk ontwikkeld en worden er werkende componenten opgeleverd. Er is besloten om de Construction fase in twee iteraties uit te voeren. In de eerste iteratie worden alle use cases ontwikkeld die aan de administratieve kant van het systeem thuis horen. In de tweede iteratie zijn meer de technische zaken ondergebracht zoals het lezen van de XML files (log files), het creëren van grafiek over bepaalde datasets.

De transition fase behoort niet binnen de scope van dit project. Dit zal op een later moment plaats vinden.

[1] Bijlage A bevat het plan van aanpak

[2] Bijlage B bevat Use Case Model

4 Functionele analyse

In dit hoofdstuk wordt de functionele analyse beschreven en de eisen (requirements) die de opdrachtgever stelt aan het systeem.

De functionele analyse en requirements zijn tot stand gekomen door een aantal gesprekken met de opdrachtgever. De Excel template die gebruikt werd voor analyse is bestudeerd. De analyse functies die in de Excel template aanwezig waren moesten zeker terug komen in het te ontwikkelen systeem.

4.1 Requirements analyse functies:

De volgende analyse functies moet in de applicatie beschikbaar zijn :

Lijndiagram Pre Force
Wordt gebruikt om de “pre-force” waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Lijndiagram Post Force
Wordt gebruikt om de “post-force” waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Lijndiagram WeldPower
Wordt gebruikt om de “gemeten lasstroom“ waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Histogram pre-force
Wordt gebruikt om de proces variabelen m.b.t. “pre-force” weer te geven. Hoe stabiel is het proces? (Sigma).

Histogram post-force
Wordt gebruikt om de proces variabelen m.b.t. “post-force” weer te geven. Hoe stabiel is het proces? (Sigma).

Histogram WeldPower
Wordt gebruikt om de proces variabelen m.b.t. “gemeten lasstroom” weer te geven. Hoe stabiel is het proces? (Sigma).

De volgende analyse functies zijn wenselijk maar zijn geen must:

Cirkeldiagram lasfouten
Geeft per laspunt aan hoeveel en welke lasfouten zijn opgetreden.

4.2 Opstellen requirements

Voor het opstellen van de requirements is gebruik gemaakt van de **MoSCoW** methode. de requirements die daar uit ontstaan zijn, zijn vervolgens verder opgesplitst in het use case model. Een belangrijke eis was dat het system erg gebruiksvriendelijk[3] en efficiënt functioneert. Een gebruiker moet met een paar muisklikken een logfile kunnen uploaden, en vervolgens d.m.v. de analysefuncties deze logfile kunnen analyseren.

[3] In hoofdstuk 8 wordt ingegaan op gebruiksvriendelijkheid en usability

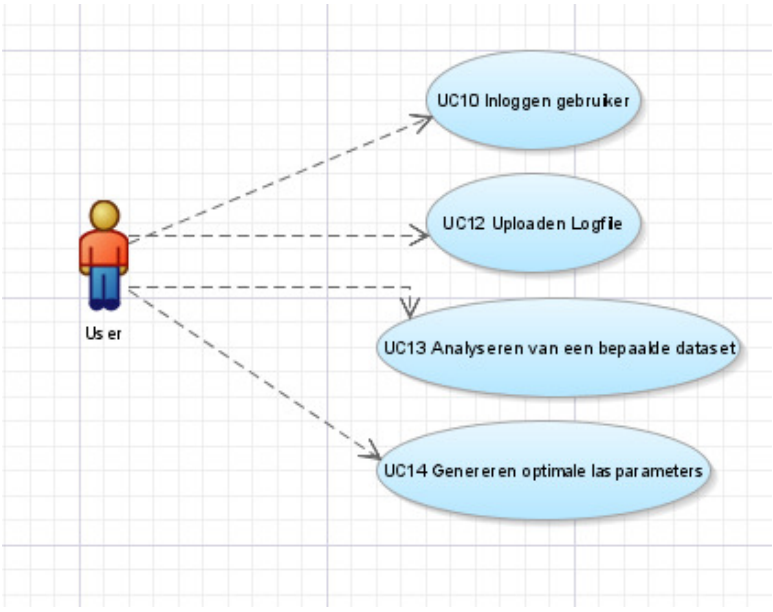
De letters in **MoSCoW** staan voor:

Must have	**(M)**	De eisen waaraan de webapplicatie **MOET** voldoen, zonder deze eisen werkt de webapplicatie niet zoals het zou moeten.
Should have	**(S)**	De eisen die de webapplicatie zou moeten hebben, als het mogelijk is.
Could have	**(C)**	De eisen die de webapplicatie zou kunnen hebben. Deze eisen worden gezien als “leuk als ze erin zitten”.
Would have	**(W)**	De eisen die de webapplicatie niet zal hebben, maar wel toegevoegd kunnen worden.

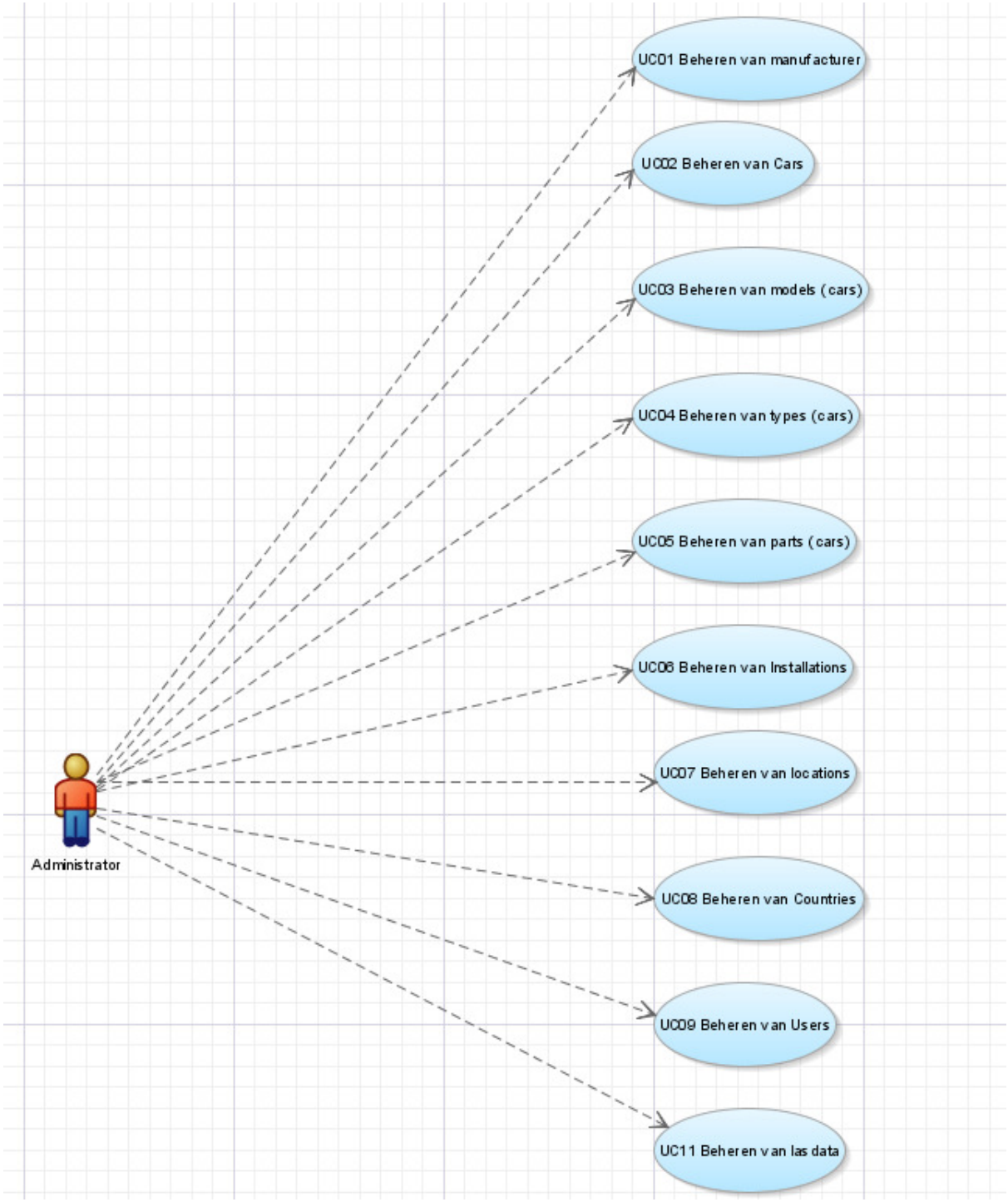
Requirements Weld Analysis System	**Prioriteit**
Beheren van gebruikers	M
Beheren van Klanten / Projecten	M
Uploaden van logfiles	M
Beheren van logfiles / lasdata	M
Analyseren van lasdata (zie H4.1)	M
Genereren van optimale lasparameters	C
Moet in een webbrowser draaien	M
Kan automatisch logfiles binnenhalen van een lascontroller	W

Figuur 6, Requirements tabel MOSCOW

Met de requirements is vervolgens een use case model opgesteld. Het use case model is een middel om de systeemeisen vast te stellen. Het visualiseert welke systeemfuncties er moeten komen en wat iedere functie precies moet doen. In dit document zijn ook de verschillende typen gebruikers geïdentificeerd, namelijk een administrator en een user.
Een Use Case is een serie acties die het systeem uitvoert om een duidelijk resultaat voor een gebruiker op te leveren. In figuur zeven en acht wordt het use case model getoond voor de administrator en een normale user.



Figuur 7, Use Case Model voor Actor User



Figuur 8, Use Case Model voor Actor Administrator

5 Inception Fase

In dit hoofdstuk wordt de Inception fase beschreven in samenhang met het project. De inception fase heeft als doel om de volgende punten bevestigd te krijgen:

Zijn we het eens over de scope?
Dit punt is reeds besproken in het plan van aanpak. Na overleg met opdrachtgever is er besloten om een webbased systeem te bouwen om lasdata te analyseren en te beheren.

Is er op zijn minst een oplossingrsrichting bekend?
Er is een oplossing bekend, namelijk een webbased systeem met database, de technieken[4] en frameworks zijn reeds geselecteerd.

Zijn we het eens over de wensen en eisen?
In de inception fase is tegelijk met het plan van aanpak het usecase model opgeleverd. Hierin staat duidelijk beschreven wat de wensen en eisen moeten zijn.

Hebben we de belangrijkste risico's en afdoende tegenmaatregelen in beeld?
Het grootste risico wat op kan treden is het realiseren binnen de gestelde tijd. Omdat het systeem geen bedrijfskritisch proces betreft is het geen probleem als de gestelde deadline niet precies gehaald wordt. Omdat het project gerealiseerd wordt met open source technieken is het kosten aspect geen risico.

Zijn we het erover eens dat de globale planning en kosteninschatting realistisch zijn? (Software Development Plan)?
De globale planning is zeer krap, maar kan realistisch zijn als er 100% van de beschikbare tijd aan het project gewerkt wordt. De kosteninschatting is realistisch en betreft alleen uren, omdat er met opensource technieken gewerkt wordt.

Zijn we het eens over het te volgen proces en de tools waarmee we de oplossing realiseren?
Voorwaarde voor de tools is dat ze opensource zijn. Met de beschikbare opensource tools is het zeer goed mogelijk dit project te realiseren.

Conclusie is dat de resultaten van de Inception fase voldoende zijn om het project voort te zetten. In het volgende hoofdstuk worden technieken en tools besproken die voor het project gebruikt zijn.

[4] Hoofdstuk 6, Technieken en tools.

6 Technieken en Tools

Tijdens de ontwikkeling van dit project zijn de volgende technieken toegepast :

- Unified Modeling Language (UML): Voor het ontwerpen van de architectuur modellen en de use cases.
- JAVA EE: voor het programmeren van de applicatie aan de server zijde.
- JSF 2.0: voor het programmeren van de web pagina's
- PrimeFaces: voor de componenten op de web pagina's
- Hibernate: voor een abstracte tussenlaag tussen de database en de webapplicatie.
- MySQL: voor de database.
- GlassFish applicatieserver: Voor het laten draaien van de applicatie in de ontwikkelomgeving.
- Netbeans IDE: voor de ontwikkelomgeving
- MySQL Workbench : Voor het ontwerpen van de database (ERD)
- StAX parser (XML): Techniek om XML files te bewerken.

De technieken die gekozen zijn, zijn allemaal gebaseerd op open source. Dit was een vereiste om de ontwikkelkosten laag te houden.

6.1 UML

De UML techniek is gebruikt vanwege de ervaringen die ik op school opgedaan heb. Vanuit de opleiding wordt er veel met deze techniek gewerkt, de ervaringen zijn positief. De UML techniek past ook precies binnen RUP. UML bevat erg veel technieken en methoden, niet alle technieken zijn gebruikt. De technieken die voor het project gebuikt zijn :

- Use case model (diagram)
- Klasse diagram

6.2 JAVA EE

Het gebruik van JAVA EE (Enterprise Edition) is mede gebaseerd op de persoonlijke ervaring die ik met JAVA heb. Vanuit de opleiding hebben we vanaf begin af aan in JAVA leren programmeren, het is een goede en krachtige taal om in te programmeren. In combinatie met het open source aspect met bijbehorende applicatie server is JAVA de juiste taal. Verder is JAVA wereldwijd een van de meest gebruikte talen, is er ontzettend veel informatie te vinden met voorbeelden, uitleg en zijn er ontelbare forums die hulp kunnen bieden als je een probleem hebt met een bepaald vraagstuk. JAVA EE is open source, ter vergelijking is er naar ASP .NET gekeken maar om dit goed te kunnen gebruiken is de Visual Studio IDE een vereiste en een Microsoft applicatie server deze zijn niet open source.

6.3 JSF2, PrimeFaces :

Dit is een logisch vervolg als er met JAVA EE gewerkt wordt. JSF 2.0 (JAVA Server Faces) biedt een heldere en duidelijk oplossing om de server side code apart te houden van de webpagina's. De webpagina's worden in XHTML geschreven, XHTML zorgt voor een goede leesbaarheid van de webpagina code.

PrimeFaces is een component suite met meer dan 100 componenten, die volledig compatible is met JSF 2. Er zitten veel kant en klare componenten in zoals bijv. een file upload. Omdat er veel standaard componenten in zitten kan de ontwikkeltijd van het project verkort worden.

6.4 MySQL

MySql is een relationele database. Tegenwoordig is MySql onderdeel van Oracle. Er bestaan enterprise editions maar er is ook een open source edition de “MySql Community Server” edition, deze is gebruikt voor het project.
Omdat dit eigenlijk de enige database is die open source is en ook voor grote schaalbare projecten ingezet wordt. Zo gebruiken Facebook en Twitter MySQL, maar dan de enterprise edition. Volgens de site van Oracle[5] is het eenvoudig om van de open source edition te upgraden naar de enterprise edition. Dit aspect is meegenomen omdat de database het hart van de applicatie is en dus bedrijfszeker moet draaien. Verder is er een goede designer tool MySQL Workbench beschikbaar.

6.5 Netbeans

Netbeans is een ontwikkel tool voor verschillende platformen maar met name gericht op JAVA en JAVA EE.
De keuze van Netbeans is gebaseerd op een vergelijking van Eclipse en Netbeans en dan met name de usability. Netbeans wordt out of the box geleverd met applicatie server GlassFish. Een JSF testapplicatie met een “hello World” pagina is in een handomdraai gemaakt en functioneert direct. Bij Eclipse moeten nogal wat instellingen en installaties gedaan worden voordat de ontwikkelomgeving gereed is. Verder is de user interface van Netbeans erg overzichtelijk en werkt zeer gebruiksvriendelijk.

6.6 Hibernate

Hibernate is een framework om database entiteiten en JAVA objecten aan elkaar te koppelen. Hibernate verzorgt de complete communicatie met de database.
Als niet voor een framework als Hibernate gekozen zou worden dan moeten alle koppelingen handmatig gecodeerd worden. Dit is erg veel werk en vergt veel testwerk.
Omdat Hibernate een lichtgewicht framework is en open source is, is dit uitermate geschikt voor deze applicatie. Mede met het oog op de toekomst als de applicatie daadwerkelijk draait op een server zorgt Hibernate voor de juiste belasting van de database connecties. Verder blijkt uit de vele voorbeelden op internet dat Hibernate, GlassFish en MySQL prima samen werken.

6.7 GlassFish

GlassFish is een applicatie server van Oracle. Het heeft een open source versie en een Enterprise edition versie. Voor het project is de open source versie gebruikt.

[5] http://www.oracle.com/us/products/mysql/mysqlcommunityserver/index.html

GlassFish is in eerste instantie geselecteerd als applicatie server omdat het met Netbeans meegeleverd wordt. Maar na het lezen van een aantal blogs[6] en de community site[7] blijkt het ook nog eens een erg goede en stabiele applicatie server te zijn. GlassFish heeft een goede webbased user interface om de vele mogelijkheden in te kunnen stellen.

6.8 MySql Workbench

MySql Workbench is een tool om database ontwerpen in te maken. Er kan grafisch een ERD opgesteld worden. Als ERD gereed is kan met MySql Workbench het ERD via een script automatisch naar de database gestuurd worden. Voor deze tool is gekozen omdat die erg goed met MySql samenwerkt. Zo kunnen veranderingen in het ERD automatisch met de database gesynchroniseerd worden.

6.9 StAX XML Parsing

Er zijn in JAVA verschillende XML technieken beschikbaar we kunnen daarin drie soorten onderscheiden[8]:

- DOM parsers
- Pull Parsers
- Push Parsers

DOM parsers lezen een complete XML file in en plaatst deze als objecten in het geheugen. Bij grote XML files kan dit performance en geheugen problemen opleveren.
De Push parsers, ook wel streaming parsers genoemd beginnen met het inlezen van de XML file en pushen de informatie naar de cliënt.
De StAX parser is een zogenaamde pull parser. Dit wil zeggen dat de cliënt (applicatie) steeds vraagt (pull) om het volgende event te sturen. De applicatie kan dan vervolgens beslissen hier iets mee te doen of het volgende event te vragen.
Voor het Weld Analysis System is de StAX parser gebruikt vanwege de goede performance en goede flexibiliteit. Doordat er per event getriggerd wordt is zo precies te bepalen welke events (log parameters) gebruikt worden om op te slaan. Met de events die niet interresant zijn wordt niets gedaan.

Feature	StAX	SAX	DOM
API Type	Pull, streaming	Push, streaming	In memory tree
Ease of Use	High	Medium	High
XPath Capability	No	No	Yes
CPU and Memory Efficiency	Good	Good	Varies
Forward Only	Yes	Yes	No
Read XML	Yes	Yes	Yes
Write XML	Yes	No	Yes
Create, Read, Update, Delete	No	No	Yes

Figuur 9, Kenmerken verschillende XML parsers

[6] https://wikis.oracle.com/display/glassfish/TipsAndBlogs
[7] http://glassfish.java.net/
[8] http://docs.oracle.com/cd/E17802_01/webservices/webservices/docs/1.6/tutorial/doc/SJSXP2.html

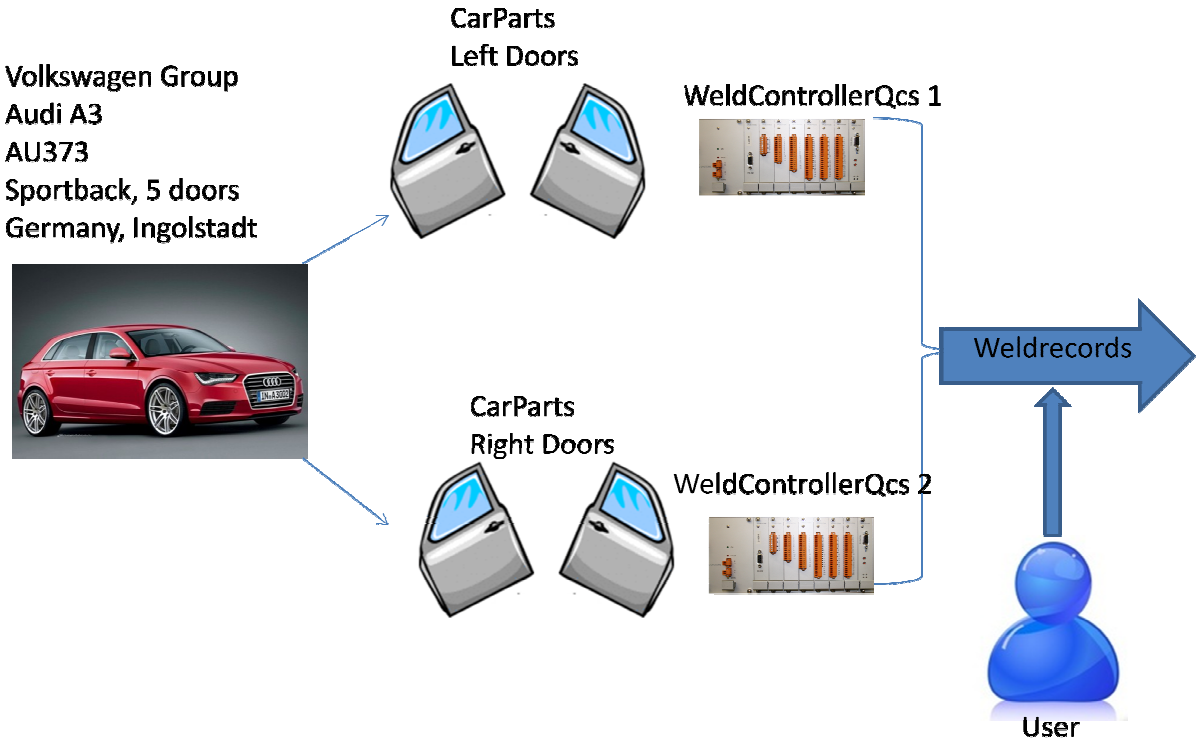
7 Design van database, iteratie 1

In dit hoofdstuk wordt beschreven hoe de database ontworpen is. Dit is tegelijk ook het design van iteratie één. Omdat er met Hibernate gewerkt wordt is het business model een reflectie van het database model. Hibernate verzorgt de mapping in dit model.

7.1 Database Design

Nadat de requirements bekend waren is de database architectuur ontworpen. Het programma wat hiervoor gebruikt is, is MySql Workbench. Het ERD wordt visueel getekend. In de tabellen kan je kolommen definiëren met bijbehorende constraints.
Als de database configuratie in orde is kan het ERD geupload worden naar de MySql database.
Als er later wijzigingen doorgevoerd dienen te worden, wordt het ERD gewijzigd. Door het ERD te synchroniseren met de database worden de wijzigingen overgenomen. Dit werkt erg snel en foutloos.

Voor een goedwerkend systeem is het van groot belang dat er een goed doordachte database architectuur beschikbaar is. Uiteindelijk draait het om de lasdata, die moet bij een uniek onderdeel van een auto, op een productie locatie horen. Omdat de fabrikanten verschillende merken voeren, verschillende modellen produceren en verschillende productie locaties hebben moest dit allemaal in de architectuur meegenomen worden.
Sommige fabrikanten spreken alleen met codenamen terwijl andere fabrikanten spreken van de modelnaam zoals die verkocht wordt. Verder was het wenselijk om het systeem een uniek nummer te laten genereren wat bij de lascontroller hoort, dit unieke nummer wordt dan vervolgens fysiek op elke lascontroller aangebracht. Zo kan een engineer in een oogopslag zien om welke lascontroller het gaat. Voor dit unieke nummer wordt de “primaire sleutel” gebruikt van de tabel “WeldControllerQcs”. Verder wordt aan elke Logrecord een user gekoppeld op deze manier is altijd te achterhalen wie welke logrecords geupload heeft.

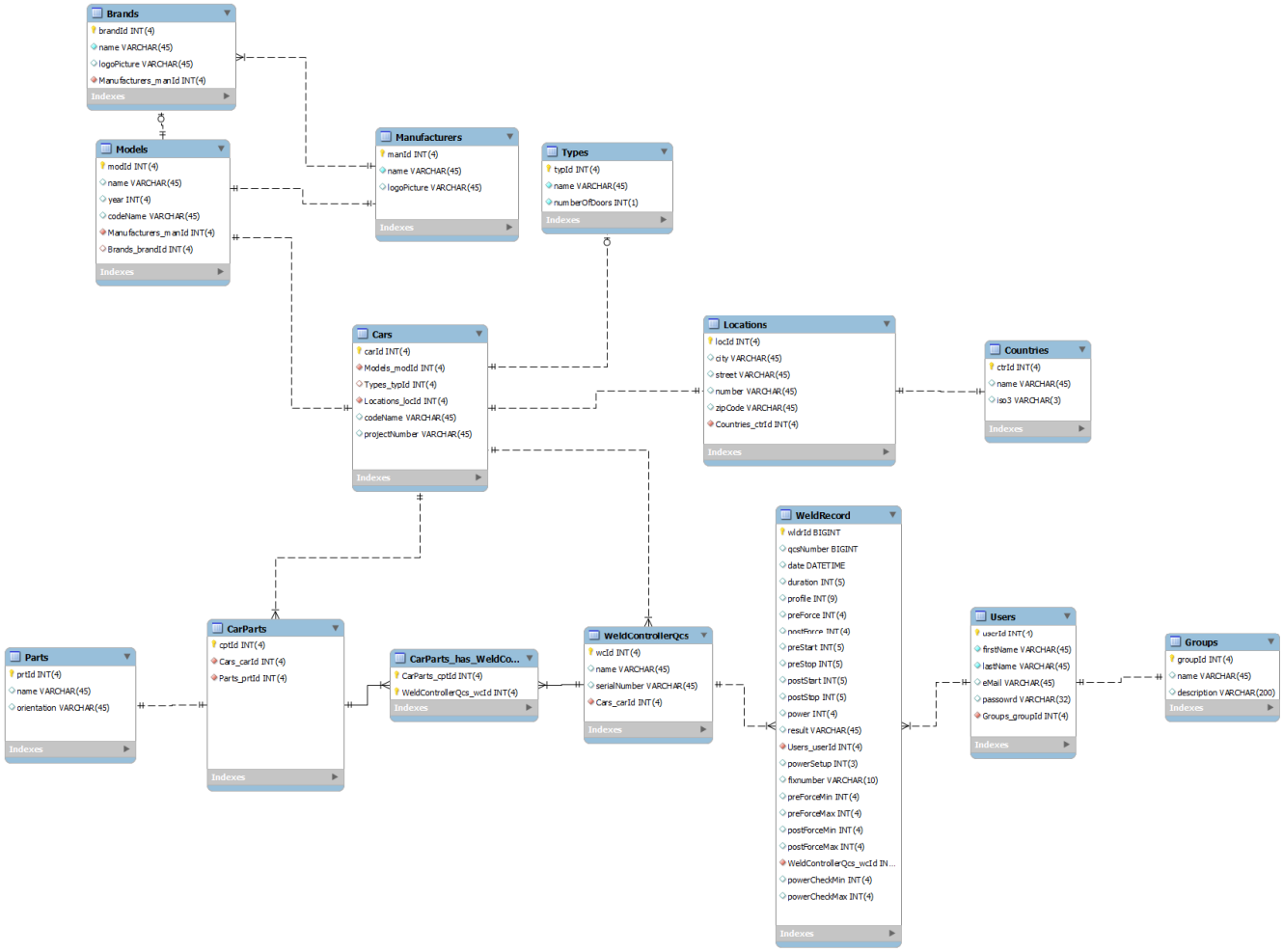


Figuur 10, Voorbeeld productie locatie.

CarId	Manufacturer	Brand	Model	Type	Year	CodeName	Country	City
0002	Volkswagen Group	Audi	A3	Sportback, 4 door	2011	A373	Germany	Ingolstadt

Figuur 11, "Car" op een bepaalde productie "Location"

Nadat alle feiten van het business model bekend waren is er een ERD[9] ontworpen.



Figuur 12, Entity Relation Diagram Weld Analysis System

Het ERD is zo ontworpen dat er met de beschikbare feiten altijd een unieke "Car" gecreëerd kan worden. Een "Car" bestaat tenminste uit een Manufacturer, Model en een Location. Bij "Model" is "Brand" niet verplicht en bij "Cars" is "Type" niet verplicht omdat dit regelmatig niet bekend is. Figuur 11 is een voorbeeld van een "Car" op een bepaalde "Location":
Als er een "Car"gecreëerd is kan hieraan een "Weldcontroller" gekoppeld worden. Aan de "Car"dienen ook de onderdelen die gelast worden door AL-S Technology ("CarParts") gekoppeld te worden. De "CarParts" kunnen vervolgens aan een "Weldcontroller"gekoppeld worden. Zo is dus van elk onderdeel ("CarPart") bekend bij welke weldcontroller die thuis hoort.

[9] Bijlage D, bevat een grotere versie van de ERD afbeelding.

Na de use cases van iteratie 1 besproken te hebben met opdrachtgever is het ERD gewijzigd. In eerste instantie was de tabel brands niet aanwezig. En was manufacturer direct met model verbonden. De manufacturer is het concern en brands zijn de merken die ze voeren. Bijv. Volkswagen Group voert verschillende merken zoals Audi, Skoda enz.

In de Construction fase is het ERD nogmaals gewijzigd. CarParts en WeldControllerQcs is een veel op veel relatie, dit was in eerste instantie een één op veel relatie. Hier is dus een koppeltabel tussen geplaatst.

Voor alle tabellen is een use case gemaakt die CRUD functionaliteiten bevatten. CRUD staat voor Create, Read, Update en Delete. De use cases worden benoemd in het Use Case Model[10]
De Actors (gebruikers) zijn een administrator en een user. De administrator heeft de rechten om alle use case uit te voeren. De user kan alleen maar logfiles uploaden, data inzien en analyseren van een bepaalde dataset.
De “Countries” tabel is gevuld met een kant en klaar MySql script. Hierin staan alle landen met de bijbehorende ISO3 afkortingen.

[10] Bijlage B bevat het Use Case Model

8 Usability

Een belangrijke voorwaarde voor het Weld Analysis System is dat het erg gebruiksvriendelijk is. Een gebruiker moet in een oogopslag kunnen zien wat het programma te bieden heeft en hoe het functioneert. Gebruikers moeten niet lang na hoeven denken over de functies en mogelijkheden.

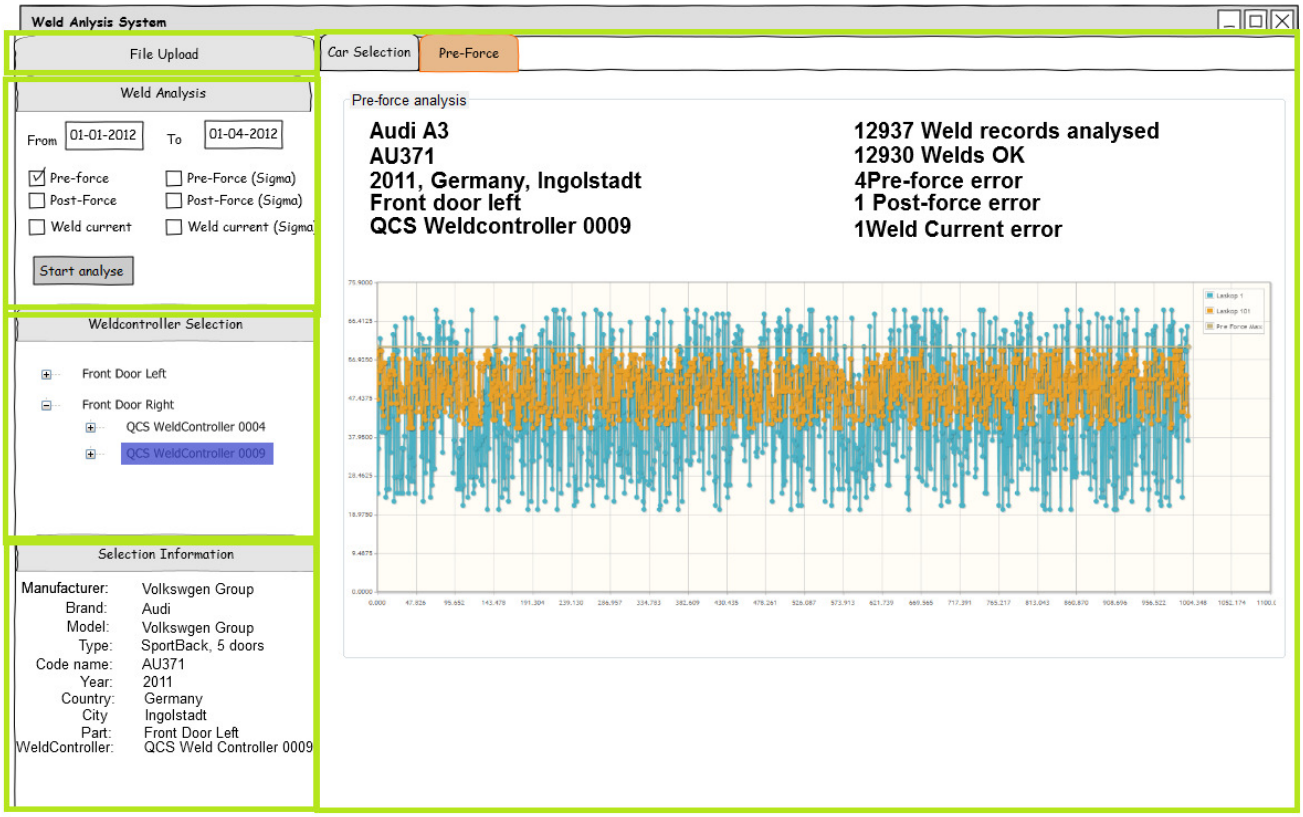
Functies die veel gebruikt worden moeten beschikbaar zijn in het hoofdscherm. Functies die de beheerder gebruikt kunnen in een menu geplaatst worden, zodat de normale gebruikers hier niet door afgeleid worden.

Door het gebruik van conventies kan de usability[11] enorm toenemen omdat gebruikers hier aan gewend zijn, en ze dus meer intuïtief kunnen handelen. Verder moet er gedacht worden aan de volgende punten:

- de bediening moet consequent zijn
- het systeem moet helpen voorkomen dat de gebruiker een fout maakt
- de bediening moet logisch zijn

Er worden nu een aantal voorbeelden gegeven van de usability conventies die in de applicatie gebruikt zijn :

Verdeel pagina's in duidelijk definieerbare gebieden
Hierdoor ziet een gebruiker duidelijk wat hij kan doen in de applicatie. In figuur 13 zijn duidelijk de gebieden (groen omlijnd) te zien die voor een gebruiker van belang zijn.



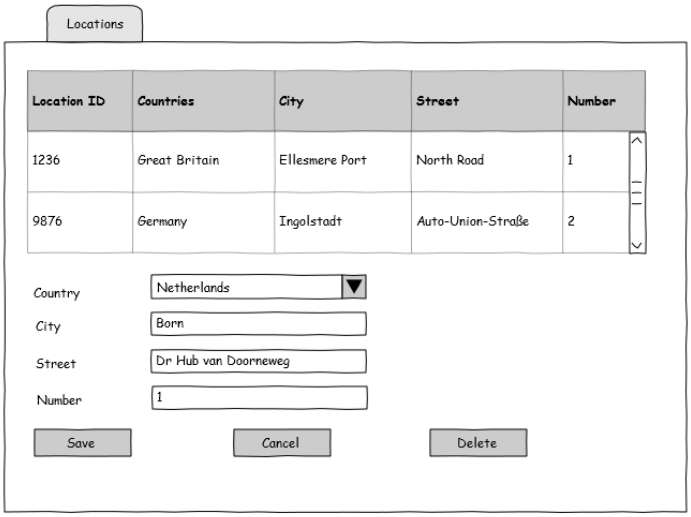
Figuur 13, User interface verdeeld in gebieden.

[11] Bron Don't make me think, second edition, Steve Krug

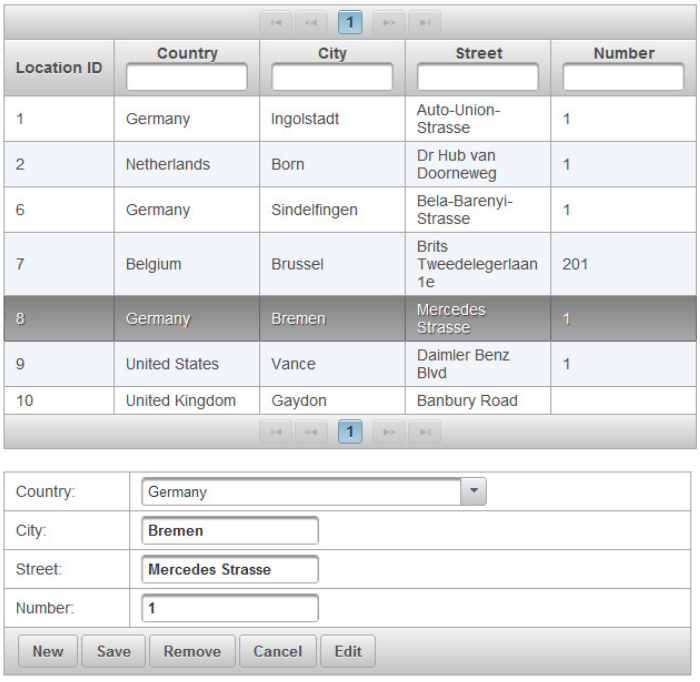
Zorg voor rustige pagina's zonder veel storing
De applicatie moet er helder en duidelijk uitzien zodat gebruikers niet afgeleid worden.

Geef duidelijk aan waar een gebruiker zich bevind
In figuur 13 is het tabblad pre-force anders gekleurd. Dit is een indicatie dat de gebruiker zich op het betreffende tabblad bevindt. Ook is te zien dat bij "weldcontroller selection" de gekozen weldcontroller blauw gekleurd is. Verder is links onderin het scherm "selection information" dit geeft nog eens duidelijk aan wat er geselecteerd is, welke klant , locatie etc.

Formulieren
Voor het gebruik van formulieren is het van belang dat alle velden en labels netjes zijn uitgelijnd. Het gebruik van drop down boxen wordt alleen gebruikt voor zaken waarvan de gebruiker van te voren al weet wat er gekozen moet worden. De dropdown box moet in alfabetische volgorde geordend worden.



Figuur 14, Ontwerp schets van formulier locations met dropdown box en uitgelijnde velden en labels



Figuur 15, Formulier locations zoals die geïmplementeerd is.

9 Elaboration Fase

In dit hoofdstuk wordt de Elaboration fase beschreven in samenhang met het project. De elaboration fase heeft als doel om de volgende punten positief bevestigd te krijgen:

Hebben we een gedetailleerd beeld van de meest kritische requirements?
Dit is het geval, alle use cases zijn bekend en opgesomd in het use case model. Alle use cases zijn uitgewerkt in use case specification document[12]. Zo zijn er ook schetsen van de user interface in het use case specification document.

Is de ontwikkelomgeving ingericht en inorde bevonden?
De ontwikkelomgeving is ingericht en getest (zie Hoofdstuk 10). De test is uitgevoerd door een eenvoudige webapplicatie te bouwen. De volgende onderdelen zijn hiervoor gebruikt:

- Database met twee tabellen
- Hibernate
- GlassFish webserver en
- PrimeFaces

Alles is geïntegreerd in de Netbeans IDE.

Hebben we een stabiele architectuur in werkende code?
De architectuur is getest in de use case “Beheren locations” (Hoofdstuk 11.3) en in orde bevonden. Dit bevestigd dat er met de bedachte architectuur de applicatie verder ontwikkeld kan worden.

Hebben we de belangrijkste risico’s overwonnen?
De risico’s waren klein. Maar er bestond een kans dat met de gekozen frameworks en technieken het niet ging lukken. Mede omdat alle technieken nieuw zijn voor mij was dit een risico. Maar het risico is zo klein mogelijk gehouden door gangbare technieken en frameworks te gebruiken waar veel informatie over te vinden is. De architectuur is in orde bevonden dus deze vraag kan met ja beantwoord worden.

Wordt de business case nog gehaald?
De Business case wordt zeker gehaald. Maar de complete business case zoals die in het plan van aanpak voorgesteld is zal niet compleet binnen de gestelde termijn gerealiseerd zijn. Het schrijven van alle code voor iteratie twee zal vertraagd opgeleverd worden. In hoofdstuk 15 zijn de oorzaken hiervan verder beschreven

[12] Bijlage C bevat een selectie van de belangrijkste Use Case Specifications.

10 Ontwikkelomgeving

In dit hoofdstuk wordt beschreven hoe de ontwikkelomgeving tot stand is gekomen.

10.1 Inrichten Ontwikkel Omgeving

Het doel van het inrichten van de ontwikkelomgeving is om een probleemloze ontwikkelomgeving te creëren. De ontwikkelomgeving moet stabiel draaien, het moet voorkomen worden dat er in een later stadium problemen ontstaan omdat de ontwikkelomgeving niet in orde is.

De IDE (Integrated development environment) die gebruikt is, is Netbeans 7.1. Netbeans bevat standaard de volgende frameworks en applicaties die voor het project benodigd zijn :

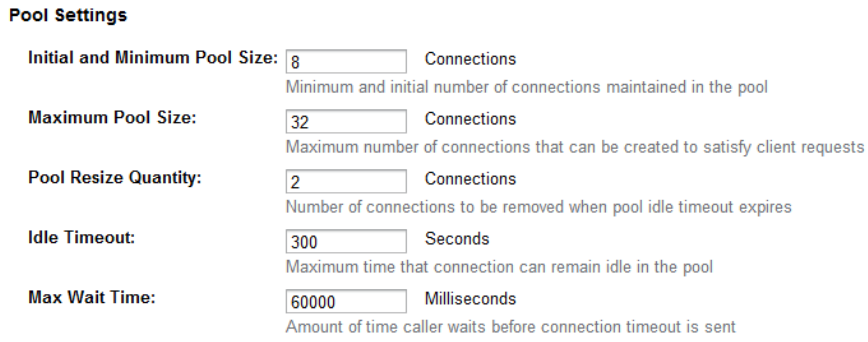
- Java EE 6
- Java Server Faces (JSF 2.0)
- Hibernate 3.2.5 Framework
- GlassFish Server Open Source Edition 3.1.2

10.1.1 Hibernate

Het Hibernate framework wat met Netbeans meegeleverd wordt is sterk verouderd, deze release komt uit 2007. Daarom is Hibernate geüpdate naar de laatste versie 4.1. Om Hibernate te updaten moeten de JAR files in de Libraries map overschreven worden met de nieuwe versie. Het is daarna noodzakelijk om sommige methodes te herschrijven omdat die niet meer functioneren.

10.1.2 Database MySql

MySql Database is in GlassFish geconfigureerd in een zogenaamde “JDBC connection pool”. Zo hoeft de applicatie niet zelf te communiceren met de database. Dit heeft veel voordelen met betrekking tot de database connecties, zo regelt GlassFish het maximum aantal connecties en de time out enz.



Figuur 16, Instellingen van connection pool in GlassFish

Voordat de MySql server in de connection pool geconfigureerd kan worden dient er nog een connector “mysql-connector-java…..” in de MySql lib folder geplaatst te worden. Deze zorgt ervoor dat GlassFish kan communiceren met MySql database.

10.1.3 PrimeFaces

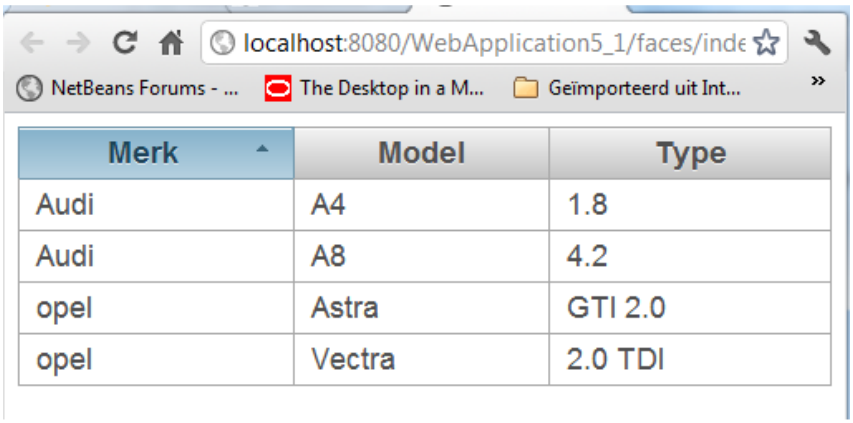
PrimeFaces is een JSF component suite. Om PrimeFaces te gebruiken hoeft alleen de JAR file van PrimeFaces in de libraries map van het project geplaatst te worden. En in de html declaratie van de XHTML pagina dient deze regel toegevoegd te worden :
“xmlns:p=http://primefaces.org/ui”
Nu kunnen alle PrimeFaces componenten gebruikt worden door de prefix P te gebruiken bij de declaraties. Bijv. <p:dataTable>.

10.1.4 Testen ontwikkelomgeving

Nadat alle onderdelen van de ontwikkelomgeving geïnstalleerd waren, is de ontwikkelomgeving getest door een eenvoudige webapplicatie. De test applicatie bestond uit de volgende onderdelen:

- Twee tabellen in MySql database.
- Hibernate die de mapping verzorgt tussen de database en de JAVA objecten.
- Hibernate communiceert met de JDBC Connection pool in GlassFish.
- De XHTML pagina is geconfigureerd voor gebruik van PrimeFaces.

Dit bleek allemaal goed te werken.
Het doel om een stabiele en werkende ontwikkelomgeving te realiseren is gehaald.



Figuur 17, Simpele testapplicatie

11 Architectuur

In dit hoofdstuk wordt de architectuur besproken zoals die voor de use cases gebruikt is. Omdat alle technieken nieuw voor mij zijn is het van belang om te bewijzen dat de bedachte architectuur ook daadwerkelijk functioneert. Dit is ook het doel van de elaboration fase in RUP, om de architectuur in werkende code te laten functioneren.
In de volgende paragrafen wordt eerst de architectuur besproken, vervolgens wordt in paragraaf 11.3 use case “beheren locations” besproken. Deze use case is gebruikt om te testen of alles functioneert zoals gepland.

11.1 MVC Model

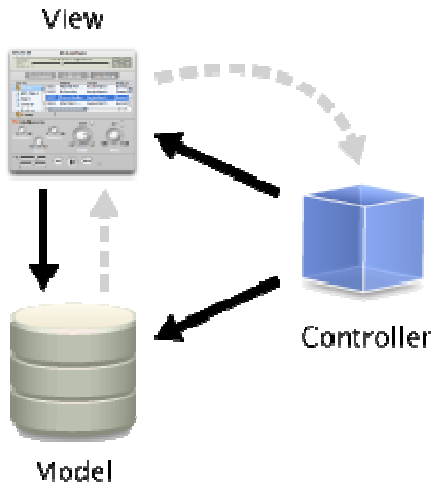
De applicatie is ontworpen volgens het MVC model (Model View Controller) . Dit houdt in dat de applicatie in 3 onderdelen wordt opgedeeld namelijk :

- Datamodel model
- Dataview view
- Applicatielogica controller

Het doel van dit model is om code van user interface te scheiden van het model. De code wordt beter leesbaar en veranderingen in user interface hebben niet direct gevolgen voor het model en vice versa.

De controller is de regelaar tussen view en model. Het model heeft geen kennis van controller of view. De controller kent het model en de view. De view kent via de controller het model. Het voordeel is dat er gemakkelijk nieuwe views gemaakt kunnen worden zonder het model aan te passen.

De controller reageert op events die van de view komen. Drukt een gebruiker op een knop dan komt er bij de controller een event binnen waarop gereageerd kan worden.



Figuur 18, Schematische weergave MVC model.

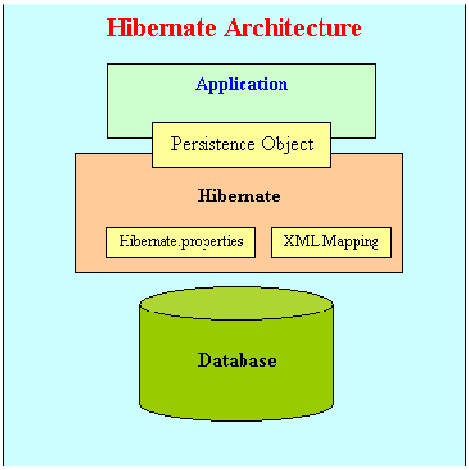
Om de gegevens van het model persistent te maken moeten deze in een database opgeslagen worden. Om een efficiënte koppeling te maken tussen model en database is het Hibernate framework gebruikt. In de volgende paragraaf wordt Hibernate verder beschreven.

11.2 Hibernate

Hibernate verzorgt de communicatie tussen de database en de applicatie. Alle tabellen in de database worden door Hibernate als JAVA objecten toegankelijk gemaakt in de applicatie. Ook de complete relationele mapping wordt door Hibernate verzorgd. Als er een entiteit als object geladen wordt dat een relatie heeft met een andere entiteit, dan heeft Hibernate deze relatie meegenomen en is benaderbaar via het object. Bijvoorbeeld:

Het object Brand : Audi heeft een relatie met Manufacturer. In het object Audi is de methode getManufacturer() beschikbaar. Deze zal in dit geval “Volkswagen Group” terug geven.

De mapping van de database entiteiten naar objecten gaat via een XML file genaamd Hibernate.cfg.xml. In deze file staan alle entiteiten die naar JAVA objecten gemapd moeten worden. Vervolgens heeft elk object een XML file genaamd yyy.hbm.xml (yyy staat voor de object naam). In deze file staan alle kenmerken van de tabellen en attributen.



Figuur 19, Schematische weergave Hibernate framework

Hibernate vraagt aan GlassFish naar beschikbare connecties (dit is reeds uitgelegd in paragraaf 10.1.2). In GlassFish is een connection pool beschikbaar met een max. aantal connecties die beschikbaar zijn voor de applicatie. Zo kunnen meerdere gebruikers tegelijkertijd de database benaderen.

Doordat Hibernate de complete communicatie met de database verzorgd is er bijna geen SQL code nodig om queries op te vragen. Voor het opvragen van alle “Cars” kan volstaan worden met deze regel code:

```
Query q = session.createQuery("from Car");
```

Deze Query kan vervolgens naar een List gecast worden :

```
carList = (List<Car>) q.list();
```

Er is nu een JAVA list beschikbaar met alle Cars als objecten, hier kunnen nu eenvoudig bewerkingen op uitgevoerd worden d.m.v. de getter en setter methodes .

Bijv. methodes getCodeName() en setCodeName() :

```
public String getCodeName() {
	return this.codeName;
}

public void setCodeName(String codeName) {
	this.codeName = codeName;
}
```

Het maken van nieuwe records in de database werkt ook d.m.v. JAVA objecten. Door een nieuw object aan te maken en deze dan te saven wordt er een nieuwe record in de database gezet. Hibernate interpreteert dit en stuurt SQL query naar de database.
Dit is de methode voor het creëren van nieuwe records :

```
session.save(car);
```

Wijzigen van records gaat via een update() methode.

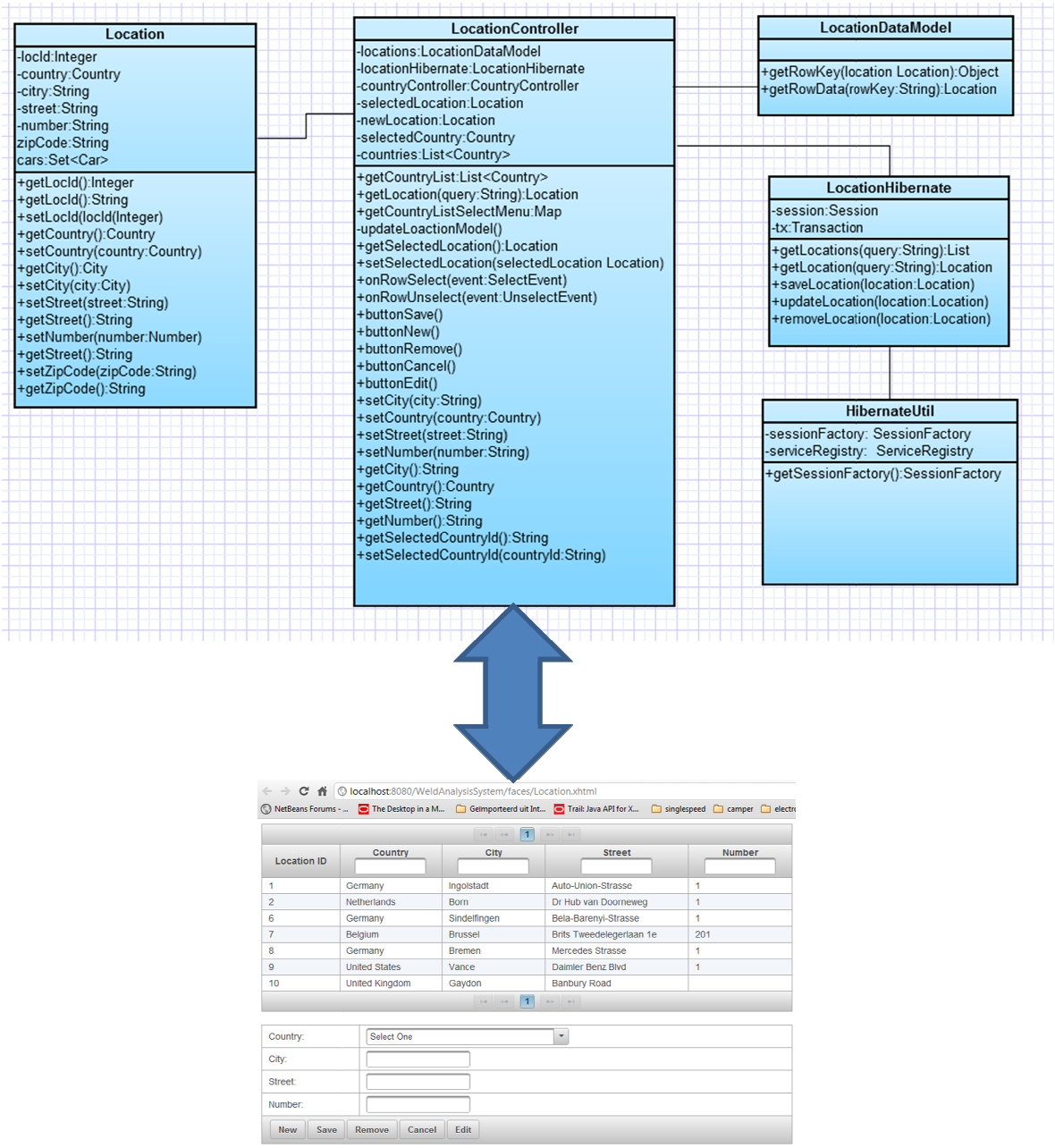
```
session.update(car);
```

11.3 Architectuur test "Location beheren"

Het MVC model wordt nu geïmplementeerd in de use case "location beheren". Er zal dus nu een werkende use case ontstaan die CRUD operaties kan uitvoeren. Dit bevestigd dat alle onderdelen in de architectuur functioneren en er op deze manier verder gewerkt kan worden.

De locationController functioneert als @ManagedBean. De webpagina Location.xhtml communiceert met de locationController.
De tabel met locaties is opgebouwd uit het LocationDataModel. Dit is een PrimeFaces component dat objecten terug geeft als er bijvoorbeeld een rij geselecteerd wordt.

Als de pagina geopend wordt dan haalt de locationController via LocationHibernate en HibernateUtil de beschikbare locaties uit de database. Vervolgens worden de locaties in het LocationDataModel gezet wat de tabel op de pagina genereert.



Figuur 20, Klasse diagram en webpagina van use case location beheren

Als er een rij geselecteerd word dan wordt via een AJAX event de methode onRowSelect(SelectEvent event) aangeroepen.

```
<p:ajax event="rowSelect" listener="#{locationController.onRowSelect}" update=":form"/>
<p:ajax event="rowUnselect" listener="#{locationController.onRowUnselect}" update=":form"/>
```

Al er een nieuwe locatie gemaakt wordt dan creëert de locationController een nieuw location object. Dit location object wordt na het indrukken van de save button via LocationHibernate in de database opgeslagen.

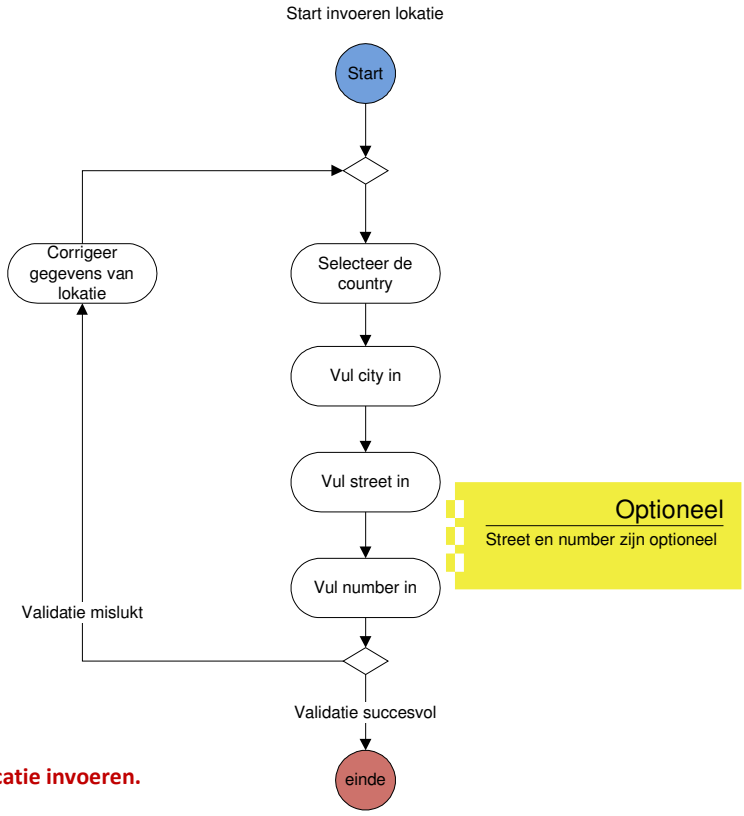
11.4 Functioneel design

Als de gebruiker het formulier locations opent worden automatisch de beschikbare locations uit de database gehaald. Deze locations worden vervolgens in de location tabel getoond. De gebruiker heeft nu verschillende mogelijkheden :

- Nieuwe location aanmaken
- Bestaande location verwijderen
- Bestaande location wijzigen

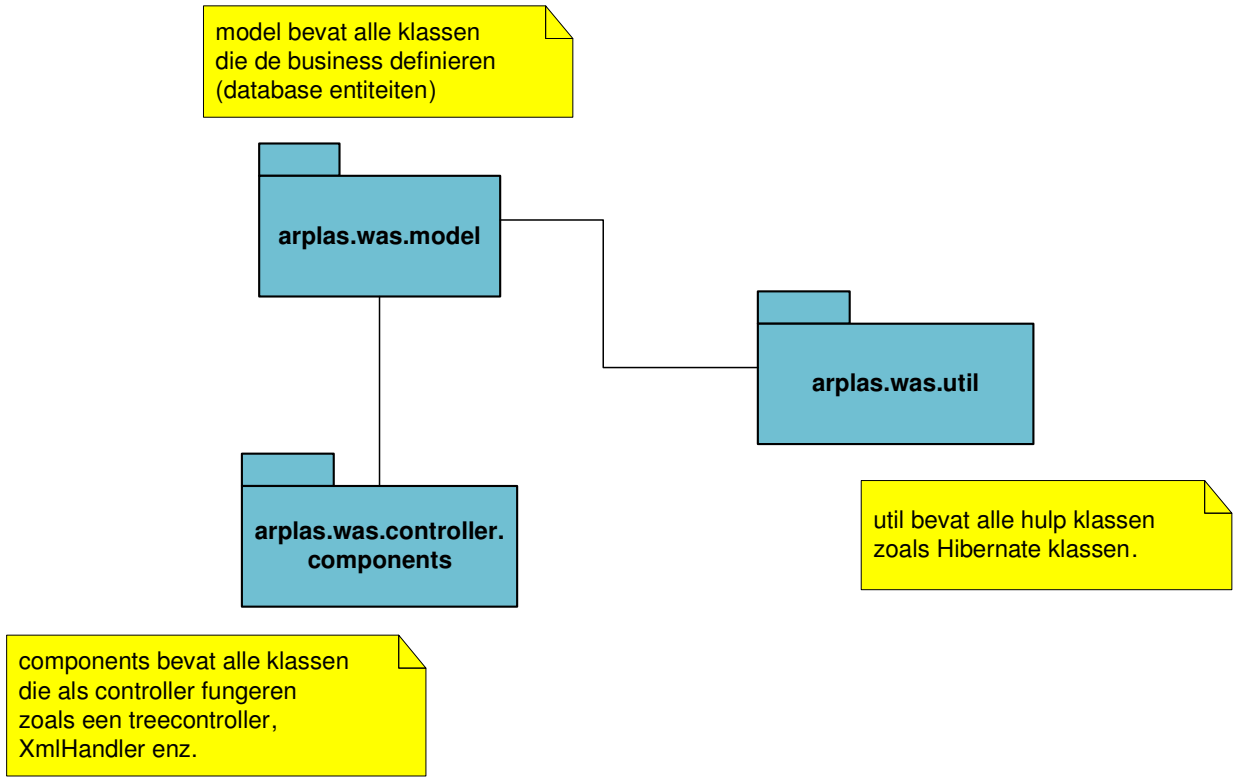
In figuur 22 is te zien welke handelingen een gebruiker doet om een nieuwe locatie in te voeren. Als de gebruiker het formulier opent wordt de dropdown lijst met countries geladen. In de database staan alle landen van de wereld.
Vervolgens vult de gebruiker de city in, dit is verplicht. De velden street en number zijn niet verplicht omdat deze soms niet bekend zijn.

Als de gebruiker op de save knop drukt worden de gegevens gevalideerd. De nieuwe locatie wordt opgeslagen in de database. De tabel wordt ververst via AJAX zodat de nieuwe locatie in de tabel getoond wordt.



Figuur 21, Flowchart nieuwe locatie invoeren.

Onderstaande afbeelding geeft een overzicht van hoe de klassen ingedeeld zijn in packages. Er is een duidelijk onderscheid gemaakt in het model (de business) en de controller en utility klassen.



Figuur 22, Overzicht van de packages

12 Design van Applicatie iteratie 2

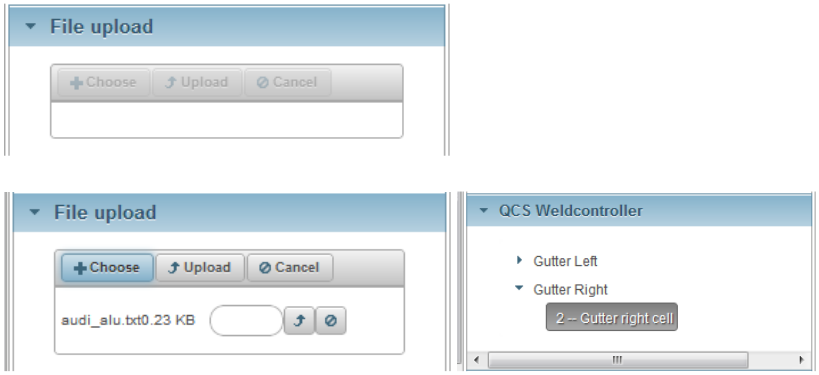
In dit hoofdstuk wordt het design van de use cases van iteratie 2 besproken.

12.1 fileupload

Fileupload is een functie waar de gebruiker XML files kan uploaden. De fileupload module is een standaard PrimeFaces component. Het heeft een aantal krachtige functies zoals :

- client side controle van type file, in dit geval zijn alleen XML files toegestaan
- client side controle van grootte van file.
- Drag and drop ondersteuning, vanuit Windows kan er een bestand gesleept worden in de file upload module.

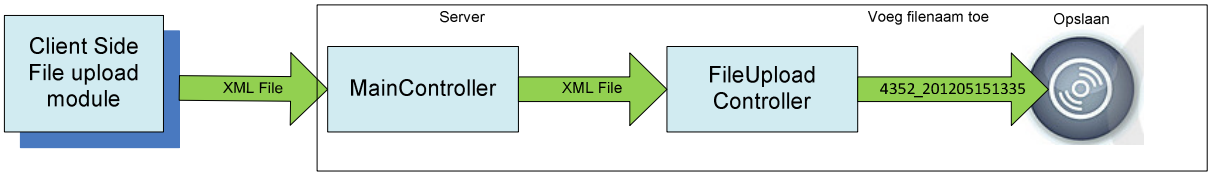
Voordat er een file geupload wordt, is het van belang dat er een QcsWeldController geselecteerd is, anders is niet bekend waar de logfile vandaan komt. De file upload module is gedisabled totdat er een QcsWeldController geselecteerd is.



Figuur 23, Fileupload boven "disabled" en onder "enabled"

Als de file uploadt module enabled is kan de gebruiker een file selecteren, via "Choose" of de gebruiker kan een file in het gebied slepen. Als de file geupload is wordt die opgeslagen op een bepaalde plaats op de server. De file krijgt de volgende naam QcsWeldControllerId + TimeStamp. Bijv. 4352_201205151335.

De file upload module communiceert via de MainController met de FileUploadController.



Figuur 24, Schematische weergave architectuur fileupload

De FileUploadController ontvangt een FileUploadEvent als de file geupload is. Dit event bevat de file, die krijgt vervolgens een naam afhankelijk van geselecteerde QcsWeldController.

```
public void handleFileUpload(FileUploadEvent event) {
        UploadedFile file = event.getFile();
        …
}
```

Als de file succesvol opgeslagen is wordt via de MainController de XmlHandler gestart zie paragraaf 10.2.

12.2 XML Handler

De XmlHandler zorgt ervoor dat de lasdata uit de geüploade XML file gehaald wordt. Niet alle parameters uit de XML file worden gebruikt, er is een selectie gemaakt vanwege performance overwegingen. Sommige parameters zijn niet nodig andere parameters zijn af te leiden. Voor een volledig overzicht zie bijlage E.

De StAX XML parser is een zogenaamde Pull Parser. Dit houdt in dat de applicatie vraagt aan de XML parser om informatie te geven uit de XML file. De applicatie vraagt steeds om het volgende event in de XML file.

Figuur 25, Schematische weergave werking StAX parser

De XML file bestaat uit events waarop de parser kan reageren.

De volgende events worden gebruikt om de benodigde informatie uit de XML file te halen :

- StartElement
- EndElement
- Characters

```
XMLInputFactory xmlFactory = XMLInputFactory.newInstance();
FileInputStream fileInputStream = new FileInputStream(fileName);
XMLEventReader xmlReader = xmlFactory.createXMLEventReader(fileInputStream);
```

Bovenstaande regels code declareren respectievelijk een XMLInputFactory, FileInputStream en een XMLEventReader.
De XMLInputFactory is de basis klasse, deze wordt gebruikt om de xmlReader te declareren. De fileInputStream is noodzakelijk om de XML file in te lezen.

```
while (xmlReader.hasNext()) {
        XMLEvent xmlEvent = xmlReader.nextEvent();
        if(xmlEvent.isStartElement()) {

          if (xmlEvent.asStartElement().getName().getLocalPart().equals("Log")) {
            weldRecordController.setNewWeldRecord();
          }

          else if (xmlEvent.asStartElement().getName().getLocalPart().equals("lg_qcsnumber")) {
            xmlEvent = xmlReader.nextEvent();
            weldRecordController.getNewWeldRecord().
            setQcsNumber(Long.parseLong(xmlEvent.asCharacters().getData()));
          }
```

Bovenstaande code geeft het principe weer van het parsen van de XML file.

Met *xmlReader.hasNext()* wordt de complete XML file doorlopen.

Om de events te detecteren die van belang zijn wordt deze methode gebruikt :
if (xmlEvent.asStartElement().getName().getLocalPart().equals("lg_qcsnumber")).

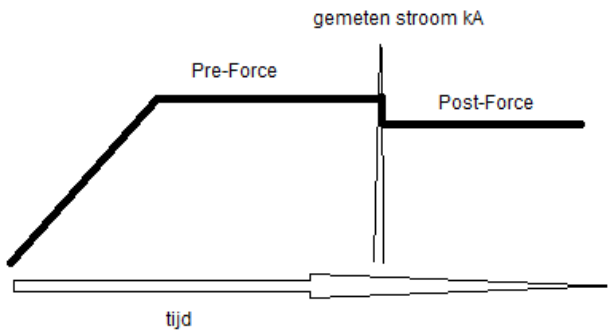
Vervolgens kan met *xmlEvent.asCharacters().getData()* de betreffende informatie gelezen worden. Deze informatie wordt dan via een set methode in een weldRecord object gezet. Dit WeldRecord object wordt vervolgens in de database opgeslagen.

12.3 data-analyse

Het data analyse gedeelte van de applicatie is voor de gebruiker het belangrijkste onderdeel. De gebruiker kan hier een analyse maken van een bepaalde hoeveelheid lasdata.

Nu wordt kort uitgelegd wat de analyse inhoudt in relatie met het AL-S Technology lassysteem.
Het lassysteem kent 3 belangrijke parameters die bepalen of de las een goede of slechte las is.

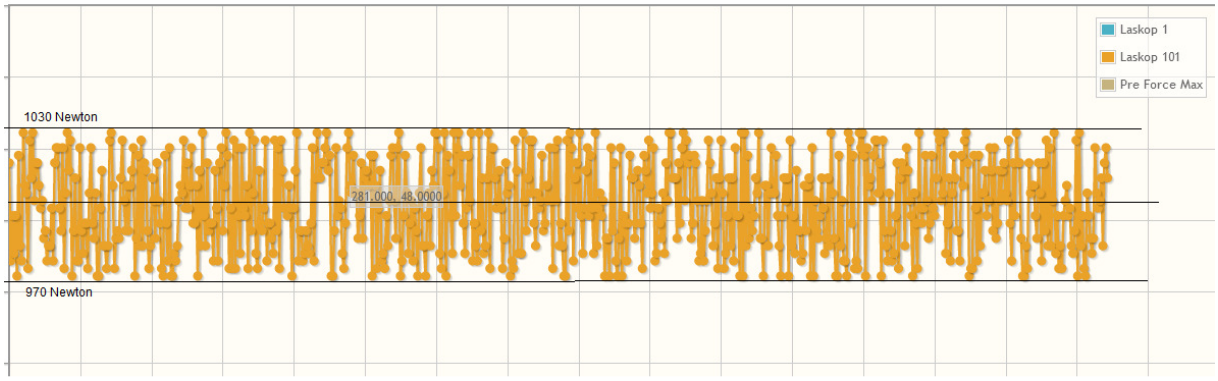
- Pre-force
- Post-Force
- Gemeten stroom in kA.



Figuur 26, AL-S Technology lasproces

Voor deze 3 parameters zijn er 2 analyse functies beschikbaar. Een lijndiagram die de waarden in de tijd plaatst. Een Histogram waar de waarden in klassen gegroepeerd worden en vervolgens wordt er per groep een frequentietelling gedaan. Hier kan men zien hoe stabiel het proces is.

Voorbeeld lijndiagram pre-force:
Er zijn 10 000 lasrecords beschikbaar van een bepaalde lascontroller. Deze lascontroller last 4 laspunten. Voor elk laspunt zijn er dan 2500 lasrecords beschikbaar. Als we de pre-force nemen worden alle pre-force waarden per laspunt in een lijndiagram geplaatst. Met op de x-as de tijd en op de y-as de pre-force waarde.
Het is belangrijk dat het lassysteem zo stabiel mogelijk last, in dit lijndiagram is goed te zien hoe het systeem zich gedraagt ten opzichte van de tijd. Onderstaande afbeelding geeft een redelijk stabiel systeem weer met een spreiding van 70 Newton. Maar door verschillende factoren kunnen de afwijkingen veel groter zijn. Denk aan slechte lasdelen met hoge toleranties, trillende robots, enz.

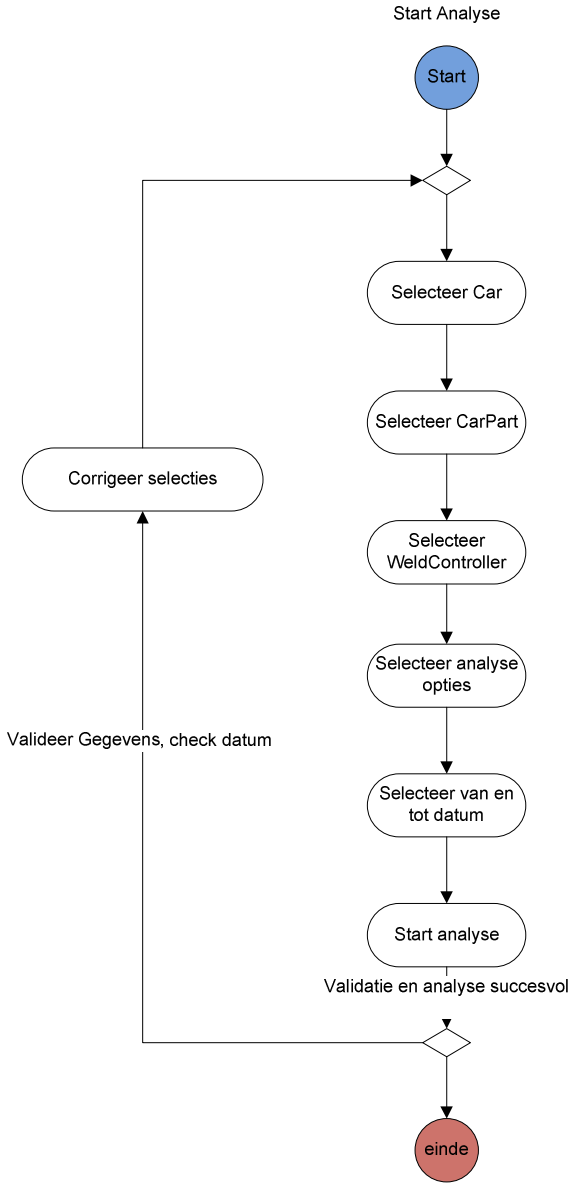


Figuur 27, Pre-force in de tijd weergegeven, elk punt is een las.

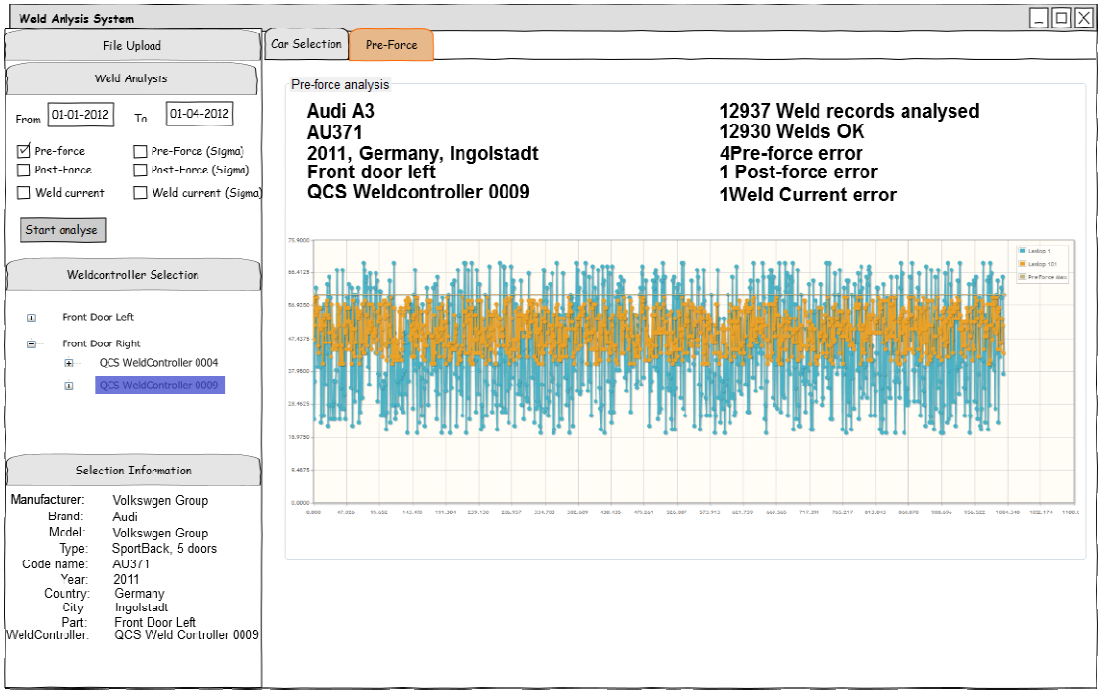
12.4 Functioneel design.

De gebruiker genereert de grafieken door een Car te kiezen , vervolgens in het linker menu een weldcontroller te kiezen. Nu geeft de gebruiker de periode, de analysefuncties en dan start analyse. De grafieken worden gegenereerd, rechts boven in de grafiek kan de gebruiker laspunten aan of uitzetten.

Aan de hand van de geselecteerde weldcontroller en de aangegeven periode worden de lasrecords uit de database gehaald. Met de dataset die opgehaald is uit de database worden de grafieken gegenereerd.
De gebruiker heeft de analyse functies geselecteerd, alleen van de geselecteerde functies worden grafieken gegenereerd. Deze grafieken zijn dan vervolgens op tabbladen beschikbaar.

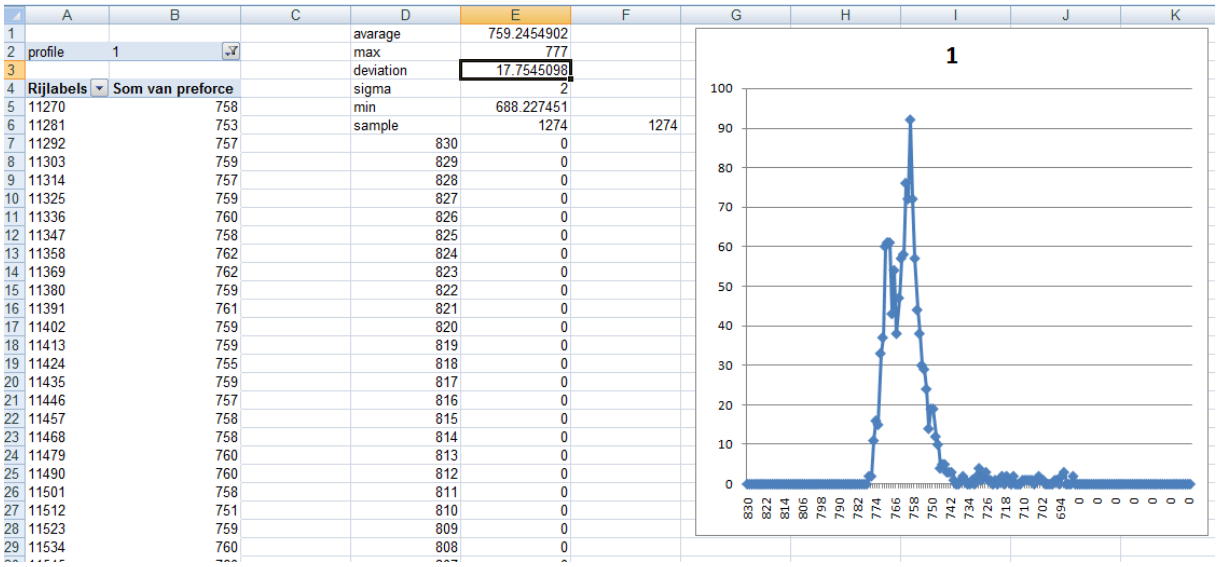


Figuur 28, Handelingen die een gebruiker uitvoert voor het analyseren van een logfile



Figuur 29, User Interface analyse functies

Voor de Histogram functie is op moment van schrijven nog geen voorbeeld beschikbaar. Maar om een idee te krijgen is het Excel werkblad toegevoegd zie figuur dertig.



	A	B	C	D	E	F
1				avarage	759.2454902	
2	profile	1		max	777	
3				deviation	17.7545098	
4	Rijlabels	Som van preforce		sigma	2	
5	11270	758		min	688.227451	
6	11281	753		sample	1274	1274
7	11292	757		830	0	
8	11303	759		829	0	
9	11314	757		828	0	
10	11325	759		827	0	
11	11336	760		826	0	
12	11347	758		825	0	
13	11358	762		824	0	
14	11369	762		823	0	
15	11380	759		822	0	
16	11391	761		821	0	
17	11402	759		820	0	
18	11413	759		819	0	
19	11424	755		818	0	
20	11435	759		817	0	
21	11446	757		816	0	
22	11457	758		815	0	
23	11468	758		814	0	
24	11479	760		813	0	
25	11490	760		812	0	
26	11501	758		811	0	
27	11512	751		810	0	
28	11523	759		809	0	
29	11534	760		808	0	

Figuur 30, Histogram in het Excel werkblad.

12.4 Berekeningen data analyse

Lijn diagram
Hier zijn geen berekeningen voor nodig, de waarden worden in een “chart model”gezet. De x-as is de tijd, de y-as pre-force, post-force of gemeten stroom.

Met onderstaande code worden de grafieken opgebouwd, er wordt eerst een ChartModel gedeclareerd. In dit model kunnen dan ChartSeries geplaatst worden. Deze zullen dan uiteindelijk op de webpagina getoond worden.

```
preForceModel = new CartesianChartModel();
weldGun_1 = new ChartSeries();
preForceModel.addSeries(weldGun_1);

weldGun_1.set(timeDate, preForce);
```

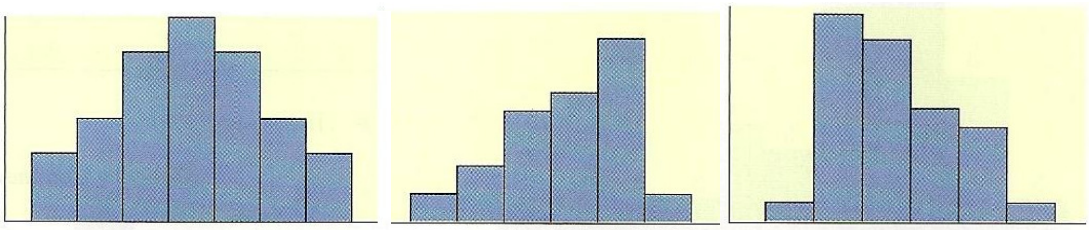
Afhankelijk van het aantal laskoppen kunnen deze aan het model toegevoegd worden. De gebruiker kan deze vervolgens eenvoudig rechts boven in de grafiek aan of uitzetten.

Histogram

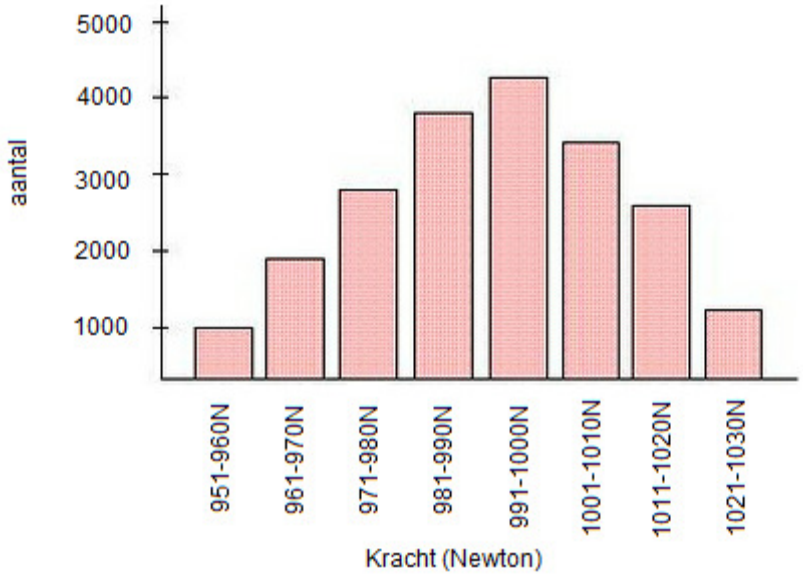
Het histogram geeft de frequentie weer van de lasparameters (pre-force, post-force, gemeten stroom). De lasdata wordt verdeeld in groepen, bijv. pre-force 990-1000N. Er wordt geteld hoe vaak deze waardes voorkomen in de dataset dit wordt vervolgens in een staaf diagram getoond.

Er kunnen bepaalde conclusies getrokken worden uit een Histogram:

- Wat is de meest voorkomende waarde?
- Is de data symmetrisch of heeft het een negatieve of positieve verschuiving?
- Wat is de variatie, een smal histogram heeft een kleine variatie t.o.v. een breed histogram



Figuur 31, Links een Symmetrisch histogram, midden een negatieve verschuiving, rechts een positieve verschuiving



Figuur 32, Histogram pre-force van lasproces

Voor het genereren van een histogram zijn een aantal waarden en berekeningen nodig namelijk:

- Gemiddelde waarde van dataset
- Minimum waarde dataset
- Maximum waarde dataset
- Modus (waarde met de hoogste frequentie in de dataset)
- Mediaan (het midden van de dataset)

Deze waarden kunnen extra informatie verstrekken over de interpretatie van het histogram.

Het midden van het Histogram wordt bepaald met de Modus. Dit is de waarde die het meest voorkomt in de dataset. Logischerwijs zullen de waarden die aan deze waarden grenzen dan vervolgens het meest voorkomen.

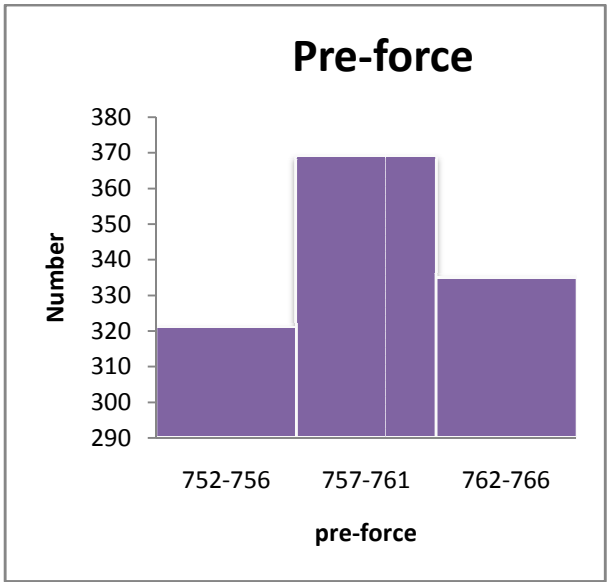
Na analyse van een dataset van een Robot applicatie Daimler USA zijn de volgende waarden gevonden. Het betreft hier één laspunt. De complete dataset bestaat uit 11 laspunten.

Aantal records	1274
Gemiddelde	759
Max	777
Min	688
Spreiding (Max-Min)	89

Pre-Force	756	757	758	759	760	761	762	763	764
Frequentie	44	57	72	92	72	76	58	57	47

Figuur 33, In het geel de Modus.

De breedte van de groepen moet instelbaar zijn. De default voor de breedte wordt 5 waarden, -2 en +2 van de Modus. De middelste groep wordt hier dus {757, 758, 759, 760, 761}



Figuur 34, Een impressie van het histogram van bovenstaande dataset (alleen middelste drie groepen).

13 Construction Fase

In dit hoofdstuk wordt de Construction fase beschreven in samenhang met het project. De construction fase heeft als doel om iets te ontwikkelen wat waarde heeft voor de Klant/opdrachtgever. In dit geval moeten er dus use cases ontwikkeld worden. Aangezien het project uit twee iteraties bestaat wordt deze fase twee keer herhaald.

Om de construction fase positief af te sluiten dienen de volgende vragen positief beantwoord te worden:

Is alle functionaliteit gerealiseerd?
Deze vraag kan met nee beantwoord worden. Op moment van schrijven is niet alle functionaliteit gerealiseerd. Binnen een korte termijn zullen overige use cases gerealiseerd worden. De use cases die op dit moment niet gereed zijn:

- Beheren van types
- Beheren van locations
- Inloggen gebruiker
- Analyseren bepaalde dataset

Is het product gereed voor Beta testing?
Deels ja, de use cases die gereed zijn, zijn gereed voor Beta testing

Zijn handleidingen en trainingsmateriaal gereed?
Nee, Dit valt buiten de scope van het project. Deze worden op een later tijdstip gerealiseerd.

14 Conclusie en aanbevelingen

Voor het ontwikkelen van het “Weld Analysis System” zijn een behoorlijk aantal technieken en frameworks gebruikt. Deze technieken en frameworks bepalen uiteindelijk het resultaat van de applicatie qua performance en usability.

De keuze voor de applicatie server GlassFish is een erg goede keuze geweest. De configuratie is helder en eenduidig. Vanaf begin af aan draait GlassFish zonder een enkel probleem in de ontwikkelomgeving.

De combinatie van MySQL, Hibernate en GlassFish zorgt voor een goede performance. Zo beheert GlassFish de database connecties via een connection pool. Hibernate op zijn beurt communiceert prima met deze connection pool. Verder zorgt Hibernate voor een naadloze integratie van de database entiteiten met JAVA objecten.

JSF 2.0 en PrimeFaces als client interface is qua usability uitermate geschikt. PrimeFaces beschikt over zeer krachtige componenten met veel AJAX support. Dit komt de usability ten goede. Een minpunt van PrimeFaces is dat de gebruikershandleiding soms te wensen over laat. Of dat er voorbeelden op de PrimeFaces website[13] niet volledig zijn. Dit kan soms wat uren zoeken en debuggen kosten.
Een ander minpunt van JSF / PrimeFaces is dat sommige componenten nogal wat backing bean code vergen om deze optimaal te gebruiken. Bijv. een drop down menu wat niet direct een object terug geeft als er iets geselecteerd wordt maar een String.
Verder is het programmeren van webpagina’s in XHTML erg duidelijk en overzichtelijk.

Het gebruik van de XML parser StAX is qua performance en integratie erg goed. De testen die uitgevoerd zijn laten dit zien een XML file van 427000 regels wordt gelezen in minder dan 3 seconden.

Het werken met RUP heeft ervoor gezorgd dat er een goede structuur in het project zat. Door het maken van het use case model en use case specification document is er goed nagedacht over functionaliteiten en user interface. Tijdens het ontwikkelen is hier ook nauwelijks meer van afgeweken.

[13] http://www.primefaces.org/showcase-labs/ui/home.jsf

15 Evaluatie

Bij AL-S Technology was behoefte aan een goede en efficiënte analyse tool om las data te analyseren. Dit was een perfect project voor mijn afstudeeropdracht. Het project is in het plan van aanpak gedefinieerd als ontwerp en een werkende applicatie in de ontwikkelomgeving.

De planning die voor dit project gemaakt is in het plan van aanpak was optimistisch maar haalbaar, mits er voldoende tijd beschikbaar was naast mijn normale werkzaamheden. Het eerste gedeelte van het project in RUP termen “Inception fase” ging voortvarend. Samen met de opdrachtgever was er snel overeenstemming bereikt over hoe de applicatie eruit moest zien. En welke functionaliteiten de applicatie moest bevatten.

Het tweede gedeelte van het project was de onderzoeksfase in RUP termen de “elaboration fase”. Ook bij deze fase zijn er weinig problemen opgetreden. De benodigde technieken en frameworks waren binnen de gestelde planning geselecteerd.

Er is wel enige vertraging ontstaan in de derde fase de “Construction fase”. Doordat er in totaal zestien use cases ontwikkeld moesten worden heeft deze fase veel meer tijd in beslag genomen als gepland.
Voor de use cases van iteratie één stond vier weken gepland dit is drie weken uitgelopen. Voor iteratie twee stond vier weken gepland welke dus niet meer volledig beschikbaar waren voor de deadline van de scriptie. Bij het schrijven van deze scriptie zijn niet alle use cases van iteratie twee gereed.

Planning is wel om de ontbrekende use case “analyseren van een bepaalde dataset” gerealiseerd te hebben voor de eindpresentatie.

Het blijkt toch dat het compleet ontwerpen en ontwikkelen van een dergelijke applicatie voor een junior software engineer meer tijd nodig heeft dan 3 maanden. Daarbij komt nog dat door een zeer drukke periode op mijn werk er soms keuzes gemaakt moesten worden. Omdat de kwaliteit niet ter discussie staat en er geen concessies gedaan kunnen worden in keuze qua usability zijn er een aantal use cases vertraagd.

De use cases die op dit moment gerealiseerd zijn zitten technisch en qua usability goed in elkaar. Ik had ook concessies kunnen doen in de keuzes van gebruikte componenten in de user interface. Dit zou de ontwikkeltijd teruggebracht hebben maar de usability niet ten goede zijn gekomen. Ik wil dan ook staan voor kwaliteit in mijn werk.

De functionaliteiten die op dit moment nog niet gerealiseerd zijn zullen binnen afzienbare tijd geïmplementeerd worden. Hierna kan de applicatie na een testfase live gaan, en kunnen de medewerkers van AL-S Technology er gebruik van gaan maken.

16 Bronnen

Usability

- Don't make me think! second edition 2008 Steve Krug

Hibernate

- Harnessing Hibernate 2008 James Elliot
- Hibernate Developer Guide 4.1.3 2012 The JBoss Visual Design Team

PrimeFaces

- PrimeFaces user guide 3.2 2012 Çağatay Çivici
- PrimeFaces ShowCase http://www.primefaces.org/showcase-labs/ui/home.jsf

GlassFish, MySQL, Hibernate

- Using Hibernate in a Web Application http://netbeans.org/kb/docs/web/hibernate-webapp.html#02

Bijlage A Plan van Aanpak

Inhoudsopgave

1 Inleiding

Dit document is het plan van aanpak voor de afstudeer opdracht “Weld Analysis System”. De opdracht wordt uitgevoerd bij het bedrijf AL-S Technology te Amersfoort.

AL-S Technology is tevens mijn werkgever, ik ben hier ruim 5 jaar werkzaam als project engineer en de laatste anderhalf jaar ook als software engineer.
AL-S Technology heeft als core business het vervaardigen en verkopen van lassystemen voor de auto industrie. Onze klanten bestaan uit de bekende autofabrikanten zoals Audi, Daimler, General Motors, BMW enz.

Het las systeem van AL-S Technology is een gepatenteerd systeem. Dit systeem kan in zeer korte tijd met hoge stromen lassen maken. De lastijd bedraagt +/- 3 tot 6 ms, met stromen tot 25kA. AL-S Technology onderscheidt zich in het feit dat traditionele puntlas systemen veel langer lassen, veel warmte inbrengen en lelijke laspunten achterlaten. Bij het systeem van AL-S Technology is dit niet het geval, zo kan er gelast worden op zeer smalle flenzen. Er kunnen min of meer onzichtbare directe lassen gemaakt worden. Bij indirecte lassen (serielassen) garandeert
AL-S Technology een volledige onzichtbaarheid aan de buitenzijde.
AL-S Technology levert ook een eigen lascontroller die de lassen realtime controleert en beoordeelt. Elke las die gemaakt wordt, wordt opgeslagen, er wordt dus veel data opgeslagen die tot nu toe niet goed benut wordt.
Als deze data geanalyseerd moet worden gebeurt dit met Excel, er moeten xml files geconverteerd worden naar csv bestanden enz. Dit is erg omslachtig en de waardevolle informatie die deze gegevens bevatten wordt niet optimaal benut.

2 Projectdefinitie

In dit hoofdstuk worden doelstelling, probleemstelling, resultaat en afbakening beschreven.

2.1 Doelstelling

Deze opdracht heeft als doelstelling om een webapplicatie te ontwikkelen voor AL-S Technology. Met deze webapplicatie kan lasdata geanalyseerd worden. De applicatie dient minimaal de volgende functies te bevatten :

1. Service monteurs en medewerkers van AL-S Technology moeten in kunnen loggen zodat bekend is wie welke files uploadt.
2. Een of meerdere beheerders moeten het systeem kunnen onderhouden, bijv. klanten beheren, log files beheren enz.
3. Gebruikers moeten files kunnen uploaden.
4. Gebruikers moeten op een dataset (geüploade logfiles) een analyse kunnen maken. Hierbij moet een zeer gebruiksvriendelijke user interface voor handen zijn.
5. De user interface moet in een webbrowser draaien, de data is centraal opgeslagen in een database.

Het op te leveren eindproduct is een werkende applicatie en een scriptie die het gehele ontwikkel traject beschrijft.

2.2 Probleemstelling

De beschikbare gegevens die door de lascontroller verzameld worden kunnen niet op een efficiënte wijze geanalyseerd worden.
De lascontroller slaat voor elke gemaakte las een log record op. Hierin zijn verschillende meetgegevens over de las opgeslagen, zoals pre-force, post-force en de energie.
Omdat de las realtime gecontroleerd wordt is het van belang dat de bewakingsgrenzen van het systeem zo optimaal mogelijk ingesteld worden. De lascontroller kan op niet destructieve wijze beoordelen of de las goed is ja of nee. Het is dus van groot belang dat de parameters goed zijn. Dat wil zeggen de grenzen mogen niet te klein zijn omdat er dan onnodig foutmeldingen komen omdat er altijd bepaalde proces variaties zijn. Maar de grenzen mogen ook niet te groot zijn omdat er dan mogelijk slechte lassen goedgekeurd worden.
Op dit moment wordt er meer op gevoel gewerkt en worden zo de parameters in de lascontroller ingesteld.

Verder is er een slecht zicht op hoe verschillende systemen presteren, hierbij valt te denken, waarom zijn bij het ene systeem de procesvariaties groter als bij het andere systeem. Ligt dit aan de aangeleverde delen of is er iets anders aan de hand.

Het te ontwikkelen systeem moet helpen om grote hoeveelheden lasgegevens te analyseren (statistisch), zodat de optimale instellingen voor het bewakingsproces snel gevonden kunnen worden. Verder moeten vroegtijdig fouten opgespoord kunnen worden om het proces te stroomlijnen.

Een ander belangrijk aspect is dat gegevens van wereldwijde klanten op een centrale en consistente manier opgeslagen worden. Zodat bijvoorbeeld een monteur in China het over dezelfde gegevens heeft als iemand uit Amersfoort. Als engineers een duidelijk zicht hebben op de prestatie van een lassysteem zal ook de motivatie toenemen om systemen te optimaliseren.

2.3 Opdrachtomschrijving

Er wordt een werkende webapplicatie opgeleverd voor het analyseren / statistisch analyseren van las data. Zie Use Case Model_Weld Analysis System voor een beschrijving van het systeem en de use cases. Er zal ook een scriptie opgeleverd worden waar de keuzes onderbouwd worden, en het verloop van het ontwikkel traject zal beschreven worden.

2.4 Deelopdrachten

- Plan van Aanpak
- Use Case Model Document
- Use Case Specification Document / functioneel ontwerp
- ERD / technisch ontwerp
- Data analyse (welke data uit de logfile gaan we gebruiken)
- Ontwikkelen use cases iteratie 1
- Ontwikkelen use cases iteratie 2

2.5 Resultaat

Het te verwachten resultaat is een werkende web applicatie om las data (logfiles) te analyseren. Hierin moet het mogelijk zijn om gebruikers aan te maken. Deze gebruikers moeten kunnen inloggen. Gebruikers moeten XML files kunnen uploaden en op deze files / gegevens een data analyse kunnen doen. Hiervan moeten grafieken gemaakt kunnen worden.
Naast de werkende applicatie wordt een scriptie opgeleverd.

2.6 Organisatorische afbakening

Binnen dit afstudeer project zijn de volgende personen betrokken:

AL-S Technology	Karel Pieterman
Hogeschool Utrecht	Gerald Ovink
Student	Martijn Brands (1554106)

2.7 Afbakening van het project

Het project heeft een bepaalde scope we zullen hier nader beschrijven wat er binnen en buiten dit project valt. Een beschrijving van de use cases vindt u terug in het volgende document Use Case Model_Weld Analysis System

Binnen het project vallen de volgende zaken:

- Beheren van gebruikers (aanmaken, wijzigen, verwijderen)
- Gebruikers kunnen inloggen
- Beheren van lasdata (aanmaken, wijzigen, verwijderen)
- Analyseren van gegevens (grafieken genereren)

We gaan nu nader in op de bepaalde analyse functies die geïmplementeerd **moeten** worden:

Lijndiagram Pre Force
Wordt gebruikt om de “pre-force” waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Lijndiagram Post Force
Wordt gebruikt om de “post-force” waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Lijndiagram WeldPower
Wordt gebruikt om de “gemeten lasstroom“ waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Histogram pre-force
Wordt gebruikt om de proces variabelen m.b.t. “pre-force” weer te geven. Hoe stabiel is het proces? (Sigma).

Histogram post-force
Wordt gebruikt om de proces variabelen m.b.t. “post-force” weer te geven. Hoe stabiel is het proces? (Sigma).

Histogram WeldPower
Wordt gebruikt om de proces variabelen m.b.t. “gemeten lasstroom” weer te geven. Hoe stabiel is het proces? (Sigma).

De volgende analyse functies zijn wenselijk maar zijn geen must:

Cirkeldiagram lasfouten
Geeft per laspunt aan hoeveel en welke lasfouten zijn opgetreden.

Buiten de scope van het project :

- Deployment op een server (extern) . De software zal volledig werkend zijn in de ontwikkelomgeving. Het systeem is wel zo ontworpen dat het klaar is om op een server geïnstalleerd te worden.

3 Uitgangspunten en Randvoorwaarden

In de volgende twee paragrafen worden de uitgangspunten en randvoorwaarden besproken.

3.1 Uitgangspunten

- Als gegevensbron worden XML files gebruikt die uit de huidige CPU komen.
- Software wordt geschreven in JAVA EE, JSF 2.0.

3.2 Randvoorwaarden

- Na acceptatie en ondertekening kunnen wijzigingen in de scope of opdracht alleen met volledige overeenstemming van betrokken partijen (paragraaf 2.6) doorgevoerd worden.

4 Projectplan

De volgende paragrafen beschrijven de aanpak en planning van het project.

4.1 Aanpak

Na ondertekening van het plan van aanpak en Use Case Model Document zal er gestart worden de use cases te specificeren in het Use Case Specification Document / functioneel ontwerp.
Hierin zal elke use case in detail besproken worden, en zullen er wat scherm ontwerpen / schetsen bijgevoegd worden.

De volgende stap is om de data die beschikbaar is te analyseren. Alle data uit de XML files (log files) moet op een gestructureerde wijze in de database opgeslagen worden. Verder moeten alle gegevens van gebruikers en klanten ontleed worden in entiteiten. De data wordt uiteindelijk opgenomen in het ERD diagram (entity relationship diagram).

Er zal een keuze gemaakt worden voor een application server en database. Belangrijke eis is dat deze open source zijn.

Als het complete systeemontwerp akkoord is, zal er gestart worden met iteratie 1

4.2 Planning

Week 8	Plan van Aanpak Use Case Model Document
Week 9	Keuze database en application server Inrichten Ontwikkelomgeving
Week 10	Data Analyse Use Case specification Document scherm ontwerpen + navigation Entity Relation Diagram (ERD)
Week 14	Database live application server live Use cases geïmplementeerd Iteratie 1
Week 18	Use cases geïmplementeerd Iteratie 2
Week 20	Scriptie gereed

De scriptie zal met het project meegroeien, bij elke “milestone” zal de scriptie opgestuurd worden naar Gerald Ovink en Karel Pieterman.

5 Projectorganisatie

Hogeschool utrecht
Gerald Ovink gerald.oving@hu.nl

AL-S Technology
Karel Pieterman k.pieterman@arplas.nl

Studenten:
Martijn Brands martijn.brands@student.hu.nl 1554106

Bijlage 1 Bronvermelding

- Use Case Model_Weld Analysis System versie 1.0

Bijlage B Use Case Model

Martijn Brands

Weld Analysis System
Use Case Model

Versie 1.0

Documenthistorie

Datum	Versie	Beschrijving	Auteur
14-02-2012	1.0	Initiële versie	Martijn Brands

Distributie

Naam	1.0													
Karel Pieterman	X													
Gerald Ovink	X													

Accordering document

Namens AL-S Technology Karel Pieterman ……

Namens Hoge School Utrecht Gerald Ovink …….

Inhoudsopgave

Inleiding

Doel van dit document

Dit document geeft een samenhangend overzicht van de Use Cases en Actors voor het te bouwen systeem. Hiermee worden ook de systeemgrenzen aangegeven. Ook is er een prioritering aan de Use Cases toegekend.

Referenties

Titel	Versie	Auteur	Vindplaats
Plan van Aanpak	1.0	Martijn Brands	

Opsomming Actors

Actors zijn menselijke gebruikers(rollen) of andere systemen die met Weld Analysis System communiceren. De aard van de Actor bepaalt het gewicht. Dit kan variëren van 1 t/m 3, waarbij 3 de hoogste waarde is (meest complex). Een menselijke Actor die interacteert via een (grafische) user interface heeft gewicht 3, een interactieve of protocolgebaseerde interface gewicht 2, en een geprogrammeerde interface gewicht 1. Dit gewicht dient als basis voor een Use Case Punten Analyse.

Code	Actor	Omschrijving	Gewicht
A1	User	Service Engineer, Medewerker AL-S Technology	3
A2	Administrator	Beheerder of Senior gebruiker (admin rechten)	3

Opsomming Use Cases

Een Use Case, een ‘gebruiksgeval’, beschrijft de interactie van een Actor met het systeem. Deze interactie leidt tot een voor de Actor waardevol doel.

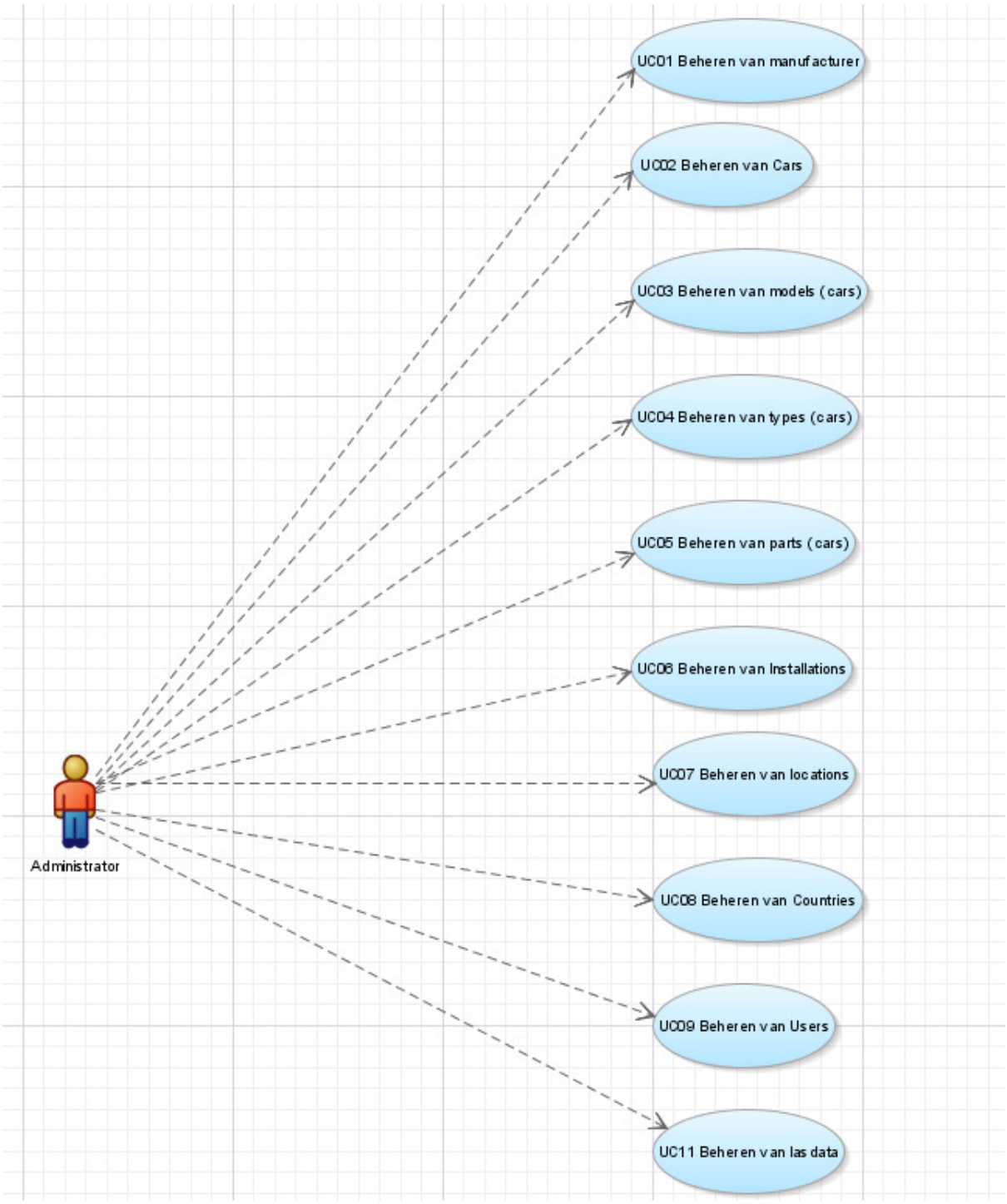
In de laatste kolom staat de prioritering van de functionaliteit voor de business. Deze dient als hulpmiddel bij de bepaling van de scope en de volgorde van realisatie. De letters die worden gebruikt zijn de medeklinkers in het woord MoSCoW. De letters staan voor:

- '**M**ust Have': deze Use Case is onmisbaar voor de bruikbaarheid van het informatiesysteem of het halen van de business case.
- '**S**hould Have': deze Use Case is sterk gewenst, maar er is een (tijdelijke) 'work-around' beschikbaar.
- '**C**ould Have': de Use Case heeft een duidelijke toegevoegde waarde, maar zonder is er nog steeds een bruikbaar systeem.
- '**W**ant to Have But Won't Have This Time Around': deze Use Case wordt in de actuele softwareontwikkellifecycle niet meegenomen, wat niet wil zeggen dat hij onbelangrijk is. Bij een volgende lifecycle kan het best een 'Must Have' zijn.

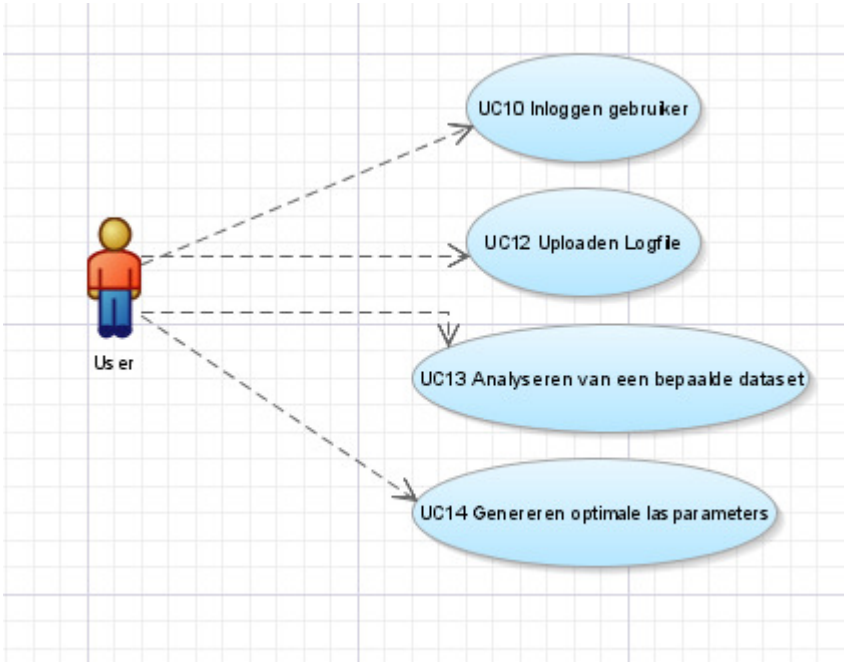
Code	Use Case Naam	Omschrijving	Iteratie	Prioritering
UC01	Beheren van manufacturer	Een administrator kan een nieuwe manufacturer toevoegen, wijzigen of verwijderen	1	M
UC02	Beheren van cars	Een administrator kan een nieuwe car toevoegen, wijzigen of verwijderen	1	M
UC03	Beheren van models	Een administrator kan een nieuw model toevoegen, wijzigen of verwijderen	1	M
UC04	Beheren van types	Een administrator kan een nieuwe type toevoegen, wijzigen of verwijderen	1	M
UC05	Beheren van parts	Een administrator kan een nieuwe parts toevoegen, wijzigen of verwijderen	1	M
UC06	Beheren van installations	Een administrator kan een nieuwe installation (lasinstallatie) toevoegen, wijzigen of verwijderen	1	M
UC07	Beheren van locations	Een administrator kan locations aanmaken, verwijderen of wijzigen.	1	M
UC08	Beheren van countries	Een administrator kan countries aanmaken, verwijderen of wijzigen.	1	M
UC09	Beheren van users	Een administrator kan users aanmaken, verwijderen of	1	M

Code	Use Case Naam	Omschrijving	Iteratie	Prioritering
		wijzigen.		
UC10	Inloggen gebruiker	Een gebruiker moet inloggen voordat die met het systeem kan werken	1	M
UC11	Beheren van Lasdata	Een administrator kan lasdata (welrecords) verwijderen of wijzigen.	2	M
UC12	Uploaden van een Logfile	Een gebruiker kan een logfile uploaden	2	M
UC13	Analyseren van een bepaalde dataset	Een gebruiker kan op een beplaade dataset bepaalde analyses uitvoeren	2	M
UC14	Genereren van optimale lasparameters	Een gebruiker kan per laspunt automatisch optimale lasparameters genereren.	2	W

Use Case diagram



Figuur 35, Use Case Diagram Administrator



Figuur 36, Use Case Diagram User

Bijlage C Use Case Specifications

Bijlage C betreft een selectie van use case Specifications uit het Use Case Specification Document. Vanwege de grootte van het document zijn de belangrijkste use cases geselecteerd.

Beheren van Model

Inleiding

In deze use case kan de beheerder een model toevoegen, wijzigen of verwijderen. Tevens is er een tabel aanwezig met de reeds ingevoerde models. Als er een record in de tabel geselecteerd wordt dan wordt het model in de detail velden weergegeven. De user heeft hier de mogelijkheid het model te wijzigen of te verwijderen. Het systeem controleert of het model /year uniek is anders wordt een foutmelding getoond. Met model wordt hier bedoeld bijv. A3 van Audi, de verdere specificatie van bijv. cabrio of sportback wordt in de use case car en type behandeld.

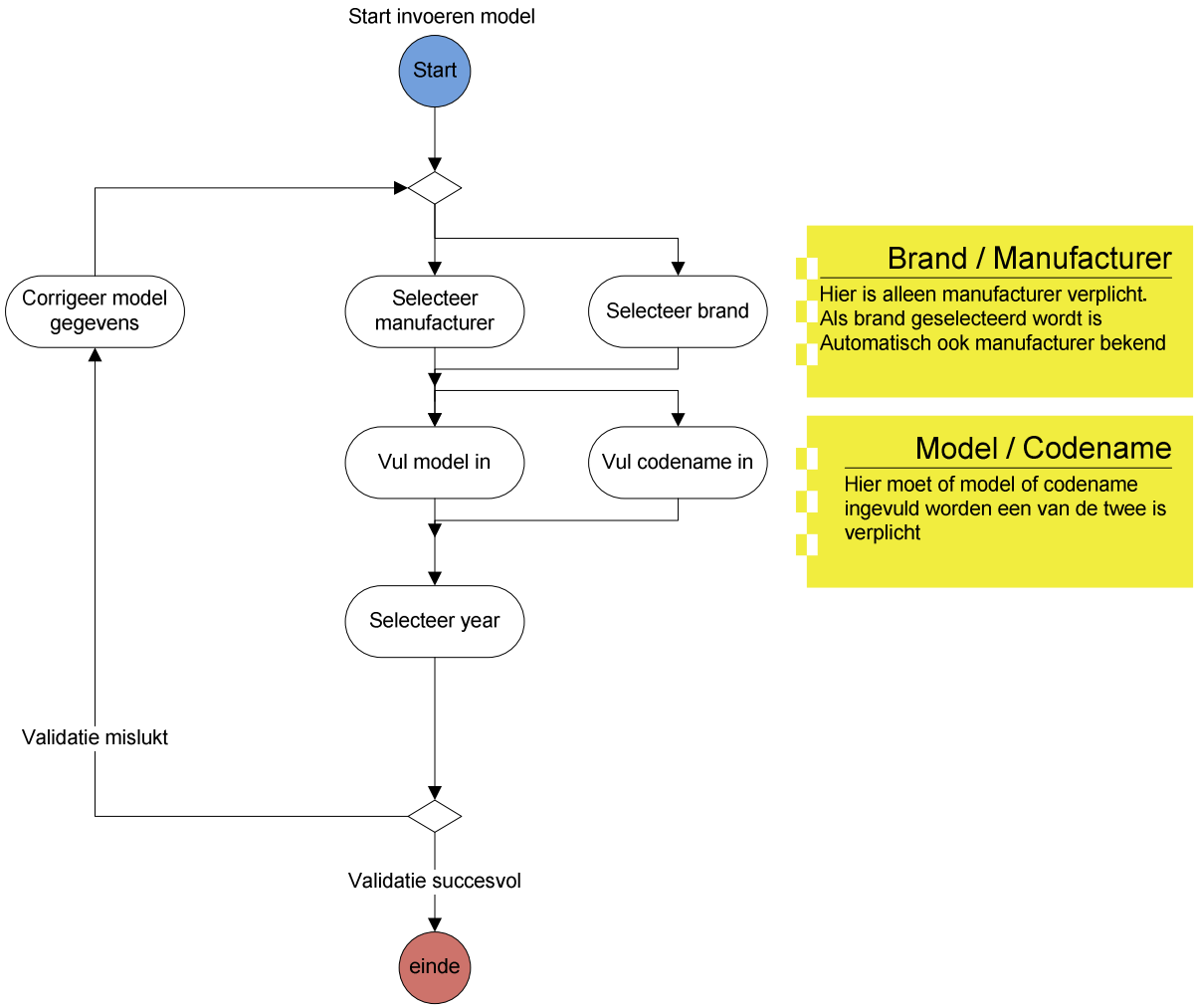
Kenmerken

Kenmerk	Omschrijving
Aanleiding (“Trigger”)	Nieuw Model invoeren of bestaand model wijzigen of verwijderen.
Actors	administrator
Wijze van uitvoering	Interactief
Samenhang met andere Use Cases	
Frequentie van uitvoering	Soms

Volgorde van gebeurtenissen

Activiteitendiagram

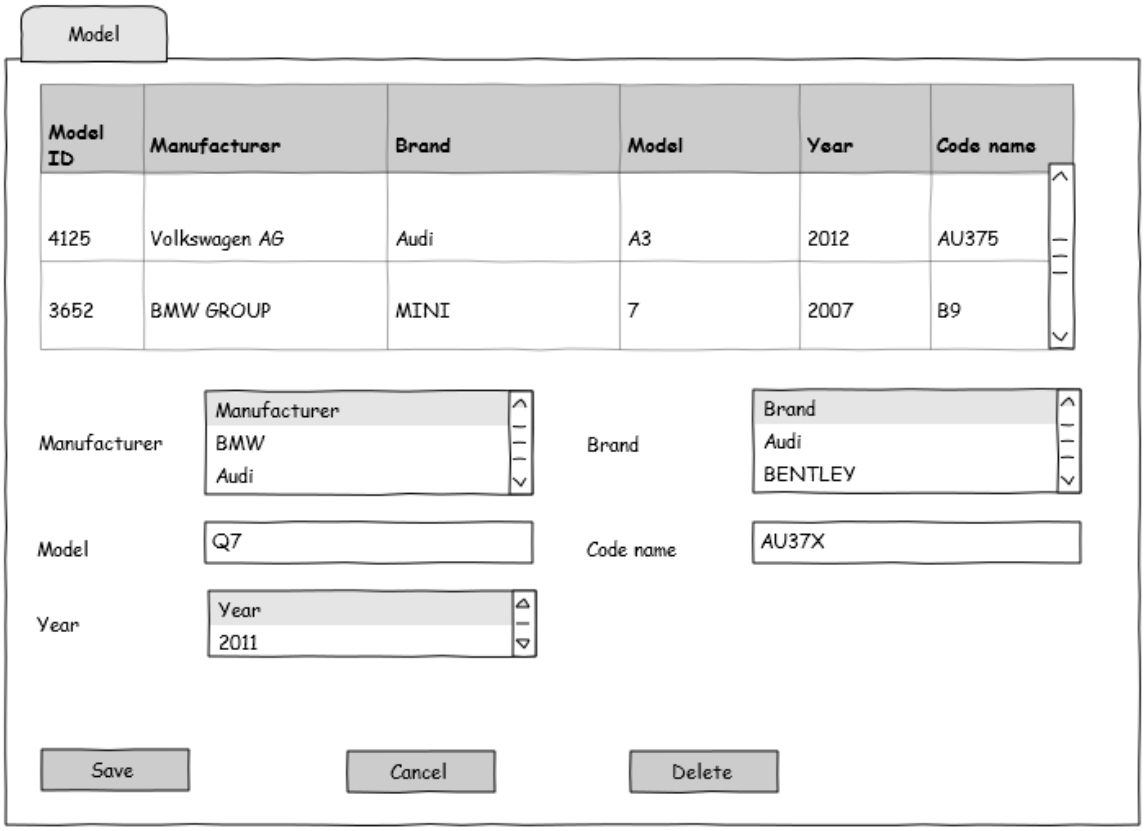
Afbeelding 7: Voorbeeld activiteitendiagram bij Use Case: Beheren van model



Basisscenario —Invoeren nieuwe manufacturer

Actor	Beheerder / Administrator
Preconditie	Gebruiker is ingelogd als administrator
Scenario beschrijving	1. De medewerker geeft te kennen, een model te willen toevoegen. 2. Het systeem toont het scherm 'Model'. 3. De medewerker selecteert een manufacturer dit kan ook door direct brand te kiezen hier is manufacturer bekend. 4. De medewerker selecteert eventueel brand. 5. De medewerker vult of codename of model in of beide. 6. De medewerker selecteert het modeljaar en bevestigt. 7. Het systeem valideert* de ingevulde gegevens. 8. Het systeem slaat de gegevens op. 9. Het systeem ververst en toont het nieuwe model in de model tabel. * 'valideert' wil zeggen dat de ingevulde gegevens gecontroleerd worden en goed bevonden.
Postconditie	geen

Afbeelding 8: Schermontwerp van scherm Beheren Model


Model
Model ID
Manufacturer
Brand
Model
Year
Code name
4125
Volkswagen AG
Audi
A3
2012
AU375
3652
BMW GROUP
MINI
7
2007
B9
Manufacturer
Manufacturer
BMW
Audi
Brand
Brand
Audi
BENTLEY
Model
Q7
Code name
AU37X
Year
Year
2011
Save
Cancel
Delete

Tabel 1: Beschrijving van velden bij scherm Beheren Model

Label	Verplicht	Status	Formaat / lengte	Validatie	Toelichting
Model ID	J	O	Numeriek(4)	n.v.t.	Wordt automatisch gegenereerd
Manufacturer	J	I	combobox	n.v.t.	Hier wordt de manufacturer geselecteerd.
Brand	N	I	combobox	n.v.t.	Hier wordt brand geselecteerd
Model	N	I	Tekst(40)	Bevat alleen de tekens A t/m Z, a t/m z, 0 t/m 9 en spatie.	Hier wordt de naam van model ingevuld. Model of codename is verplicht.
Codename	N	I	Tekst(40)	Bevat alleen de tekens A t/m Z, a t/m z, 0 t/m 9 en spatie.	Hier wordt de codename ingevuld. Model of codename is verplicht.
Year	J	I	combobox	n.v.t.	Hier wordt het jaar van het model geselecteerd. Hier wordt bedoeld jaar van start project.

Beheren van Car

Inleiding

In deze use case kan de administrator een car toevoegen, wijzigen of verwijderen. Tevens is er een tabel aanwezig met de reeds ingevoerde cars. Als er een record in de tabel geselecteerd wordt dan wordt de car in de detail velden weergegeven. De user heeft hier de mogelijkheid de car te wijzigen of te verwijderen. Van het cars overzicht is geen schermontwerp aanwezig. Afbeelding 12 betreft het invoeren van een Car. Car is hier de fysieke auto die op een bepaalde plaats geproduceerd wordt. De car bestaat tenminste uit een manufacturer, model (tenminste codename), country, location als type bekend is kan deze ook gespecificeerd worden. De combinatie van deze attributen is uniek.

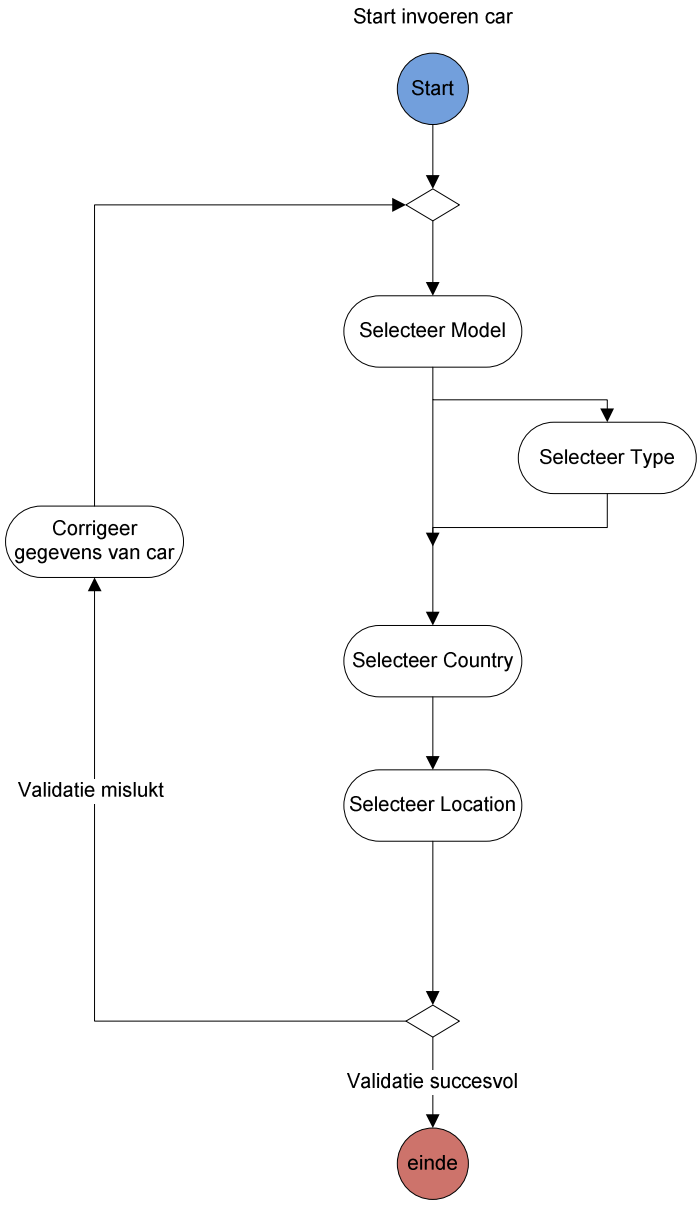
Kenmerken

Kenmerk	Omschrijving
Aanleiding (“Trigger”)	Nieuwe car invoeren of bestaande car wijzigen of verwijderen.
Actors	administrator
Wijze van uitvoering	Interactief
Samenhang met andere Use Cases	
Frequentie van uitvoering	Regelmatig

Volgorde van gebeurtenissen

Activiteitendiagram

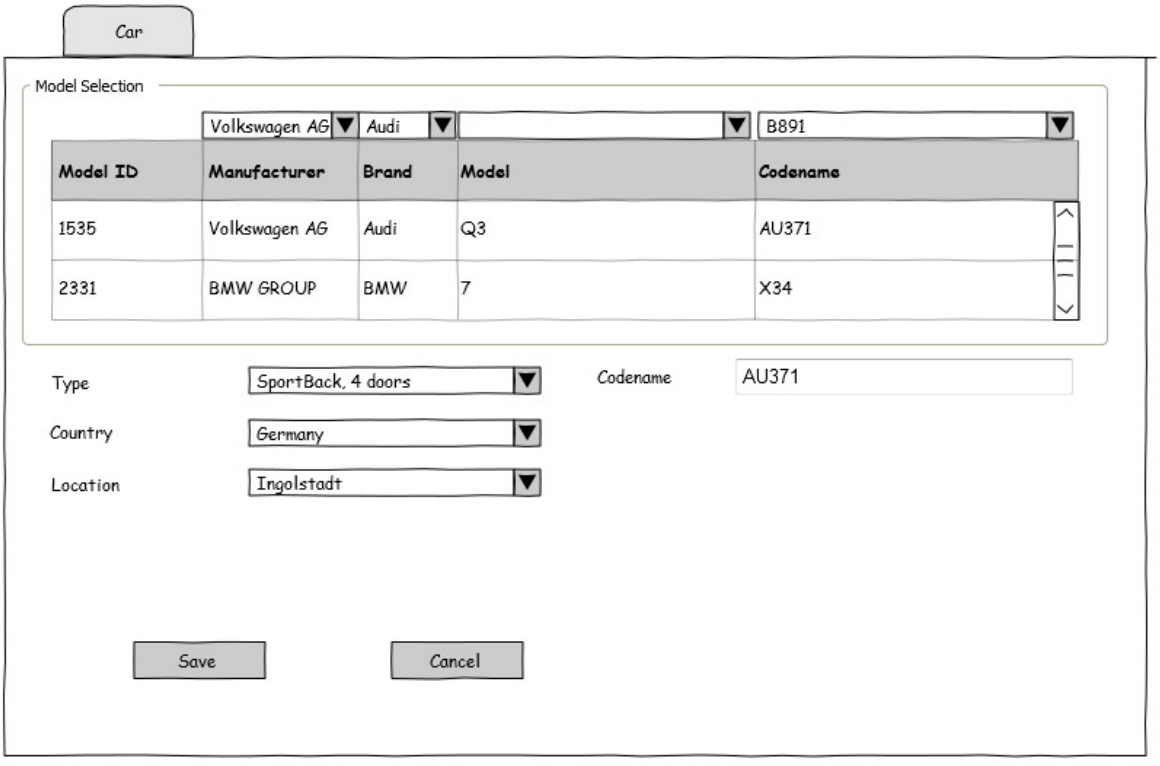
Afbeelding 11: Voorbeeld activiteitendiagram bij Use Case: Beheren van car



Basisscenario —Invoeren nieuwe manufacturer

Actor	Beheerder / Administrator
Preconditie	Gebruiker is ingelogd als administrator
Scenario beschrijving	1. De medewerker geeft te kennen, een car te willen toevoegen. 2. Het systeem toont het scherm 'Car'. 3. De medewerker selecteert het juiste model. 4. De medewerker selecteert eventueel een type. 5. De medewerker kan eventueel de codename invullen of nader specificeren. 6. De medewerker selecteert een country 7. De medewerker selecteert een location en bevestigt 1. Het systeem valideert* de ingevulde gegevens. 8. Het systeem slaat de gegevens op. 9. Het systeem ververst en toont de nieuwe car in de car tabel. * 'valideert' wil zeggen dat de ingevulde gegevens gecontroleerd worden en goed bevonden.
Postconditie	geen

Afbeelding 12: Schermontwerp van scherm Beheren Car



Tabel 2: Beschrijving van velden bij scherm Beheren Type

Label	Verplicht	Status	Formaat / lengte	Validatie	Toelichting
Model	J	O	Tabel	n.v.t.	Hier het wordt model geselecteerd door op de betreffende rij te klikken.
Codename	N	I	Edit combobox	Bevat alleen de tekens A t/m Z, a t/m z, 0 t/m 9	Hier wordt de codename geselecteerd d.m.v. een combobox en eventueel verder gespecificeerd.
Type	N	I	combobox	n.v.t.	Hier wordt het eventueel type geselecteerd d.m.v. een combobox
Country	J	I	combobox.	n.v.t.	Hier wordt de country geselecteerd d.m.v. een combobox
Location	J	I	combobox	n.v.t.	Hier wordt de location geselecteerd d.m.v. een combobox

Beheren van CarParts

Inleiding

In deze use case kan de administrator carparts aan een car toevoegen of verwijderen. Zo ontstaan er unieke carparts. Als de administrator de juiste car geselecteerd heeft, dit is de fysieke car op een bepaalde locatie. Dan kan de administrator parts toevoegen d.m.v. drag en drop. Bijv. voor de Audi A3 5 doors Hatchback Hungary Gyor. Bij deze car sleept de administrator de 4 deuren die door AL-S Technology gelast worden naar de Arplas weld parts list box. Als de administrator op save drukt worden er 4 carparts (records) opgeslagen dit zijn nu unieke carparts.

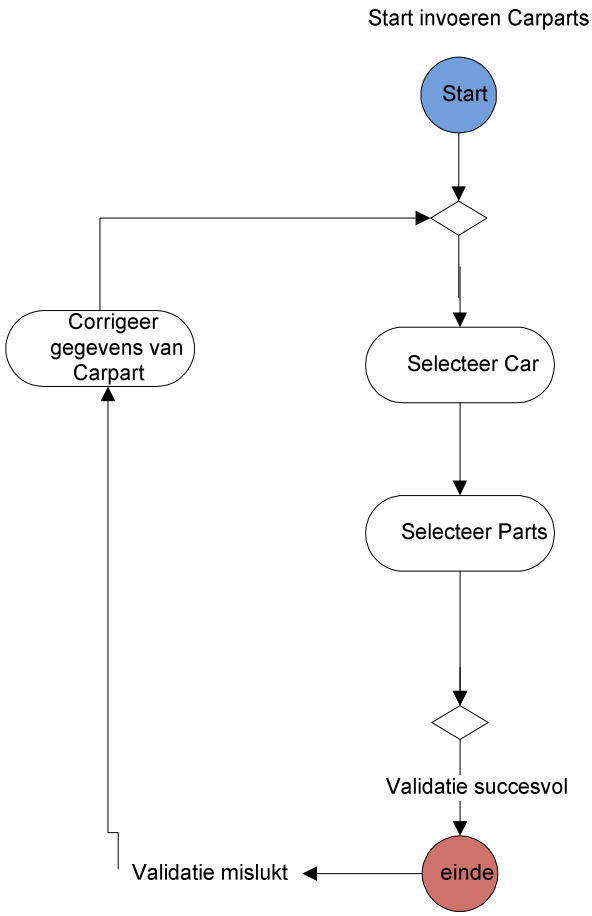
Kenmerken

Kenmerk	Omschrijving
Aanleiding (“Trigger”)	Nieuw car part invoeren of bestaand car part wijzigen of verwijderen.
Actors	administrator
Wijze van uitvoering	Interactief
Samenhang met andere Use Cases	
Frequentie van uitvoering	Regelmatig

Volgorde van gebeurtenissen

Activiteitendiagram

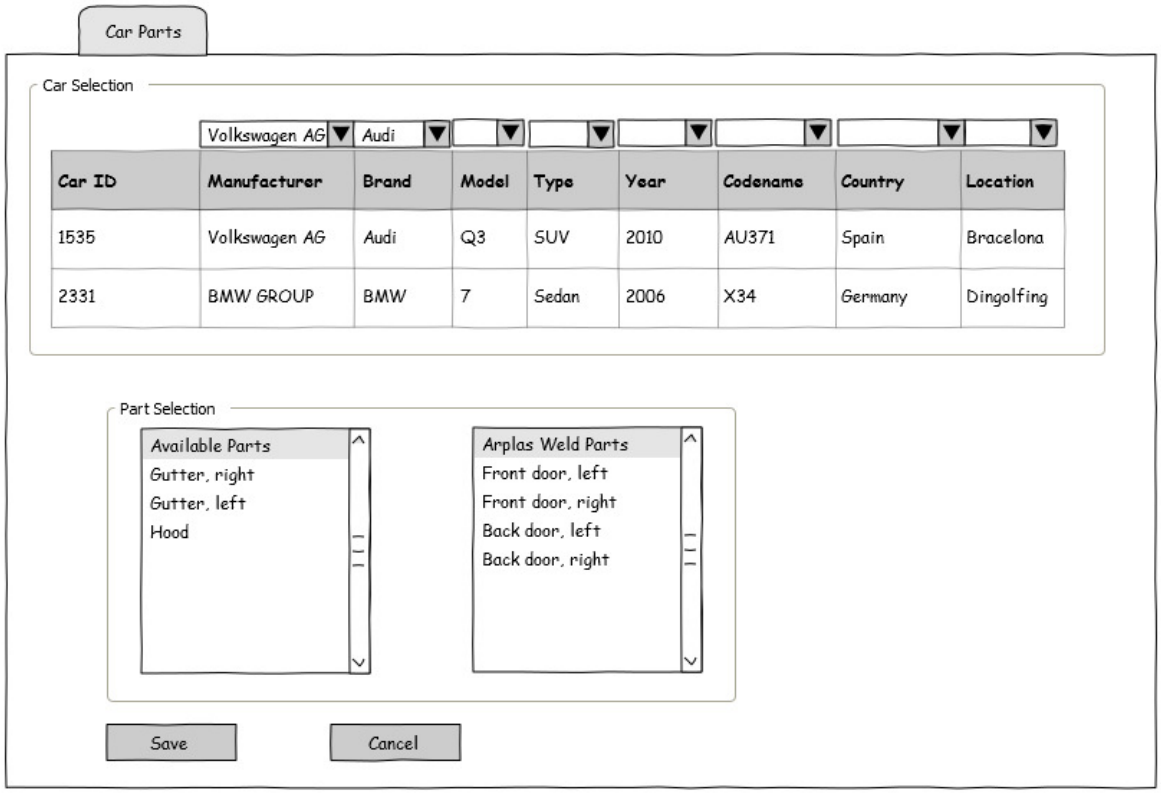
Afbeelding 15: Voorbeeld activiteitendiagram bij Use Case: Beheren van Carparts



Basisscenario —Invoeren nieuwe Carparts

Actor	Beheerder / Administrator
Preconditie	Gebruiker is ingelogd als administrator
Scenario beschrijving	1. De medewerker geeft te kennen, een Carpart te willen toevoegen. 2. Het systeem toont het scherm 'CarPart'. 3. De medewerker selecteert car 4. De medewerker selecteert de parts die bij dit model horen. 5. Het systeem valideert* de ingevulde gegevens. 6. Het systeem slaat de gegevens op. * 'valideert' wil zeggen dat de ingevulde gegevens gecontroleerd worden en goed bevonden.
Postconditie	geen

Afbeelding 16: Schermontwerp van scherm Beheren CarParts



Tabel 3: Beschrijving van velden bij scherm Beheren carparts

Label	Verplicht	Status	Formaat / lengte	Validatie	Toelichting
Part selection	J	I	n.v.t.	n.v.t.	Hier worden door middel van “drag en drop” de parts geselecteerd die bij dit model en type auto horen en door Arplas gelast worden.

Beheren van Weldinstallation

Inleiding

In deze use case kan de administrator een weldinstallation invoeren, wijzigen of verwijderen. Als er een model geselecteerd is kunnen er weldinstallations toegevoegd worden. Een weldinstalltion is hier de fysieke welcontroller. In een later stadium bij de inbedrijfname kunnen de Carparts gekoppeld worden aan de weldinstallation. Hierbij dient het serienummer als unieke identificatie voor het weldinstallation.

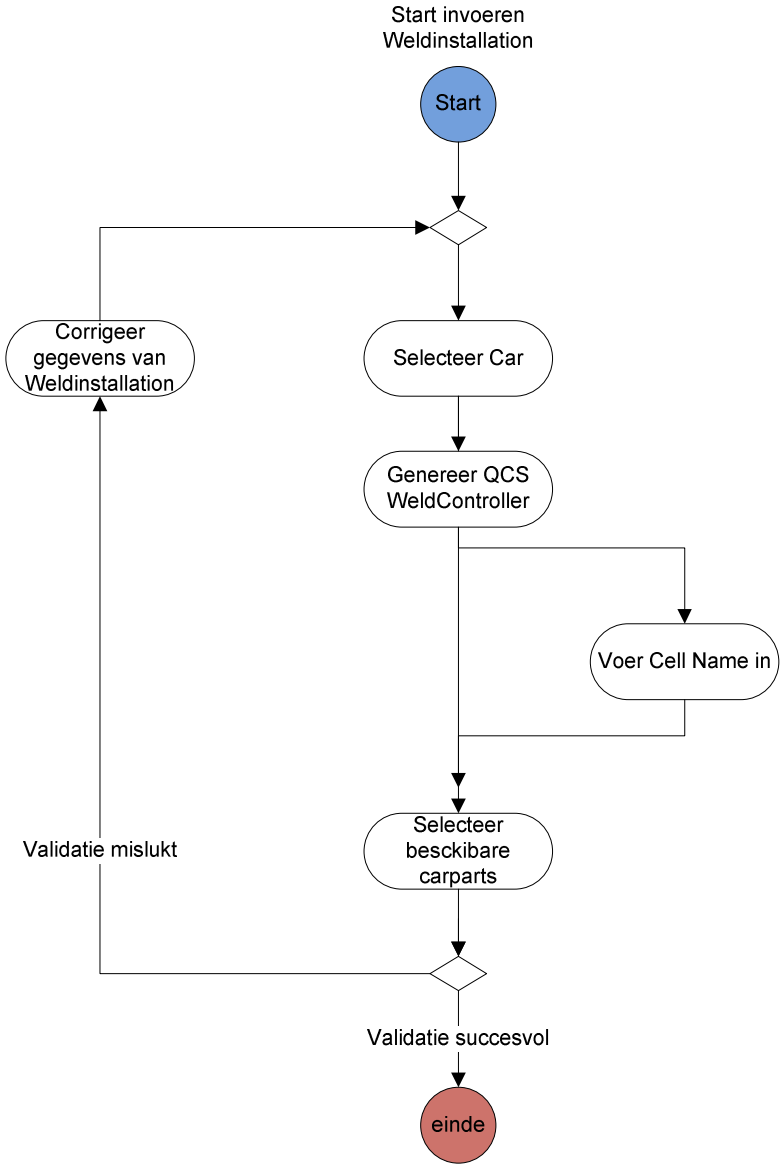
Kenmerken

Kenmerk	Omschrijving
Aanleiding (“Trigger”)	Nieuw weld installation invoeren of bestaande weld installation wijzigen of verwijderen. Tevens kunnen hier de carparts aan het weldinstallation gekoppeld worden.
Actors	administrator
Wijze van uitvoering	Interactief
Samenhang met andere Use Cases	
Frequentie van uitvoering	Regelmatig

Volgorde van gebeurtenissen

Activiteitendiagram

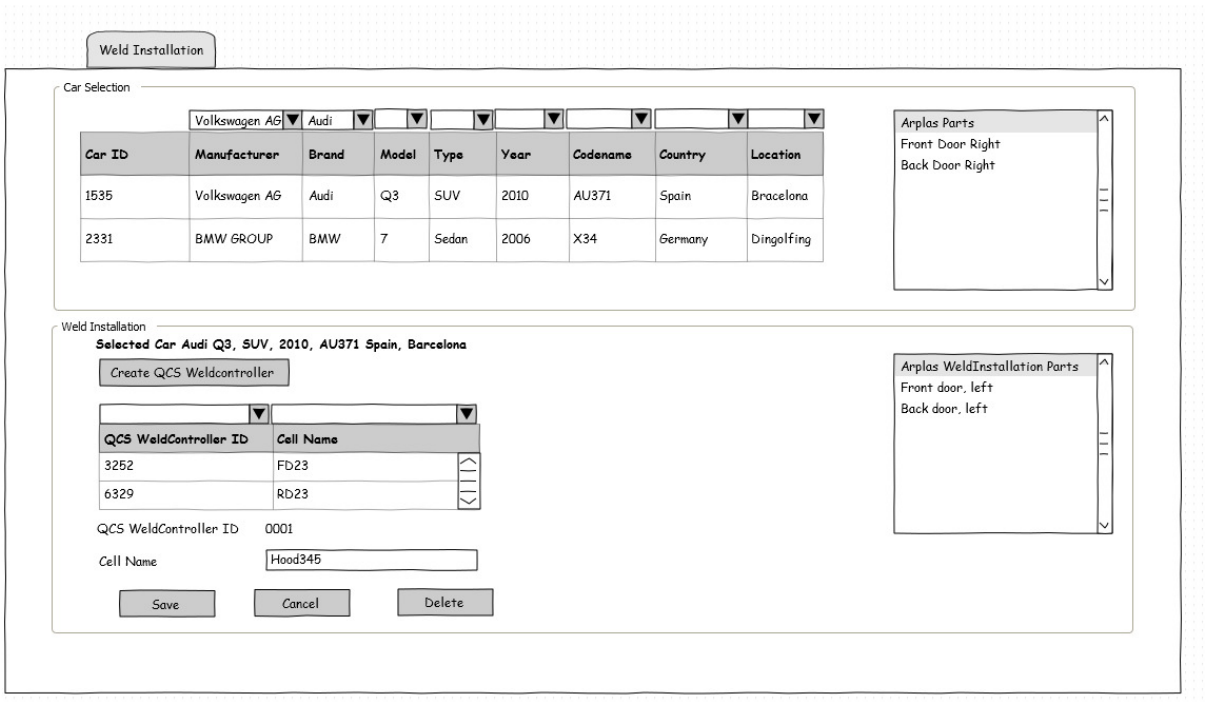
Afbeelding 17: Voorbeeld activiteitendiagram bij Use Case: Beheren van Weldinstallation



Basisscenario —Invoeren nieuwe WeldInstallation

Actor	Beheerder / Administrator
Preconditie	Gebruiker is ingelogd als administrator
Scenario beschrijving	1. De medewerker geeft te kennen, een WeldInstallation te willen toevoegen. 2. Het systeem toont het scherm 'WeldInstallation'. 3. De medewerker selecteert car 4. De medewerker creëert een nieuwe QCS WeldController 5. De medewerker voert eventueel cell name in 6. De medewerker selecteert eventueel Parts 7. Het systeem valideert* de ingevulde gegevens. 8. Het systeem slaat de gegevens op. * 'valideert' wil zeggen dat de ingevulde gegevens gecontroleerd worden en goed bevonden.
Postconditie	geen

Afbeelding 18: Schermontwerp van scherm Beheren WeldInstallation



Tabel 4: Beschrijving van velden bij scherm Beheren weldinstallation

Label	Verplicht	Status	Formaat / lengte	Validatie	Toelichting
Car	J	O	Table	n.v.t.	Hier wordt de juiste car geselecteerd met juiste type en aantal deuren
QCS Weld controller ID	J	O		n.v.t.	Wordt automatisch gegenereerd
Cell Name	N	I	Tekst(40)	Alle ASCII Tekens	Hier wordt eventueel de naam van de installatie ingevuld.
Arplas Weld Parts	N	I	Drag en drop	n.v.t.	Hier worden de parts geselecteerd d.m.v. drag en drop die bij de installatie horen.

Uploaden Logfile

Inleiding

In deze use case kan de user een logfile uploaden. De user selecteert de juiste Car, WeldParts, WeldController en start de upload. De XML file wordt op de server ontvangen en de logrecords worden uit de XML file gehaald. Vervolgens worden de logrecords in de database opgeslagen.

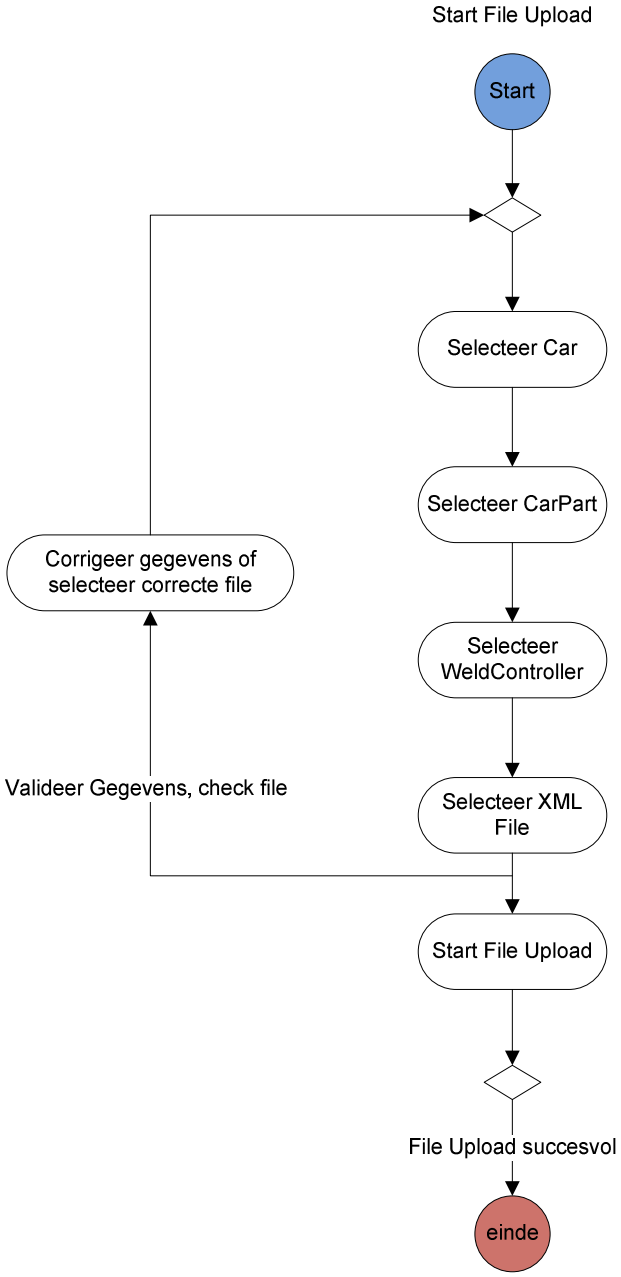
Kenmerken

Kenmerk	Omschrijving
Aanleiding (“Trigger”)	Uploaden van een logfile
Actors	User en Beheerder
Wijze van uitvoering	Interactief
Samenhang met andere Use Cases	
Frequentie van uitvoering	Regelmatig

Volgorde van gebeurtenissen

Activiteitendiagram

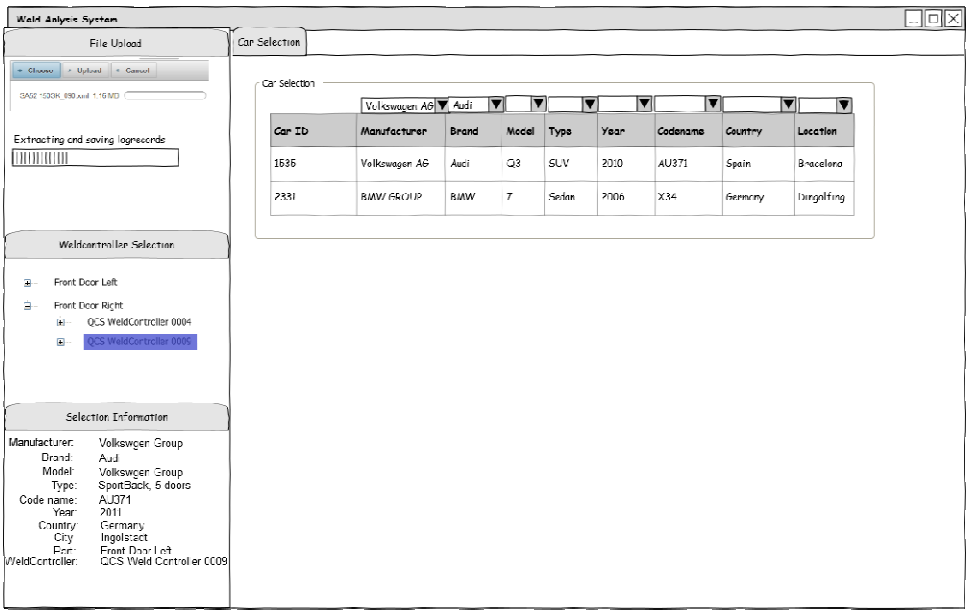
Afbeelding 22: Voorbeeld activiteitendiagram bij Use Case: Uploaden Logfile



Basisscenario —Invoeren nieuwe user

Actor	Beheerder / Administrator
Preconditie	Gebruiker is ingelogd als administrator of gebruiker.
Scenario beschrijving	1. De medewerker geeft te kennen, een file te wille uploaden 2. De medewerker selecteert de juiste Car 3. De medewerker selecteert de juiste CarPart 4. De medewerker selecteert de juiste QcsWeldController 5. De medewerker selecteert de juiste XML file 6. De medewerker start de file upload 7. Het Systeem valideert of de file een XML file is. 8. Het systeem slaat de file op. 9. Het systeem ontleed de XML file en slaat de records op in de database. 10. Het systeem geeft een melding of de upload geslaagd / niet geslaagd is. * 'valideert' wil zeggen dat de ingevulde gegevens gecontroleerd worden en goed bevonden.
Postconditie	geen

Afbeelding 22: Schermontwerp van scherm File Upload



Analyseren van een bepaalde dataset.

Inleiding

In deze use case kan een gebruiker of administrator een analyse doen op een bepaalde dataset. De dataset bestaat uit logrecords van een geselecteerde QcsWeldcontroller. Van de gegevens wordt een grafiek gemaakt. De gebruiker kan kiezen uit verschillende analyses:

Lijndiagram Pre Force
Wordt gebruikt om de “pre-force” waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Lijndiagram Post Force
Wordt gebruikt om de “post-force” waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Lijndiagram WeldPower
Wordt gebruikt om de “gemeten lasstroom“ waarden in de tijd per laspunt met elkaar te vergelijken, het moet mogelijk zijn om laspunten aan of uit te vinken. Tendenslijn weergeven.

Histogram pre-force
Wordt gebruikt om de proces variabelen m.b.t. “pre-force” weer te geven. Hoe stabiel is het proces? (Sigma).

Histogram post-force
Wordt gebruikt om de proces variabelen m.b.t. “post-force” weer te geven. Hoe stabiel is het proces? (Sigma).

Histogram WeldPower
Wordt gebruikt om de proces variabelen m.b.t. “gemeten lasstroom” weer te geven. Hoe stabiel is het proces? (Sigma).

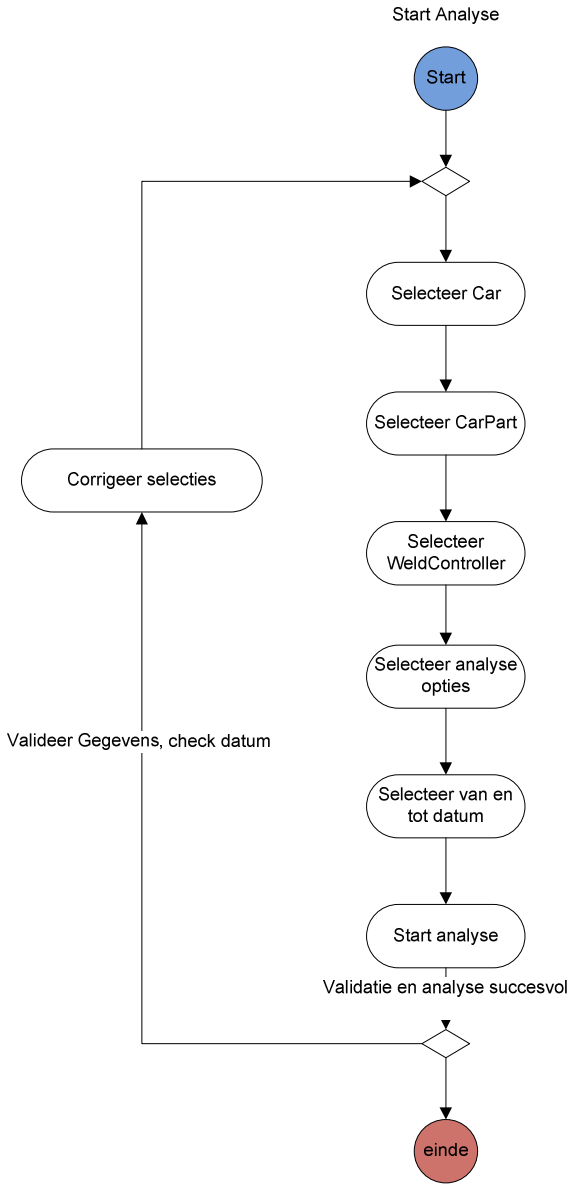
Kenmerken

Kenmerk	Omschrijving
Aanleiding (“Trigger”)	Het maken van een analyse op een bepaalde dataset
Actors	Gebruiker / Beheerder
Wijze van uitvoering	Interactief
Samenhang met andere Use Cases	
Frequentie van uitvoering	Regelmatig

Volgorde van gebeurtenissen

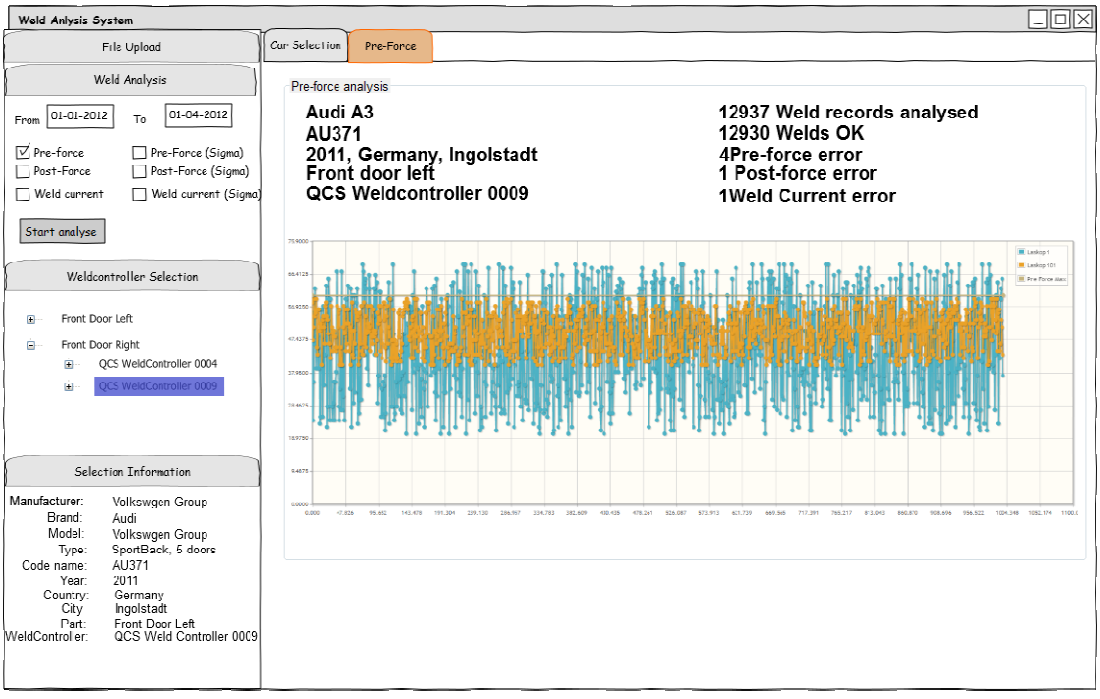
Activiteitendiagram

Afbeelding 21: Voorbeeld activiteitendiagram bij Use Case: Analyseren van een bepaalde dataset



Basisscenario —Invoeren nieuwe user

Actor	Beheerder / Administrator
Preconditie	Gebruiker is ingelogd als administrator
Scenario beschrijving	1. De medewerker geeft te kennen, een analyse te willen uitvoeren. 2. Het systeem toont het scherm Weld Analyse 3. De medewerker selecteert een Car 4. De medewerker selecteert Carpart 5. De medewerker selecteert een Qcs weldcontroller 6. De medewerker selecteert de analyse opties 7. De medewerker geeft een van en tot datum in. 8. De medewerker start de analyse 9. Het systeem valideert* de ingevulde/geselecteerde gegevens. 10. Het systeem ververst en toont de analyse opties in tabbladen. * 'valideert' wil zeggen dat de ingevulde gegevens gecontroleerd worden en goed bevonden.
Postconditie	geen

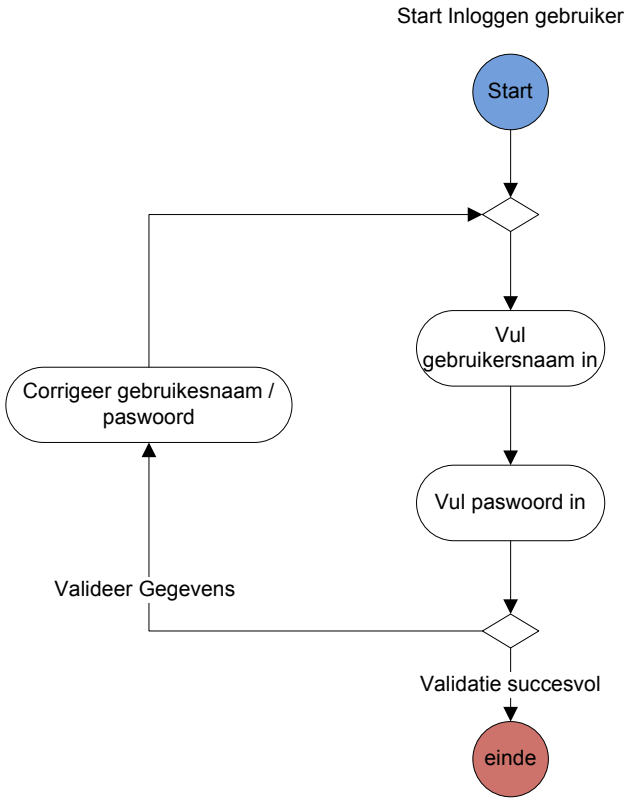


Figuur 37

Volgorde van gebeurtenissen

Activiteitendiagram

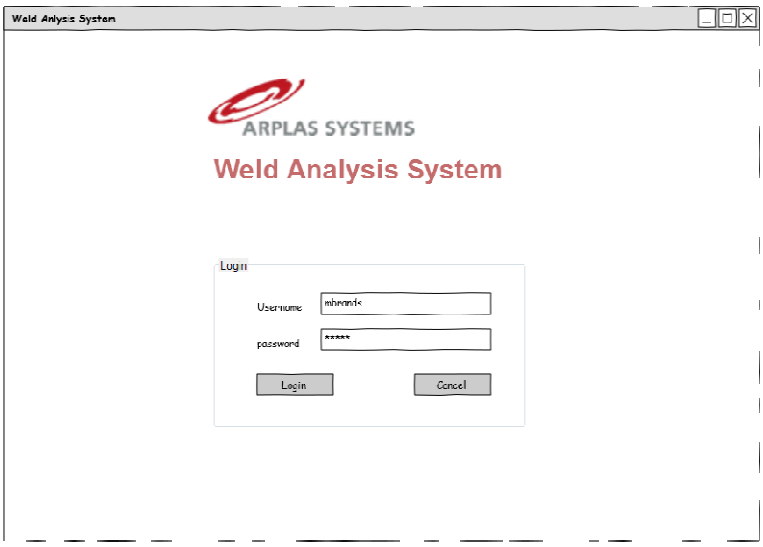
Afbeelding 21: Voorbeeld activiteitendiagram bij Use Case: Inloggen gebruiker.



Basisscenario —Invoeren nieuwe user

Actor	Beheerder / Administrator
Preconditie	Gebruiker is ingelogd als administrator
Scenario beschrijving	1. De medewerker geeft te kennen, in te willen loggen 2. Het systeem toont het scherm 'inloggen'. 3. De medewerker vult zijn gebruikersnaam in. 4. De medewerker vult zijn paswoord in. 5. Het systeem valideert* de ingevulde gegevens. 6. Het systeem ververst en toont het startscherm. * 'valideert' wil zeggen dat de ingevulde gegevens gecontroleerd worden en goed bevonden.
Postconditie	geen

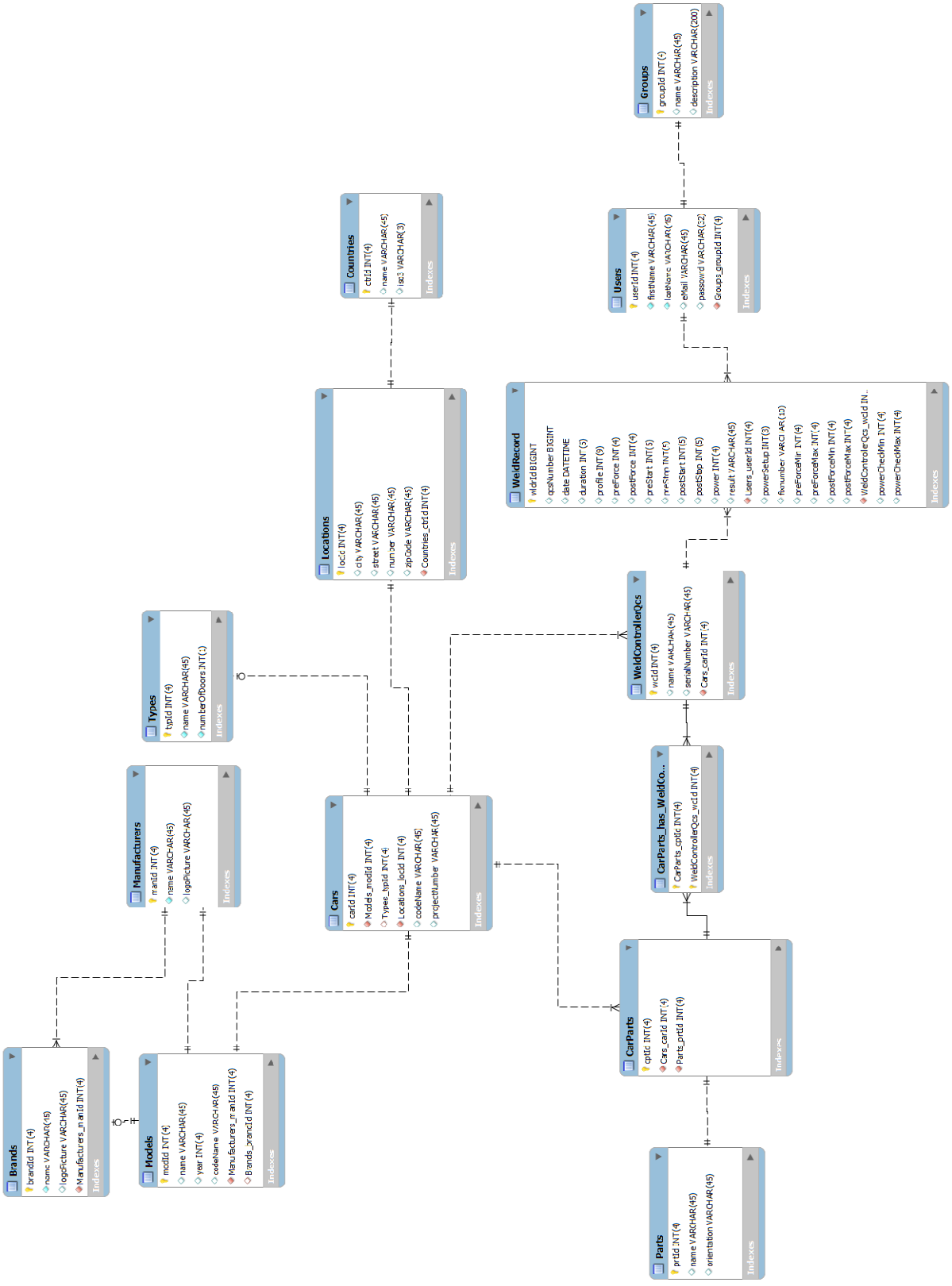
Afbeelding 22: Schermontwerp van scherm Inloggen



Tabel 5: Beschrijving van velden bij scherm Beheren groups

Label	Verplicht	Status	Formaat / lengte	Validatie	Toelichting
Username	J	O	Numeriek(4)		Wordt automatisch gegenereerd
Password	J				

Bijlage D Entity Relation Diagram



Bijlage E Data Analyse Logrecord

De gemarkeerde velden zullen opgeslagen worden in de database.

<Log>

<lg_qcsnumber>806</lg_qcsnumber>

<lg_index>0</lg_index>

<lg_fixnumber>0</lg_fixnumber>

<lg_date>2011-06-07T13:34:43+02:00</lg_date>

<lg_duration>1439</lg_duration>

<lg_result>0</lg_result>

<lg_profile>1</lg_profile>

<lg_error>false</lg_error>

<lg_nouse>false</lg_nouse>

<lg_nopower>false</lg_nopower>

<lg_reported>true</lg_reported>

<lg_preforce>685</lg_preforce>

<lg_prestart>142</lg_prestart>

<lg_prestop>774</lg_prestop>

<lg_dropforce>78</lg_dropforce>

<lg_dropstart>1023</lg_dropstart> is hetzelfde als lg_powerstart.

<lg_dropstop>1390</lg_dropstop>

<lg_powerstart>1024</lg_powerstart>

<lg_powerused>268</lg_powerused>

<lg_pre_force_min>400</lg_pre_force_min>

<lg_pre_force_max>850</lg_pre_force_max>

<lg_drop_force_min>5</lg_drop_force_min>

<lg_drop_force_max>160</lg_drop_force_max>

<lg_power_setup>60</lg_power_setup>

<lg_power_check_min>200</lg_power_check_min>

<lg_power_check_max>900</lg_power_check_max>

<_lg_result_text>OK</_lg_result_text>

<lg_postforce>607</lg_postforce>

</Log>