spliethoff
AMSTERDAM

Eindverslag

Analyse technische scheepsprestaties
(MaDo gegevens)

Amsterdam, 9 juni 2004

Student:	E. Koeleman
Nummer:	20007726
Opleiding:	Haagse Hogeschool Sector Informatica IVIT-4
Examinatoren:	Dhr. B. Pieters Dhr. A. van der Molen
Bedrijf:	Spliethoff BV
Begeleiders:	Dhr. Ir. P. Van de Venne Dhr. B. Tesselaar

Referaat

Dit eindverslag gaat over de ontwikkeling van een applicatie waarmee scheepsprestaties geanalyseerd en bijgehouden kunnen worden. Er wordt ingegaan op het ontwikkel traject en de daarbij gemaakte keuzes.

- 3-Tier
- Borland Delphi 7
- IAD
- Microsoft IIS 5.1
- Microsoft SQL Server 2000
- Nevrona Rave Reports
- Scheepsprestaties
- SOAP
- Spliethoff
- UML
- Webservices
- WSDL

Voorwoord

Dit verslag is geschreven in het kader van het afstudeertraject aan de Haagse Hogeschool, sector Informatica, IVIT-4. Het is bedoeld als verslaglegging van de werkzaamheden en gevonden oplossingen tijdens het afstudeertraject.

Graag wil ik Dhr. Van de Venne bedanken voor het aanbieden van de afstudeerplaats bij Spliethoff. Verder wil ik ook Dhr. Tesselaar bedanken voor de begeleiding op gebied van programmeren en Dhr. Kuiper voor het uitleggen van technische scheepsaspecten.

Inhoudsopgave

1 Inleiding

Spliethoff en Biglift Shipping exploiteren gezamenlijk een vloot van ongeveer 60 uiterst moderne zeeschepen voor het vervoer van hoogwaardige lading over de gehele wereld. Het totale management van deze vloot vindt plaats vanuit Amsterdam. Binnen Spliethoff is de technische dienst verantwoordelijk voor het monitoren en in optimale conditie houden van de schepen.
De opdracht die uitgevoerd is, bestaat uit het ontwikkelen van een analyse, “kijk” en “print” functionaliteit welke betrekking heeft op de technische prestaties van de Spliethoff schepen.

Dit verslag is opgedeeld in tien hoofdstukken, welke weer een onderverdeling hebben in paragrafen.

Een beschrijving van het afstudeerbedrijf, Spliethoff, wordt gegeven in hoofdstuk twee. Tevens bevindt zich hier een beschrijving van de exacte plaats van de afstudeerder binnen het bedrijf.

Het doel van hoofdstuk drie is het aangeven hoe de opdracht tot stand is gekomen en wat de taak van de afstudeerder hierin is geweest.

In hoofdstuk vier wordt ingegaan op hoe de situatie was op het gebied van relevante aspecten tijdens de start van het traject.

Hoofdstuk vijf geeft een beschrijving van het proces zoals deze gevolgd is tijdens de definitiefase van het traject.

De beschrijving van het proces tijdens de pilotontwikkeling is opgenomen in de hoofdstukken zes, zeven en acht.

In hoofdstuk negen wordt aangegeven hoe de invoering van het uiteindelijke resultaat heeft plaatsgevonden.

Als afsluiting wordt in hoofdstuk tien het proces en de opgeleverde producten geëvalueerd.

2 Spliethoff's Bevrachtingskantoor

In dit hoofdstuk wordt het bedrijfsprofiel van Spliethoff behandeld. Er wordt ingegaan op de geschiedenis van de organisatie en de afdeling waar het afstudeertraject heeft plaatsgevonden.

2.1 Geschiedenis van Spliethoff

Spliethoff's Bevrachtingskantoor B.V. bestaat vanaf 1921 en vond zijn oorsprong in het vervoer t.b.v. de houthandel. Spliethoff is begonnen als een bevrachtingskantoor die bemiddelaar was tussen de lading- en de scheepseigenaren. Dit was de hoofdactiviteit tot 1946. In dat jaar heeft men tevens het eerste schip gekocht, de Heerengracht. Daarna heeft Spliethoff verschillende schepen laten bouwen. Het eerste schip dat speciaal voor Spliethoff werd gebouwd, was de Keizersgracht, gelijknamig aan de straatnaam waar hun eerste vestiging gehuisvest was.

Tegenwoordig hebben alle Spliethoff schepen nog steeds een naam die eindigt op "gracht". Er zijn wel verschillende series schepen (P, L, A, E en S-serie), maar uiterlijk lijken de schepen veel op elkaar. Echter verschillen zijn wel aanwezig. Naast een verschil in afmeting wordt getracht om alle ervaringen opgedaan met de voorgaande schepen zoveel mogelijk te verwerken in een volgende serie te bouwen schepen. Hierdoor ontstaan telkens weer schepen die niet alleen doelmatig zijn, maar tevens weer voldoen aan de eisen van deze tijd.

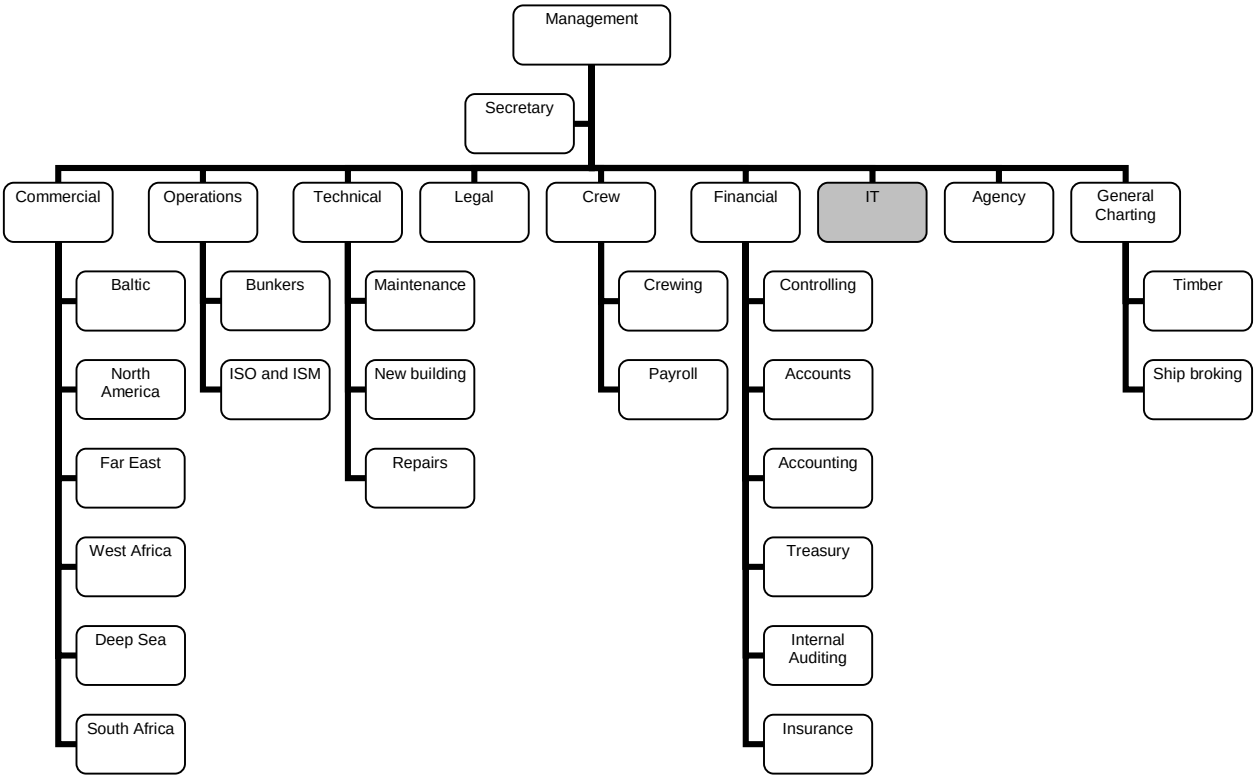
Enkele jaren geleden heeft Spliethoff een meerderheidsbelang verkregen in BigLift Shipping B.V. (voorheen Mammoet Shipping), deze vloot bestaat uit 13 "heavy lift" schepen met een capaciteit tot 1100 ton.

De totale Spliethoff vloot telt nu 55 zeeschepen variërend in grootte tussen de 8.000 en 20.000 ton. De schepen vervoeren alle soorten lading, van papier tot luxe jachten en van cacao tot containers.

Tijdens de zomer van 2002 heeft Spliethoff een meerderheidsbelang gekregen in Transfennica, Finland's logistieke leider op het gebied van hout services. Wijnne Barends uit Delftzijl is sinds september 2003 de laatste aanwinst voor de Spliethoff Groep.

Spliethoff heeft ongeveer 1150 medewerkers aan boord van de schepen en 150 medewerkers op het kantoor in Amsterdam.

2.2 De organisatie

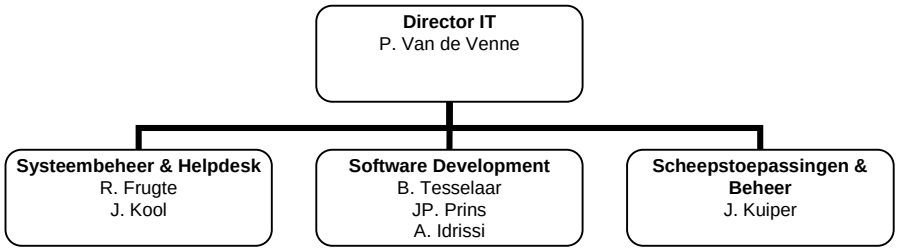


Figuur 1: Organigram Spliethoff

In het organigram is te zien dat Spliethoff een platte organisatie is. Bij de dagelijkse bedrijfsvoering komt dit ook duidelijk naar voren, zo wordt er gewerkt in grote ruimtes. Verder heerst er een zeer korte communicatielijn, juist doordat iedereen zo bij elkaar zit.

2.3 Plaats van de afstudeerder

Het afstudeertraject heeft plaatsgevonden op de IT afdeling van Spliethoff. Hoewel in het organigram van de organisatie geen directe afdelingen te onderscheiden zijn, zijn deze in de praktijk wel aanwezig.



Figuur 2: De IT afdeling

Het systeembeheer en de helpdesk zorgen voor het onderhouden en oplossen van problemen binnen het Spliethoff netwerk. De software development is verantwoordelijk voor het maken van de inhouse en maatwerk software zoals deze gebruikt wordt binnen het bedrijf. Omdat schepen een aparte benadering vragen (weinig in de buurt, hoge eisen aan installatie gemak, remote support) is dit een speciaal aandachtsveld binnen de IT, hiervoor wordt gezorgd bij Scheepstoepassingen & Beheer.
De directe plaats tijdens het afstudeertraject was bij de software development.

3 De afstudeeropdracht

De kernpunten van de afstudeeropdracht worden in dit hoofdstuk beschreven. Aan bod komen onder andere de probleemstelling, de doelstelling en de te gebruiken methoden en technieken. Voor een totaal overzicht van de opdracht wordt verwezen naar het plan van aanpak welke meegeleverd is als externe bijlage.

3.1 De probleemstelling

Deze opdracht vindt zijn oorsprong naar aanleiding van een verouderd systeem waarmee de scheepsprestaties bijgehouden kunnen worden. Deze prestatiegegevens, ook wel MaDo gegevens (zie paragraaf 4.1) genoemd, worden gedeeltelijk automatisch ingelezen. Met behulp van een Microsoft Access applicatie worden deze gegevens bekeken en eventueel aangepast. Om afwijkende waarden te herkennen, moet door een gebruiker zelf goed gekeken worden naar de gegevens.

3.2 Het doel van de opdracht

Het doel van de afstudeeropdracht is het realiseren van een automatisch afwijkingssignaleringssysteem ter bewaking van de technische scheepsprestaties met daarin grafiek en printfunctionaliteiten, gebaseerd op de aanwezige client-server architectuur. De applicatie moet geprogrammeerd worden met behulp van Borland Delphi.

Op de geleverde opdrachtomschrijving van Spliethoff zijn enkele punten opgenomen welke verwezenlijkt moesten worden. Op basis hiervan is voorafgaand aan het afstudeertraject met de opdrachtgever gesproken om deze eisen te concretiseren.

Wanneer aan de volgende punten voldaan wordt, zal de doelstelling als behaald beschouwd worden:

- De oude en nieuwe (zogenaamde maandag/donderdag) berichten moeten ingelezen dan wel geconverteerd en geanalyseerd kunnen worden;
- De gebruiker moet de geanalyseerde gegevens kunnen bekijken en wijzigen;
- De gegevens moeten geprint en geëxporteerd kunnen worden;
- Indien mogelijk moet het systeem automatisch kunnen signaleren wanneer er een negatieve trend optreedt.

Zoals in deze lijst beschreven moeten de eerste drie punten volbracht worden. De volbrenging van het vierde punt wordt als extra beschouwd.

3.3 Te gebruiken methoden en technieken

Tijdens het eerste gesprek met de opdrachtgever is ter sprake gekomen hoe de opdracht verwezenlijkt zou worden op het gebied van een te gebruiken methode. Binnen Spliethoff zelf wordt geen ontwikkelmethode gebruikt, waardoor hier een vrije keuze in was. Wel heeft de opdrachtgever aangegeven dat hij zeer betrokken wilde blijven bij de ontwikkeling. Hierdoor kwam meteen de IAD methode naar boven, deze is zeer geschikt voor een ontwikkelproces waar interactie met gebruikers en opdrachtgevers belangrijk is. Verder had de opdrachtgever aangegeven dat tijdens het ontwikkeltraject nieuwe functionaliteiten naar boven zouden kunnen komen. Dit was wederom een aanwijzing in de richting van IAD, want tijdens de iteraties zouden deze systeemeisen wel naar boven kunnen komen. Een andere mogelijke methode zou XP kunnen zijn, deze is ook geschikt voor een iteratieve aanpak in een omgeving waar systeemeisen kunnen wijzigen.
Een belangrijk voordeel voor het gebruik van de IAD methodiek is dat deze al tijdens de opleiding bij meerdere projecten gebruikt is, waardoor geen tijd vrij gemaakt hoeft te worden om de methode te leren kennen, dit in tegenstelling tot bij XP.

De IAD methode bestaat uit drie fasen, te weten de definitiestudie, de pilotontwikkeling en de pilot invoering. Tijdens de definitiestudie worden de doelen en beperkingen van het te ontwikkelen systeem geanalyseerd. De pilotontwikkeling betreft het specificeren van de technische specificaties en de daadwerkelijke bouw van het systeem. Ten slotte de invoering, waarin het systeem operationeel gemaakt wordt.

Een techniek welke gebruikt wordt tijdens de definitiestudie om het systeemconcept in kaart te brengen is het opstellen van use-cases. Hiermee wordt ook meteen de betrokkenheid van de gebruiker en/of opdrachtgever vergroot aangezien een use-case in natuurlijke taal geschreven is. Een andere techniek voor het bepalen van de uiteindelijke database structuur, is het opstellen van een domein klassendiagram. Beide technieken vallen onder UML [Warmer99].
Tijdens de pilotontwikkeling zullen XUAN (eXtended User Action Notation) modellen gebruikt worden. Hiermee kan bij de functionele structuur duidelijk de schermen en hun interacties in kaart gebracht worden. Andere technieken die bij de ontwikkeling van de pilots gebruikt zullen worden is het toestandsdiagram om de toestand van een gegevens bericht in bepaalde situaties in kaart te brengen en de flowchart voor het analyseren van bepaalde te automatiseren processen.
Om tussentijdse resultaten van een pilot te testen zal gebruikt worden gemaakt van prototyping. Dit is een goede manier om de belanghebbenden te betrekken bij de ontwikkeling. Zo zal op afgesproken tijden een pilot gedemonstreerd worden om zo feedback van de belanghebbenden uit te lokken.

3.4 Het plan van aanpak

In de eerste week bij Spliethoff is het plan van aanpak samengesteld. Dit is gebeurd op basis van de opdrachtomschrijving zoals deze voor de opleiding opgesteld is. Het plan van aanpak is door de opdrachtgever gecontroleerd en op enkele punten aangevuld. Zo werd aangedragen dat Dhr. Alblas van BigLift een aanvulling zou kunnen zijn om informatie in te winnen en ideeën op te doen. De reden hiervoor was omdat tijdens een voorstel ronde bij Spliethoff duidelijk werd dat deze ook te maken had met het bijhouden van gegevens van schepen.
De planning is ook besproken, hierin stond voor het datamodel alleen in het begin tijd gereserveerd. De opdrachtgever gaf aan dat het waarschijnlijk wel voor zou komen dat het datamodel tijdens de ontwikkeling ook nog wel zou wijzigen. Aangegeven is dat op het aangegeven punt het grootste gedeelte wel klaar zou moeten zijn, kleine bijkomstigheden in de toekomst zouden verder geen probleem moeten vormen.
In het plan van aanpak zijn ook de bekende data van mijlpalen opgenomen. Hieronder valt bijvoorbeeld de datum dat pilot 1 en het gehele project afgerond zou moeten zijn.

3.5 Bijhouden van uren

In de planning was per week aangegeven hoeveel uur voor een bepaalde activiteit gereserveerd stond. Om de ook daadwerkelijk gebruikte uren bij te houden, stelde de opdrachtgever voor om dit te doen met behulp van een algemeen Excel sheet.

Week	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Oriëntatie	14																	
Plan van aanpak	10	2																
Kennismaking Delphi	16	16																
Interviews			2															
Definitiestudie		8	4	4	2				10				6		4			
Datamodel		8	4							2								
Pilotontwikkeling			28	32	34	36	36	28	26	36	34	32	26	16	26	24	4	
Pilottest				2				4				4		4		8		
Pilotinvoering																	12	
Projectorganisatie		2						4			4				2			
Eindverslag		4	2	2	4	4	4	4	4	2	2	4	10	30	24	8	24	16
	40	40	40	40	40	40	40	40	40	40	40	40	42	50	56	40	40	16

Figuur 3: Urenschema

Een Excel sheet met dezelfde opmaak wordt ook gebruikt door andere stagiaires en afstudeerders, waardoor het voor de opdrachtgever makkelijker wordt om inzicht in de verbruikte tijd te krijgen. Een grijs gekleurde weeknummer staat voor een verstreken week, de aangegeven tijd die hierbij staat is de werkelijk gebruikte tijd voor een onderdeel. Voor de nog resterende weken betreft het een schatting van de benodigde tijd zoals deze in het begin van het afstudeertraject is opgesteld.

4 Aanvangssituatie

In dit hoofdstuk worden de relevante onderdelen besproken zoals deze aanwezig waren tijdens de start van het afstudeertraject. Denk hierbij aan de benodigde prestatie berichten en de huidige applicatie en de reeds aanwezige geanalyseerde gegevens volgens de huidige situatie.

4.1 Prestatie berichten (MaDo)

Het gehele afstudeerproject berust op de technische prestaties van de schepen van Spliethoff. Deze berichten worden door elk schip per telex of e-mail verstuurd op elke maandag en donderdag in de week, daarom ook de naam MaDo (MonThu in het Engels). Overigens komen de telex berichten als e-mail binnen bij de IT afdeling van Spliethoff.
In deze berichten staan allerlei actuele technische gegevens van een schip zoals bijvoorbeeld de turbo druk en de diepgang van een schip. De wijze waarop deze berichten opgesteld moeten worden, zijn beschreven in het OBA (Orders van Blijvende Aard) Booklet Spliethoff.

```
 1)  STADION
 2)  12/02/2004
 3)  35-45N 022-18W
 4)  2255
 5)  ROTTERDAM | P18/02/04/12:55 B18/02/04/15:15
 6)  58.5
 7)  98.5
 8)  .98
 9)  28000/28100
10)  45
11)  398
12)  0.75
13)  790/840
14)  NW-LY SWELL
15)  -0.25
16)  11.5
17)  PTO ON
```

Voorbeeld 1: Opmaak van een MaDo bericht

Tijdens het invoeren van deze gegevens aan boord van het schip is er echter geen controle op de wijze waarop het bericht is opgesteld. Dit resulteert er in sommige gevallen in dat er bijvoorbeeld door een typfout een MaDo bericht met een onjuiste waarde verstuurd wordt. Inmiddels is er een nieuwe applicatie ontwikkeld waarin er wel controle plaatsvindt, waardoor de kans op fouten in MaDo berichten zeer verkleind zullen worden. Echter, deze nieuwe applicatie is nog niet ingevoerd, dat zal pas gebeuren wanneer het nieuwe MaDo inlees- en analyse deel ingevoerd is.

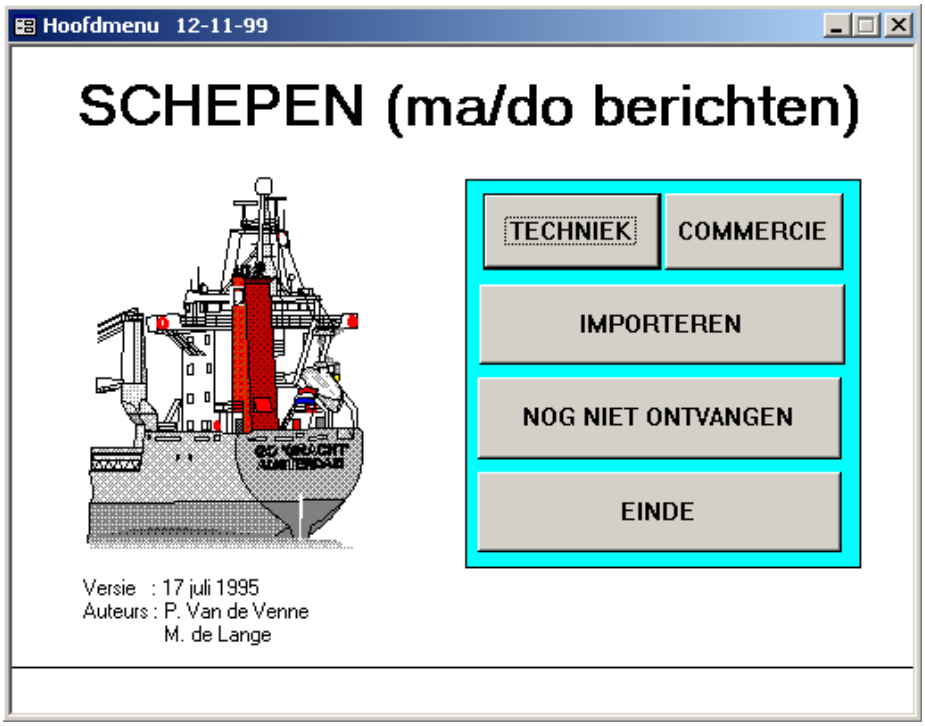
4.2 Inlees en analyse module

Voor het inlezen en analyseren van de aangeleverde MaDo berichten vanaf de schepen werd gebruik gemaakt van een module geschreven in Microsoft Access. Deze module analyseerde alle e-mail berichten in een vooraf gedefinieerde map in een mailbox. De analyse controleert de schrijfwijze van de MaDo gegevens in de berichten op basis van syntax fouten en waarden. Er is een functie ingebouwd waarmee veel voorkomende fouten gedetecteerd kunnen worden. Wordt een veel voorkomende syntax fout gedetecteerd, dan wordt deze door de analyse module, wanneer mogelijk hersteld, denk hierbij aan bijvoorbeeld de letter “o” in plaats van het cijfer “0” bij een positie waarde van een schip. Na deze syntax check wordt de waarde ten opzichte van vaste gegevens gecontroleerd en wanneer nodig wordt er een melding gegeven van de onjuistheid hiervan. Het grootste

probleem van de analyse module is de ontbrekende optimalisatie voor de onjuiste en eventueel reeds voorkomende MaDo berichten zoals hierboven beschreven. Hierdoor kwam het vaak voor dat de gegevens niet goed opgeslagen werden wat erin resulteerde dat overzichten niet meer het juiste beeld gaven. Deze module zorgt er ook voor dat de geanalyseerde gegevens opgeslagen worden in de database.

4.3 Bestaande kijk en bewerkingsapplicatie

De reeds bestaande applicatie voor het bekijken van de MaDo gegevens is aanwezig in de vorm van een Microsoft Access applicatie. Deze applicatie had toegang tot tabellen binnen Microsoft SQL Server voor het opvragen van de gewenste MaDo gegevens.



Figuur 4: Oude MaDo applicatie

Binnen de applicatie heeft de gebruiker de mogelijkheid om per schip de geanalyseerde MaDo berichten te bekijken. Tevens kan door de gebruiker tussen voorgedefinieerde grafieken gekozen worden om te tonen of af te drukken.
Echter wordt door de technische dienst geen gebruik meer gemaakt van de applicatie. De MaDo berichten worden nog wel aangeleverd, maar verder niet meer geanalyseerd.

4.4 Aanwezige MaDo gegevens

Voordat er gestopt werd met het werken met de MaDo Access applicatie werden alle berichten nog geanalyseerd en ingelezen in de Microsoft SQL Server database. Hierdoor zijn er al zeer veel MaDo gegevens aanwezig welke niet verloren mogen gaan. Deze gegevens zouden geconverteerd moeten worden naar de uiteindelijke nieuwe database om zo gebruikt te kunnen worden bij de nieuw te ontwikkelen applicatie. Tevens moeten deze gegevens ook gecontroleerd worden door de nieuw te ontwikkelen analyse module zodat ook bij deze oude gegevens de afwijkende waarden gedetecteerd kunnen worden.

4.5 Fleetcom, het systeem van Spliethoff

Veel applicaties bij Spliethoff worden door de eigen programmeurs gemaakt. Op deze manier kan er goed ingegaan worden op de specifieke eisen en wensen van de gebruikers en kunnen eventuele wijzigingen snel aangebracht worden. In plaats van dat elke ontworpen applicatie los op de systemen bij de gebruikers gezet worden, zijn deze samengevoegd in één schil, Fleetcom.

Figuur 5: Hoofdscherm van Fleetcom

Niet iedere gebruiker krijgt toegang tot alle modules binnen Fleetcom. Met behulp van een autorisatie tabel is per gebruiker binnen Spliethoff aangegeven wat voor soort rechten deze heeft en op basis daarvan wordt de mogelijkheid tot het starten van een module aangegeven in de menubalk. Het is de bedoeling dat de MaDo applicatie een module binnen Fleetcom zal worden.

4.6 Client-server met webservices

Binnen Spliethoff wordt gebruik gemaakt van een 3-tier client-server architectuur met webservices. Vele ontwikkelde programma's maken gebruik van deze manier van netwerkcommunicatie. Het voordeel van webservices is dat deze te gebruiken zijn in combinatie met een breed scala van applicaties op bijvoorbeeld computers, PDA's en zelfs GSM's. Een ander voordeel van webservices is dat het server gedeelte in een andere programmeertaal geschreven kan zijn dan het client deel, waardoor je het kunt hergebruiken. Het gebruik van webservices houdt ook in dat je op deze manier het zware werk door de

server kan laten doen. De client roept een functie op de server aan, welke voor de verdere afhandeling zorgt. Op deze manier kan met de client ongestoord verder gewerkt worden. Voor het gebruik van webservices zijn enkele technologieën beschikbaar welke in de volgende subparagrafen globaal beschreven worden. Voor een gedetailleerde omschrijving van SOAP wordt verwezen naar de internetsite http://www.w3.org/TR/soap/.

4.6.1 SOAP

SOAP staat voor Simple Object Access Protocol en is een lichtgewicht protocol voor de uitwisseling van gestructureerde gegevens in een gedecentraliseerde omgeving in XML formaat. SOAP kan gebruikt worden over verschillende soorten protocollen zoals TCP, HTTP en SMTP, bij Spliethoff wordt het HTTP protocol gebruikt.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV=”http://schemas.xmlsoap.org/soap/>
        <SOAP-ENV:Body>
                <m:Voorbeeld xmlns:m=www.spliethoff.com>
                        <Param1>12</Param1>
                        <Param2>Testgebruiker</Param2>
                </m:Voorbeeld>
        </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Voorbeeld 2: Een SOAP bericht

Een SOAP bericht bestaat uit twee delen, een “Header” en een “Envelope”. De “Header” is optioneel en bevat aanvullende informatie over een RPC (Remote Procedure Call). In de “Envelope” staat het overige deel van het bericht met de daadwerkelijke methode aanroep in de “Body”.

4.6.2 WSDL

WSDL (Web Services Description Language) wordt gebruikt door de server waar de webservice op draait om aan de buitenwereld duidelijk te maken hoe de structuur van de service is. Het server systeem genereert op basis van de webservice een WSDL bestand welke door een client systeem ingeladen moet worden. Hierna kan de client gebruik maken van de aangeboden services. De opmaak van WSDL is net als bij SOAP in het XML formaat.

5 Definitiefase analyse scheepsprestaties

Dit hoofdstuk geeft de beschrijving zoals het proces heeft plaatsgevonden tijdens de definitiefase van het afstudeertraject. Onderdelen die aan bod komen zijn onder andere het bepalen van het ontwikkelscenario, het opstellen van de systeemeisen en de organisatorische gevolgen.

5.1 Activiteiten binnen de definitiestudie

Binnen de definitiestudie van de IAD methode vinden verschillende activiteiten plaats die allen bijdragen aan het resultaat van de fase. Niet alle activiteiten hoeven ook daadwerkelijk doorlopen te worden, voor het project zijn de volgende activiteiten van de definitiestudie uitgevoerd:

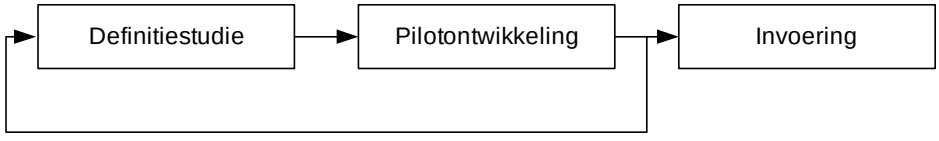
- Bepalen ontwikkelscenario;
  {het vaststellen van de context van het ontwikkeltraject}
- Opstellen systeemeisen;
  {het opstellen en actualiseren van een geprioriteerde lijst van systeemeisen}
- Bepalen systeemconcept;
  {het modelleren van de functionaliteiten vanuit het oogpunt van de gebruiker}
- Beschouwen technische structuur;
  {definiëren van de aanwezige hard- en software vereist voor het project}
- Beschouwen organisatorische inrichting;
  {het vaststellen van wat de impact van de in te voeren pilots op de organisatorische inrichting zal hebben}
- Definiëren pilotplan.
  {het definiëren van de pilots op basis van het systeemconcept}

Het proces bij de bovenstaande activiteiten zal in de volgende paragrafen besproken worden.

5.2 Bepalen van het ontwikkelscenario

In deze fase van het de definitiestudie wordt de reikwijdte en de context van het te ontwikkeltraject vast gesteld. Denk hierbij ook aan de kenmerken van het ontwikkelproces zoals de gekozen iteratiestrategie en de definiëring van de te ontwikkelen pilots.

De iteratiestrategie binnen het IAD model bepaalt in welke stappen een systeem ontwikkeld zal worden. Het bestaat altijd uit drie delen, te weten de definitiestudie, de pilotontwikkeling en de invoering.
Er zijn vier soorten voorgedefinieerde iteratiestrategieën welke gebruikt kunnen worden. Dit is het evolutionair ontwikkelen, het incrementeel opleveren, het incrementeel ontwikkelen en het big-bang invoeren [Tolido – IAD, het evolutionair ontwikkelen van informatiesystemen]. Echter is het ook mogelijk om een andere variant te gebruiken welke helemaal toegespitst zal zijn op de organisatie. Na gesprekken met de opdrachtgever en de hoofdprogrammeur zijn enkele punten naar boven gekomen waarna is gekozen voor de big-bang invoering strategie.



Figuur 6: Gekozen iteratiestrategie

Deze strategie houdt in dat er na de definitiestudie en de pilotontwikkeling een mogelijke iteratie plaatsvindt en wederom de definitiestudie en pilotontwikkeling doorlopen worden. Afhankelijk van het systeem kunnen iteraties meerdere malen plaatsvinden. Wanneer het systeem volledig ontwikkeld is, wordt deze in één keer ingevoerd in de organisatie.

Twee punten waarbij de Big-bang invoering van pas komt in relevantie met het afstudeertraject:

- *Snelle oplevering is niet nodig;*
  Dit is van toepassing tijdens het afstudeertraject, er bestaat geen urgentie voor de invoering van de te ontwikkelen onderdelen.

- *Organisatie niet veranderingsgereed;*
  Hoewel de opdrachtgever wel veranderingsgereed is, is de uiteindelijke gebruiker dit niet, deze wil meteen een volledige applicatie hebben. Het is bekend wat de gebruiker wil hebben, en verder wil deze niet veel te maken hebben met de ontwikkeling van de applicatie.

Een mogelijk nadeel van de gekozen iteratiestrategie zal in de praktijk meevallen:

- *Te weinig feedback uit de praktijk;*
  Ook al is er weinig communicatie met de toekomstige gebruiker, de opdrachtgever zelf weet ook wat het systeem moet kunnen. Hierdoor zijn vele testmomenten mogelijk om zo de tot dan toe volbrachte onderdelen te toetsen aan de eisen en wensen.

5.3 Opstellen functionele en non functionele systeemeisen

De systeemeisen vormen de weergave van de organisatie over de verwachting van de te ontwikkelen applicatie. Dit deel binnen de definitiefase is zeer belangrijk, omdat het de basis is voor de ontwerpactiviteiten.

Voorafgaand aan het opstellen van de eisen is eerst de bestaande inlees en analyse functie alsmede de bestaande Microsoft Access bewerkingsapplicatie bekeken. De reden hiervoor is dat hiermee een betere indruk van de toekomstige functionaliteiten verkregen kan worden. Het reeds bestaande systeem was echter niet meer volledig in gebruik, echter de prestatieberichten vanaf de schepen werden nog wel verstuurd. Deze werden opgevangen in een database zodat ze in een later stadium alsnog verwerkt zouden kunnen worden.

De systeemeisen zijn verdeeld in twee hoofdsoorten van eisen, te weten de functionele en non-functionele eisen. Zoals de naam al zegt hebben de functionele eisen puur invloed op de functies die het toekomstige systeem moet leveren en de non-functionele eisen niet.
De non-functionele eisen zijn derhalve weer onderverdeeld in onder andere de interface-, integriteits- en performance eisen.

Het opstellen van deze eisen is gedaan aan de hand van de opdrachtomschrijving verkregen van Spliethoff en een aanvullend gesprek met de opdrachtgever. Voorafgaand aan dit gesprek is een lijst opgesteld met mogelijke eisen. Deze zijn in samenspraak met de opdrachtgever aangevuld of verwijderd en er zijn enkele eisen toegevoegd. Tevens, door de gekozen iteratiestrategie, zijn enkele functionele eisen na het testen van een pilot naar boven gekomen. In tweede instantie zijn de eisen gecontroleerd door de hoofdprogrammeur. Na enig commentaar en een verdere uitleg met betrekking tot de eisen en wensen zijn deze op enkele gedeelten bijgesteld.

Een voorbeeld van een gedefinieerde functionele systeemeis:

“De gebruiker moet de MaDo gegevens kunnen aanpassen van een geselecteerd schip.”

Hiermee wordt bedoeld dat een geautoriseerde gebruiker nadat een MaDo record geanalyseerd en opgeslagen is, enkele gegevens hiervan ook nog kan wijzigen. De reden hiervoor is dat de analyse functie een waarde niet goed heeft kunnen analyseren omdat de syntax fout is. De gebruiker zou dan bij het bekijken van de gegevens deze fout kunnen herstellen aangezien de gebruiker de fout in de syntax wel herkend. Overigens wordt hier dan vastgelegd door wie en wanneer een wijziging heeft plaatsgevonden.

5.4 Bepalen van het systeemconcept

In het systeemconcept wordt ingegaan op de oplossingen beschreven vanuit het gezichtspunt van de gebruiker. Binnen het systeemconcept is ervoor gekozen om gebruik te maken van use-cases en domein-klassendiagram en een use-case diagram.

5.4.1 Gedefinieerde use-cases

Om een beeld te vormen van de interacties die tussen de gebruiker en het systeem plaats zullen vinden, zijn use-cases gemaakt. Use-cases worden geschreven in natuurlijke taal, waardoor deze ook zeer geschikt zijn om gebruikers globaal te laten zien hoe het systeem zal gaan werken.

De use-cases zijn gebaseerd op de reeds eerder gedefinieerde functionele systeemeisen. Op deze manier worden alle eisen met betrekking tot de werking van het toekomstige systeem beschreven. In totaliteit zijn er voor de te ontwerpen applicatie zestien use-cases geschreven, te weten:

1. Analyseren van MaDo berichten;
2. Opslaan van MaDo berichten;
3. Bekijken van technische MaDo gegevens;
4. Bekijken van commerciële MaDo gegevens;
5. Aanpassen van MaDo gegevens;
6. Origineel MaDo bericht weergeven;
7. Bekijken van MaDo grafieken;
8. Tonen van afwijkende en laatst ontvangen berichten;
9. Afdrukken van MaDo gegevens;
10. Afdrukken van MaDo grafieken;
11. Wijzigen van scheepsserie of status;
12. Autoriseren per gebruiker;
13. Autoriseren per serie;
14. Aanpassen van grenswaarden;
15. Aangeven van afwijkingssignalering;
16. Signaleren van afwijkingen.

De controle of alle eisen daadwerkelijk beschreven worden in de use-cases vind plaats tijdens de validatie, een later stadium binnen de definitiefase.

Naam	*Bekijken van technische MaDo gegevens.*
Samenvatting	*Het bekijken van de technische MaDo gegevens van een schip.*
Actoren	*Gebruiker.*
Beschrijving	*(1) De gebruiker geeft aan technische MaDo gegevens te willen bekijken;* *(2) Het systeem opent een venster waarin de scheepsseries waartoe de gebruiker rechten heeft getoond worden;* *(3) De gebruiker selecteert binnen een scheepsserie het gewenste schip;* *(4) Het systeem toont van het geselecteerde schip de naam en een overzicht van de technische MaDo gegevens (datum, positie, mijlen, bestemming, loodsdatum, kadedatum, rackstand, schroefstand, turbo druk, turbo A toeren, turbo B toeren, oplaadtemperatuur, uitlaattemperatuur, drukverschil, diepgang voor, diepgang achter, weertype, stroomsnelheid, grondsnelheid, opmerkingen) van het afgelopen jaar met een indicatie of de gegevens binnen de grenswaarden voor de betreffende serie vallen;* *(5) De gebruiker heeft de mogelijkheid een ander van- en tot datum aan te geven, wanneer de gebruiker hiervoor kiest gaat de use-case verder naar stap 6;* *(6) Het systeem haalt alle records op binnen het geselecteerde tijdsbestek.*
Uitzonderingen	
Resultaat	*De gewenste technische MaDo gegevens voor het geselecteerde schip worden getoond aan de gebruiker.*

Voorbeeld 3: Use-case in de definitiestudie

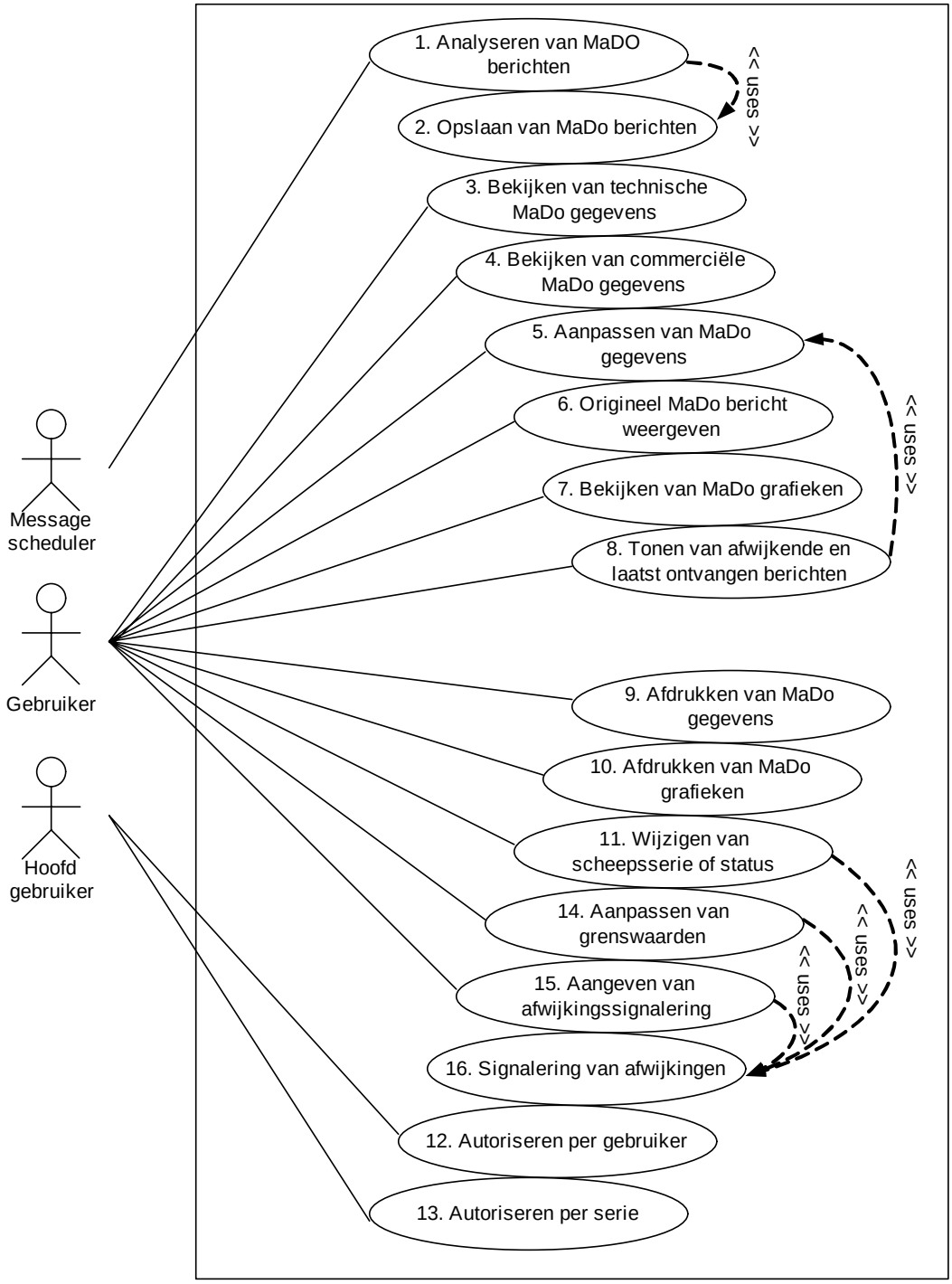
De bovenstaande use-case (Bekijken van technische MaDo gegevens) is opgesteld naar aanleiding van de volgende functionele systeemeis:

“De gebruiker moet de technische MaDo gegevens kunnen bekijken van een geselecteerd schip.”

Evenals bij het opstellen van de systeemeisen, zijn de beschrijvingen van de use-cases opgesteld nadat er gesproken was met de opdrachtgever en de hoofdprogrammeur. Zo is zelf gekomen met voorbeelden hoe een bepaalde systeemeis gedekt kon worden, waarna de mening en eventueel aanvulling gevraagd is. Met de eventuele aanvullingen zijn de use-cases aangepast en is wederom de situatie voorgelegd. Door het duidelijk aangeven van de gewenste stappen qua interactie tussen de gebruiker en het systeem, heeft het weinig moeite gekost om tot de use-case beschrijvingen te komen.

5.4.2 Use-case diagram

Het laatste onderdeel tijdens het bepalen van het systeemconcept is het opstellen van het use-case diagram. Zoals de naam al doet vermoeden, worden in dit diagram grafisch alle use-cases aangegeven met de bijbehorende actoren en hun verbindingen.



Figuur 7: Use-case diagram

In Figuur 7 zijn drie actoren voor het systeem te zien, de message scheduler, de gebruiker en de hoofdgebruiker. De message scheduler is een ander programma dat gebruik maakt van de te ontwikkelen analyse functie. De gebruiker kan gezien worden als een medewerker van de Technische Dienst en de hoofdgebruiker is de manager van de Technische Dienst of een ander geautoriseerd persoon.

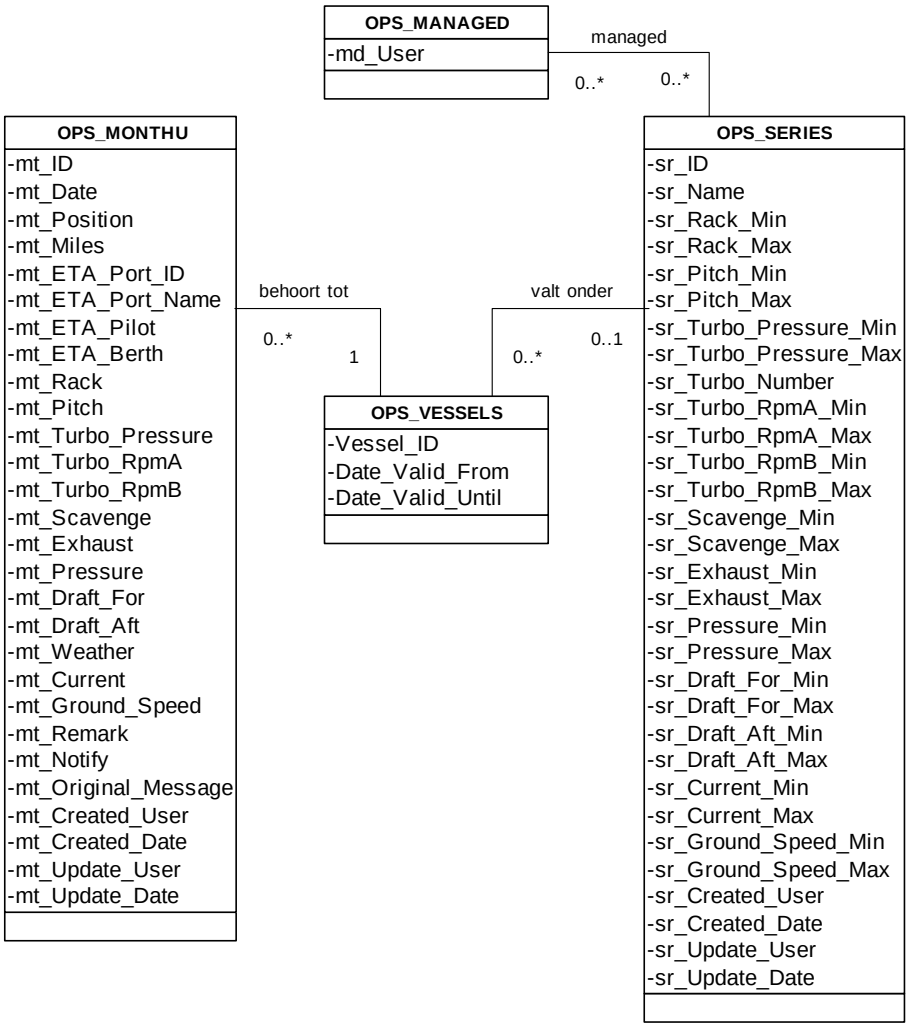
Door middel van dit diagram komt er niet meer nieuwe informatie naar boven, het toont slecht een gestructureerd overzicht van de koppeling tussen de actoren (message scheduler, gebruiker en hoofdgebruiker) en de interacties. Verder is in dit diagram te zien dat use-cases ook van elkaar gebruik maken. Dit is aangegeven door middel van “<< uses >>” en een gestreepte lijn. Binnen een use-case zelf wordt dit aangegeven als “gaat verder naar stap 1 van de use-case …”. Op deze manier kan je losstaande use-cases meerdere malen gebruiken.

5.4.3 Ontwerp van het domein klassendiagram

Het domein klassendiagram toont de structuur van de gegevens welke in het toekomstige systeem gebruikt zullen worden. Naast de klassen zelf wordt ook het verband ertussen getoond. Met behulp van een domein klassendiagram kan de database structuur gedefinieerd worden. Het domein klassendiagram verschilt van een normaal klassendiagram door het ontbreken van de operaties. Het domein klassendiagram kan gezien worden als een andere tekenwijze van een ER model [Vandenbulcke – Databasesystemen voor de praktijk]. De keuze voor het domein klassendiagram in plaats van een ER model is gemaakt op basis van de leesbaarheid van beide modellen, waarbij het domein klassendiagram persoonlijk de voorkeur krijgt.

Het domein klassendiagram is opgesteld aan de hand van de gemaakte use-cases. Om tot de gewenste klassen te komen, wordt aan de hand van de use-cases een selectie gemaakt van kandidaat klassen. Dit gebeurt door alle zelfstandige naamwoorden in een use-case te selecteren. Hierna worden alle kandidaat klassen geselecteerd op basis van relevantie en de duidelijkheid ervan. Het kan ook zijn dat een zelfstandig naamwoord geen klasse maar een attribuut is, dit wordt in deze zelfde stap beschreven. Na deze selecties worden de associaties tussen de klassen bepaald. Dit gebeurd op basis van de werkwoorden in de use-cases, maar niet elk werkwoord hoeft ook daadwerkelijk te betekenen dat het een associatie is. Na deze stappen komt het domein klassendiagram naar boven.

Nadat de eerste versie van het domein klassendiagram gemaakt was, is deze met de opdrachtgever en de hoofdprogrammeur doorgenomen. Een punt dat aan de orde kwam was de benamingen van de attributen. Het was een vereiste dat Engelse benamingen gebruikt zouden worden, dit aangezien Spliethoff een internationaal bedrijf is. Mede hierdoor is tijdens een latere versie van het domein klassendiagram de benaming hierop aangepast. Als voorbereiding op de database werden de benamingen van de klassen ook aangepast.
Bij Spliethoff is er een standaard voor de benaming, het bestaat enkel uit hoofdletters en wordt voorafgegaan door drie letters voor een aanduiding tot welk project het behoort. In dit geval was de aanduiding “OPS” wat staat voor “Operations”, omdat het een in een later stadium een database wordt voor de operationele kant van het bedrijf.
Zoals in het domein klassendiagram te zien is, is gekozen om de attributen te laten beginnen met een aanduiding tot welke klasse deze behoort. Hier is met het oog op een latere omzetting naar de database voor gekozen. Het kan voorkomen dat gegevens uit meerdere tabellen, in één enkel SQL statement opgevraagd worden. Mocht in een tabel het attribuut “Date” voorkomen en in een andere ook, dan moeten deze in het SQL statement vooraf gegaan worden door de tabel naam om duidelijk aan te geven welke benodigd is. Met deze naamswijziging zal dat in de meeste gevallen niet meer voor hoeven te komen, enkel bij het scheepsnummer (VesselID). Dit attribuut krijgt geen aanduiding vooraf omdat deze naam uniform gehouden wordt aan benaming in de overige databases binnen Spliethoff.



Figuur 8: Het domein klassendiagram

De ontwikkeling van het domein klassendiagram is dankzij de duidelijkheid van de functionele systeemeisen en use-cases soepel verlopen.

5.5 Technische structuur voor het systeem

Bij de technische structuur wordt geïnventariseerd welke hardware en software aanwezig en nodig zijn voor het te ontwikkelen systeem. Ook wordt er gekeken naar de mogelijkheid van het herbruiken van reeds eerder ontwikkelde componenten.

Om een overzicht te krijgen van de aanwezige hardware en software, welke relevant is voor het te ontwikkelen systeem, is gesproken met de systeembeheerder en met de hoofdprogrammeur. De te gebruiken besturingssystemen zijn allen afkomstig van Microsoft, het betreft Windows 2000 Server voor de servers en Windows 2000 Professional voor de werkstations. Het DBMS dat gebruikt wordt is eveneens van Microsoft; SQL Server 2000. Voor de ontwikkeling van applicaties wordt binnen Spliethoff gebruik gemaakt van Borland Delphi 7, in combinatie met enkele extra aanvullingen. Deze aanvullingen moeten gezien worden als 3rd-party grids of knoppen welke gebruikt kunnen worden tijdens de ontwikkeling.

Er wordt binnen Spliethoff gebruik gemaakt van een 3-tier communicatie model tussen de verschillende clients en servers. Hierop wordt in een volgend hoofdstuk verder ingegaan.

Omdat alle benodigde hardware en software al aanwezig waren, hoefde geen nieuwe systemen of versies van bepaalde software aangeschaft te worden.

5.6 Organisatorische gevolgen

Wanneer er een nieuw systeem in een organisatie ingevoerd wordt, zou dit grote gevolgen voor werkzaamheden kunnen betekenen. Verder moet ook duidelijk zijn of gebruikers cursussen of handleidingen nodig hebben.

De te ontwikkelen applicatie zal geen grote gevolgen voor de organisatie hebben. Het is puur voor de technische dienst van Spliethoff ontwikkeld. Deze gebruikers zullen voor hun werk een ondersteunende applicatie krijgen waardoor het werk eventueel sneller gedaan kan worden.
Ook cursussen zullen niet nodig zijn, de nieuwe applicatie werkt in grote lijnen hetzelfde als de oude Access applicatie, hierdoor zullen de gebruikers snel de nieuwe applicatie beheersen. Wel is voor de zekerheid gekozen om een helpfile te ontwikkelen. Zo zou een gebruiker bij enige twijfel kunnen opzoeken wat voor gevolgen bijvoorbeeld een wijziging heeft.

5.7 Definiëren van de pilotplannen

De volgende fase tijdens de definitiestudie is het definiëren van het pilotplan. Een pilotplan bestaat uit een lijst van pilots die ontwikkeld gaan worden. Per pilot wordt een overzicht gegeven van de use-cases uit het systeemconcept welke gebruikt gaan worden. Aan de hand van deze use-cases wordt er een tijdsschatting gegeven.

Op basis van de use-cases is er voor gekozen om het ontwikkeltraject te splitsen in drie pilots:

- Inlezen en analyseren gegevens
- Bewerken en tonen gegevens
- Aanmaken van rapporten

Dit zijn tijdens de ontwikkeling en ook voor het gebruik duidelijk verschillende soorten functionaliteiten. Het inlezen en analyseren gebeurt volledig automatisch door het systeem. Het bewerken en het tonen van de gegevens met alles wat daarbij hoort zoals het autoriseren van gebruikers is eigenlijk de hoofdapplicatie. Het aanmaken van rapporten is mede door het gebruik van een ander programma gedefinieerd als een andere pilot. In plaats van het echte programmeren met Delphi, moet hier met behulp van Nevrona Rave Reports, rapporten gemaakt worden.

5.8 Validatie van systeemconcept

Het laatste gedeelte van de definitiestudie is de validatie. In dit hoofdstuk worden de systeemeisen tegenover de use-cases uitgezet. Op deze wijze is snel en overzichtelijk een beeld te krijgen welke systeemeisen door welke use-cases verwezenlijkt worden. Een tweede controle is het uitzetten van de use-cases tegenover de te ontwikkelen pilots.

Door een matrix te maken met de systeemeisen en use-cases kan door middel van een eenvoudige aanduiding aangegeven worden of een systeemeis de ja of de nee beschreven is in een use-case.

Use-case Systeemeis	Analyseren van MaDo berichten.	Afdrukken van MaDo gegevens.	Autoriseren per serie.	Aanpassen van MaDo gegevens.
Het systeem moet MaDo berichten kunnen analyseren op basis van de opmaak.	X			
De gebruiker moet MaDo gegevens kunnen afdrukken van een geselecteerd schip.		X		
De gebruiker moet de MaDo gegevens kunnen aanpassen van een geselecteerd schip.				X

Tabel 1: Voorbeeld validatie

In dit voorbeeld zijn de beschrijvingen van de systeemeisen en use-cases gebruikt, in werkelijkheid betreft het hier enkel het nummer van de systeemeis of use-case.

De bovenstaande werkwijze is ook gebruikt voor het uitzetten van de pilots ten opzichte van de use-cases.

Er is gebleken dat alle systeemeisen ook daadwerkelijk beschreven worden in de use-cases. Er is één use-case meer dan dat er systeemeisen zijn. Deze use-case is een sub-case aangezien deze enkel vanuit andere use-cases aangeroepen wordt, het betreft hier de “Signaleren van afwijkingen” use-case.

6 Pilot 1: Het analyseren van prestatiegegevens

Dit hoofdstuk beschrijft de werkwijze en bevindingen welke gebruikt zijn bij de ontwikkeling van de analyse pilot. Het resultaat van deze pilot moet ervoor zorgen dat de aangeleverde prestatieberichten geanalyseerd en opgeslagen worden.

6.1 Voorbereidingen voor de analyse

Tijdens de definitiefase zijn eisen opgesteld welke ook betrekking hebben op deze pilot. Voordat er daadwerkelijk met de ontwikkeling van de pilot begonnen is, zijn er nog voorbereidingen geweest, welke hier beschreven zullen worden.

6.1.1 De programmeertaal Delphi

Bij Spliethoff wordt geprogrammeerd in Borland Delphi, dit is een taal afgeleid van Pascal. Om Delphi te leren kennen, is gebruik gemaakt van een syllabus programmeren van de Hogeschool InHolland Diemen. In deze syllabus staan allerlei voorbeelden voor beginners tot aan gevorderden. Zo zijn enkele voorbeelden overgenomen in Delphi waarna deze gestart werden om zo de werking ervan te aanschouwen.

```
if achternaam = 'Testgebruiker' then
begin
        // Hier kan code gezet worden wat moet worden uitgevoerd
        // wanneer de voorwaarde voldoet
end
else
begin
        // En hier is de code te plaatsen wanneer de voorwaarde
        // niet voldoet
end;
```

Voorbeeld 4: Een if statement in Delphi

Door de aanwezige kennis in andere programmeertalen, te weten PowerScript, Java en C# was de overgang niet groot. In het voorbeeld is te zien dat bij een "if" statement voor een "else" geen puntkomma mag staan. Wanneer dit wel gedaan zou worden, dan zou dit betekenen dat de opdracht voor de "else" afgebroken wordt. Een puntkomma betekent in Delphi namelijk dat een opdracht afgesloten wordt. Echter zou de ontwikkelomgeving deze fout herkennen en er een melding van geven aan de programmeur. Dit zijn kleine punten en de gewenning hieraan is zeer soepel verlopen.

6.1.2 De opmaak van de prestatieberichten

Bij het verwezenlijken van de use-cases tijdens de definitiefase, zijn de prestatieberichten puur doorgekeken op welke informatie er in staat. De echte structuur is in deze ontwikkelingsfase bekeken.
Om inzicht te krijgen hoe de aangeleverde prestatie eruit zien qua opmaak is door Dhr. Kuiper een voorbeeld van een oud en een nieuw bericht geleverd. Het oude bericht komt uit de "Orders van Blijvende Aard" map van Spliethoff, de nieuwe is een export vanuit het systeem van Dhr. Kuiper. Tijdens de definitiefase is al kort uitgelegd wat de gegevens in de berichten betekenen, maar om hier een betere indruk van te krijgen is samen met Dhr. Kuiper de lijst doorgelopen en heeft hij waar nodig het één en ander toegelicht op technisch gebied.

Bij de opmaak van de berichten zelf, is te zien dat elke waarde vooraf gegaan wordt door een getal en een “)”. Dit is zowel het geval bij de oude en de nieuwe berichten. Hiermee zou dus een waarde geïdentificeerd kunnen worden.

Nieuw

1) STADION
2) 12/02**/2004**
3) 35-45N 022-18W
4) 2255
5) ROTTERDAM **| P18/02/04/12:55 B18/02/04/15:15**
6) 58.5
7) 98.5
8) .98
9) 28000/28100
10) 45
11) 398
12) 0.75
13) 790/840
14) NW-LY SWELL
15) -0.25
16) 11.5
17) PTO ON

Oud

1) STADION
2) 12/02
3) 35-45N 22-18W
4) 2255
5) ROTTERDAM 18/02 L:12:55
6) 58.5
7) 98.5
8) .98
9) **A=**28000 **B=**28100
10) 45
11) 398
12) 0.75
13) 790/840
14) NW-LY SWELL
15) -0.25
16) 11.5
17) PTO ON

Voorbeeld 5: Overzicht van een nieuw en oud bericht

Bij de berichten in Voorbeeld 5 is dikgedrukt het verschil aangegeven tussen de oude en de nieuwe opmaak. Hierbij moet wel aangemerkt worden dat dit een “goed geschreven” oud bericht is. In de praktijk komt het echter zeer vaak voor dat de berichten fouten bevatten op het gebied van de schrijfwijze. Zo kan ergens in plaats van een waarde bijvoorbeeld een streepje staan of is er helemaal niets ingevoerd. Hiermee zou tijdens de ontwikkeling rekening gehouden moeten worden.

6.1.3 Bekijken oude analyse module

Om inzicht te krijgen in hoe de analyse opgezet zou kunnen worden, is bij navraag de code van de oude analyse module door de opdrachtgever aangeleverd. Deze code is geschreven in Visual Basic en de eerste opzet stamt uit 1995.

```
If mDiepgvoor < 200 Or mDiepgvoor > 975 Then
        mMelding = mDiepgvoor
        x = Logging(mDatum, mMelding, mschip, mOrg, "Diepgvoor", mBestand)
        mError = mError + 1
End If
```

Voorbeeld 6: Grenswaarde controle bij oude analyse

Bij de bovenstaande code wordt gecontroleerd of de diepgang binnen de grenswaarden valt. Hier komt meteen een duidelijk nadeel van de oude module aan het licht, de grenswaarden staan vast in de code en is dus niet flexibel. Kijkend naar het domein klassendiagram uit de definitiestudie, is duidelijk dat dit probleem opgelost zal worden door de onderverdeling van de grenswaarden naar series.

Binnen de oude analyse wordt op basis van een nummer op een regel gevolgd door een “(“ de waarde doorgegeven aan een verdere controle zoals hierboven weergegeven is. Omdat deze kenmerken bij het analyseren van de berichten ook naar boven kwamen, zou dit gebruikt worden voor de herkenning van een MaDo waarde.

6.1.4 Opslag van de gegevens in een database

Voor de opslag van de technische gegevens wordt gebruik gemaakt van een database, en bij Spliethoff wordt hier Microsoft SQL Server 2000 voor gebruikt. Voordat met de pilot begonnen zou worden, zou eerst een database aangemaakt worden voor de opslag van de geanalyseerde gegevens. De structuur hiervan is gebaseerd op het domein klassendiagram zoals die in de definitiefase ontwikkeld is. Op basis hiervan is een representatiemodel gemaakt waarna een implementatiemodel is opgesteld. Met een implementatiemodel kan de database structuur ook daadwerkelijk in het DBMS geladen worden.

```
CREATE TABLE OPS_VESSELS(
        Vessel_ID           integer         not null,
        Series_ID           varchar(15)     ,
        Date_Valid_From     datetime        ,
        Date_Valid_Until    datetime        ,
PRIMARY KEY(Vessel_ID),
FOREIGN KEY(Series_ID) REFERENCES OPS_SERIES(sr_ID));
```

Voorbeeld 7: Create statement voor de database

Met behulp van het bovenstaande SQL script wordt binnen de database de OPS_VESSELS tabel aangemaakt met de verschillende attributen. Met behulp van de “foreign key” regel wordt aangegeven dat dit attribuut gekoppeld is aan de primaire sleutel van een andere tabel.

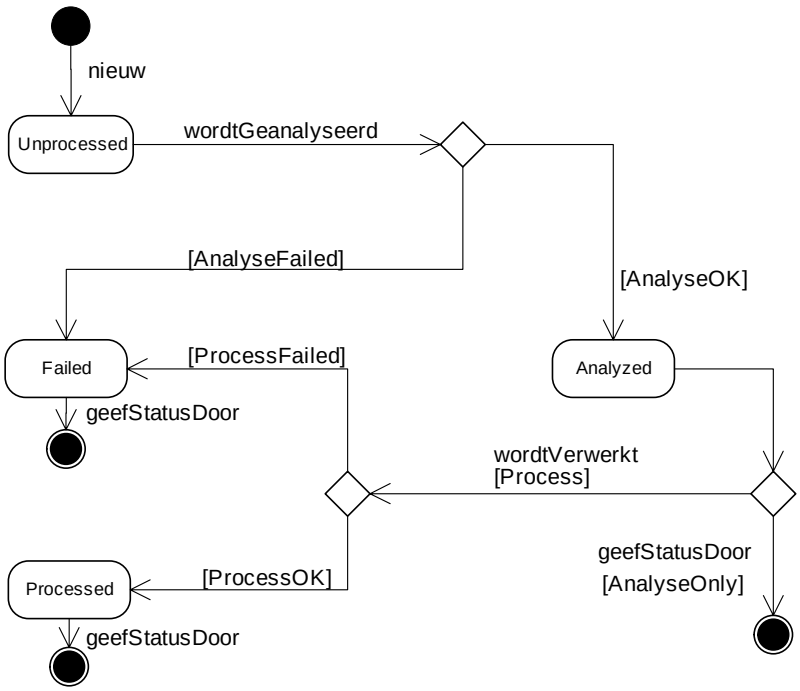
In een later stadium werd overigens geen gebruik meer gemaakt van het implementatie model voor het aanmaken en aanpassen van de tabellen. Aanpassingen op tabellen in de database werden via het visuele “SQL Server Manager” programma gedaan. Dit is een tool dat bij SQL Server geleverd wordt, met functionaliteiten voor het aanpassen van een tabel structuur. De reden hiervoor is dat deze aanpak eenvoudiger is dan iedere keer tabellen verwijderen en dan weer aanmaken via de implementatie code.

6.1.5 Test berichten in lokale omgeving

De scheepsberichten die digitaal naar Spliethoff verstuurd worden, worden opgeslagen in een “messages” tabel in de database. Zo worden niet alleen MaDo berichten, maar ook bijvoorbeeld aankomstberichten verstuurd. Vanuit deze tabel zullen de berichten verder verwerkt worden. Het aanleveren van berichten aan de analyse module zou in de productie omgeving plaatsvinden met behulp van de “message scheduler”. Dit is een door Spliethoff ontwikkelde tool welke aan de structuur van berichten herkent waarvoor deze zijn. Echter was tijdens de start van het ontwikkeltraject deze tool nog niet gereed. Dit had geen gevolgen voor het project aangezien door de hoofdprogrammeur een lokale berichten aanlever tool werd opgezet. Deze tool heeft verder geen intelligentie op het gebied van het bepalen van het doel van een bericht. Om nu te voorkomen dat deze tool alle berichten uit de “messages” tabel zou inlezen en doorsturen, is een lokale berichten tabel aangemaakt. Hierin zijn door de hoofdprogrammeur ongeveer 2200 berichten geplaatst welke tijdens de ontwikkeling van de analyse module gebruikt kunnen worden. Niet alle berichten waren ook daadwerkelijk MaDo berichten, waardoor ook meteen getest zou kunnen worden hoe de module hierop zou reageren.

6.2 Ontwerpen van de module

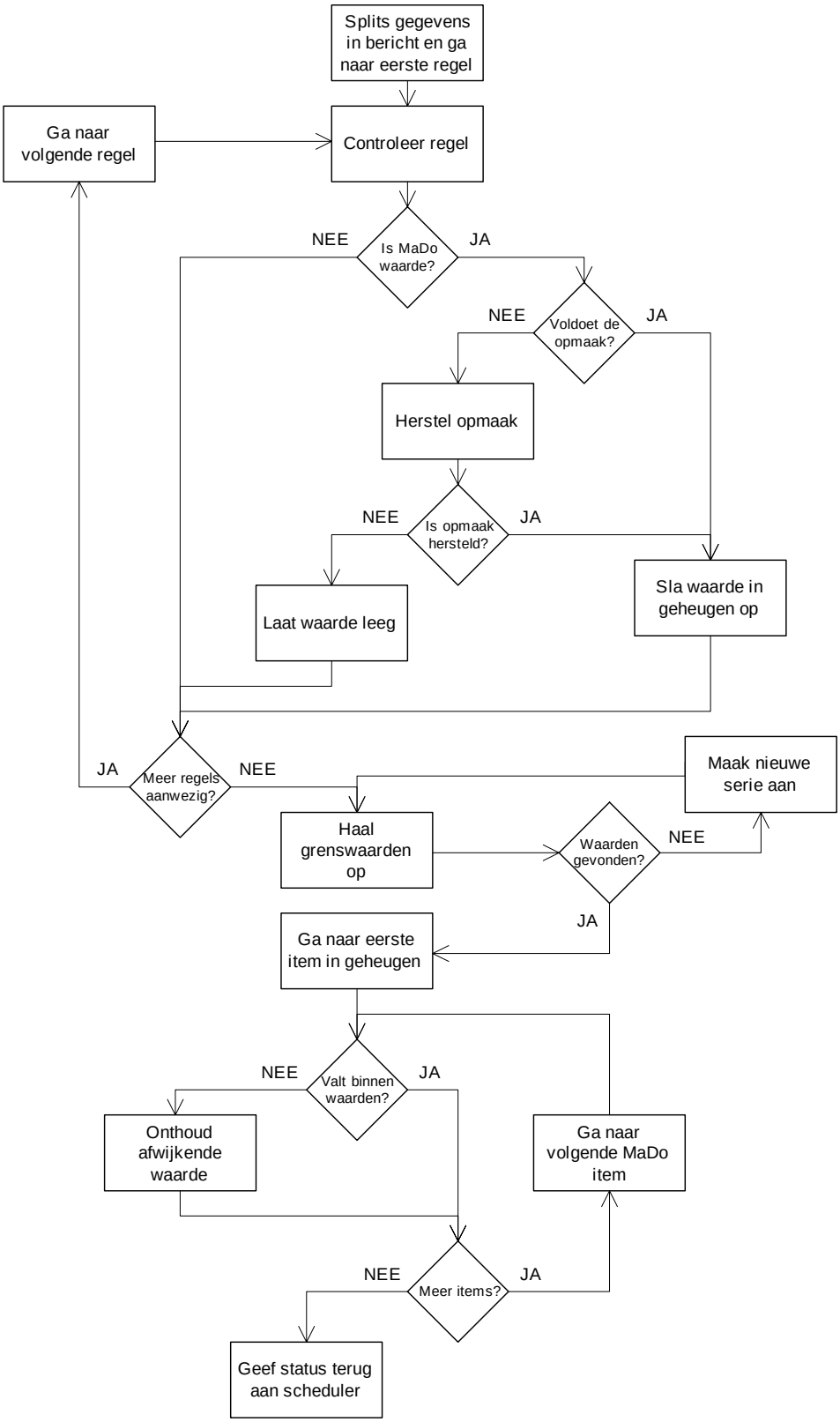
Volgens de richtlijnen bij Spliethoff wordt er aan een bewerkingsmodule doorgegeven of een bericht alleen geanalyseerd of geanalyseerd en ook opgeslagen moet worden. Na het verwerken van een bericht is het ook de bedoeling om de status van een bericht terug te geven. Om in kaart te brengen wat voor status het aangeleverde bericht zou kunnen hebben en in welke gevallen is hiervoor een toestandsdiagram opgesteld.



Figuur 9: Toestand van een bericht

Een geval waarin bijvoorbeeld de analyse zal mislukken is wanneer op basis van de scheepsnaam niet het bijbehorende scheepsnummer gevonden kan worden. Als reactie hierop zal de analyse module aan de message scheduler teruggeven dat de analyse van een bericht mislukt is. De message scheduler slaat de status op waarna later door een medewerker van Spliethoff handmatig gecontroleerd kan worden waarom een bericht niet goed door een analyse heen is gekomen.

Voordat begonnen werd met het daadwerkelijk programmeren van de analyse module is eerst de wijze waarop de analyse plaats zou vinden globaal uitgewerkt. Dit is gedaan met behulp van een flowchart. Hiermee is duidelijk te modelleren hoe een bepaald proces in elkaar moet komen te zitten.



Figuur 10: Flowchart voor de analyse

Op basis van de ontwikkelde flowchart is met de opdrachtgever en de hoofdprogrammeur besproken wat zij van het proces vonden. Het leek hun ook de beste manier om een bericht per regel te splitsen en daarna te controleren of het een MaDo waarde zou zijn, dit dus op basis van een nummer gevolgd door een “)”.

6.3 De ontwikkeling van de module

Nadat het gewenste proces in kaart was gebracht, is daadwerkelijk met de ontwikkeling van de analyse module in Delphi begonnen. Uiteindelijk zou de ontwikkelde module een webcomponent worden welke aangesproken zal worden met behulp van SOAP. In het begin van de ontwikkeling was dit niet het geval omdat het volgens de hoofdprogrammeur in het begin makkelijker was om het echt lokaal te ontwikkelen. Het idee was om eerst te zorgen dat berichten goed geanalyseerd zouden worden, waarna de module naar een webcomponent omgezet zou worden. Een tweede reden is dat je bij het volledig lokaal ontwikkelen gebruik kan maken van debugging, dit is het stap voor stap doorlopen van bepaalde delen code om problemen op te lossen. Je kunt op deze manier precies zien waar een bepaalde situatie fout gaat.

6.3.1 Lokale ontwikkeling

Gestart is met het eerst volledig kunnen analyseren van de berichten in de lokale database. Omdat in het begin ook oude berichten geanalyseerd moeten worden, moeten binnen de analyse ook syntax fouten hersteld kunnen worden. Hiervoor is per item waar de syntax zou kunnen afwijken een controle functie gemaakt.

```
if Data[6] = 'Z' then
        Data[6] := 'S';

if Data[14] = 'O' then
        Data[14] := 'E';
```

Voorbeeld 8: Syntax controle bij nieuwe analyse

In dit eenvoudige voorbeeld wordt gecontroleerd of de notatie van de positie van een schip voldoet. Een juiste positie heeft “42-10N 005-17E” als opmaak, dit zijn de coördinaten in noorderbreedte en oosterlengte. De code in Voorbeeld 8 herstelt de Nederlandse aanduiding van **Z**uid en **O**ost naar het Engelse **S**outh en **E**ast.

Door het steeds aanbieden van berichten aan de analyse module en het controleren waar nog eventuele fouten hersteld kunnen worden is de module geoptimaliseerd. Van de 1231 MaDo berichten in de database, de overige waren geen MaDo berichten, wordt bij 20 daarvan de datum van verzending niet juist herkend, wat betekent dat 98,4% wel goed herkend wordt. Met dit resultaat waren de opdrachtgever en hoofdprogrammeur tevreden, waardoor besloten werd de module om te zetten naar een webcomponent.

6.3.2 Omzetting naar webcomponent

Nu volgens de hoofdprogrammeur de analyse functie voldoende gegevens goed analyseerde en opsloeg, werd de module omgezet naar een webservice. Hiervoor werd Microsoft IIS op de lokale machine geïnstalleerd om de ontwikkeling nog lokaal te houden. De reden hiervoor was dat dit makkelijker was tijdens de ontwikkeling. Zo is het nodig, iedere keer nadat het webcomponent tijdens een test aangesproken is en er een nieuwe versie gecompileerd moet worden, de webservice herstart moet worden. Dit is een probleem met IIS waarvoor volgens de programmeurs bij Spliethoff geen oplossing is.

Tijdens de eerste test via de webservice bleek dat de analyse functie niet goed functioneerde. Hoe langer de analyse bezig was, hoe trager het systeem werd. Na het raadplegen van de Windows taskmanager was te zien dat al het geheugen van het systeem

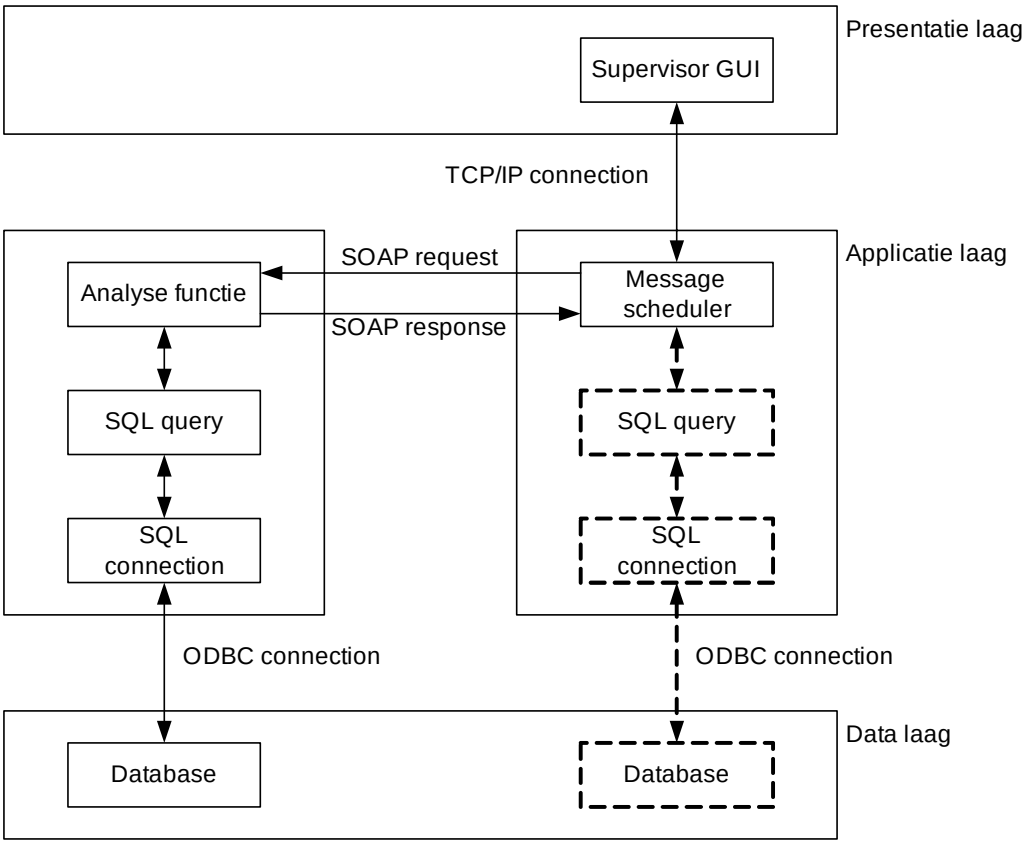
verbruikt werd. Een ontwikkelaar van de afdeling raadde aan om te controleren op “memory leaks”. Dit is een probleem dat ontstaat nadat geheugen dat gebruikt wordt door een programma niet meer vrijgegeven wordt. Met behulp van een bepaalde code werd gecontroleerd waar het geheugen vastgehouden werd, dit gebeurde door een tekstbestand aan te maken met daarin de naam van de functie welke voor de problemen zorgde. Door deze tekstbestanden goed door te nemen en de code van de analyse waar nodig aan te passen was dit probleem verholpen.

6.3.3 Conversie van de oude database

Nu de analyse module ook goed werkte via de webservice, werd met de opdrachtgever en de hoofdprogrammeur afgesproken dat de gegevens in de oude database omgezet zouden worden naar de nieuwe. Deze oude gegevens zouden dan ook langs de ontwikkelde analyse module moeten komen om ook hier meteen eventuele fouten uit de gegevens te halen. Deze conversie zou meteen een goede test zijn, aangezien het gaat om ongeveer 20.000 berichten. Wanneer de analyse functie hier niet op vast zou lopen, zou deze in de testomgeving ingezet worden.
Het was alleen de vraag hoe de gegevens uit de tabelstructuur zo aan de analyse aangeboden konden worden, zodat deze de gegevens als een bericht zou zien. Om dit voor elkaar te krijgen is eerst de structuur van de oude tabel bekeken. Op basis hiervan kon opgemaakt worden welke kolommen overeenkomen met regels in een origineel MaDo bericht. Met deze informatie is een functie gemaakt welke per record van de oude database de gegevens ophaalt en deze in een tekst variabele binnen Delphi (StringList) zo opstelt als in een MaDo bericht.
Deze tekst variabele is vervolgens aangeboden aan de analyse functie, waarna een identiek pad gevolgd wordt als bij de “normale” analyse. Dit proces van conversie was zeer tijdrovend door het grote aantal records, maar vooraf was in samenspraak met de opdrachtgever en hoofdprogrammeur afgesproken dat dit één keer plaats zou vinden. Mochten eventuele kleine fouten niet hersteld worden, dan was dit verder geen probleem aangezien de gegevens enkele jaren oud waren.

6.4 Invoering in testomgeving

Na de conversie van de oude database is de analyse module als DLL aangeboden aan de ontwikkelaar die intussen de message scheduler ontwikkeld had. Nu kon de werking van de analyse getest worden in een geïsoleerde omgeving met volledige functionaliteiten. De gebruikte berichten voor de test waren dezelfde als die gebruikt werden tijdens de lokale ontwikkeling, aangevuld met iets nieuwere berichten. De structuur van de netwerkcommunicatie werd op dit moment als in Figuur 11.



Figuur 11: Netwerkcommunicatie bij de analyse

De message scheduler haalt berichten op uit de database, waarna de tekst van het opgehaalde bericht doorgegeven wordt aan de analyse module. Deze gebruikt eigen SQL statements om zo gegevens op te halen voor de analyse en wanneer nodig voor het opslaan van de verwerkte gegevens.

Deze test was ook tegelijk bedoeld als stresstest aangezien de message scheduler de berichten per twintig tegelijk aanleverde. De analyse module bleek hier goed mee om te gaan waardoor de test een succes was. Een andere test op dit moment was die van de message scheduler, deze kon nu ook voor het eerste met een “echte” module getest worden. Na deze stap was de analyse module klaar voor de invoering in de productie omgeving.

7 Pilot 2: De MaDo applicatie

In dit hoofdstuk wordt het proces beschreven zoals deze gevolgd is bij de ontwikkeling van de applicatie voor het bekijken en bewerken van de MaDo gegevens. Enkele fasen die besproken worden zijn het ontwerp en de ontwikkeling. Ook wordt aangegeven op welke wijze er getest is.

7.1 De te bouwen functionaliteiten

Voor de ontwikkeling zijn de use-cases, gemaakt tijdens de definitiestudie, aangehouden als globale beschrijving van de gewenste functionaliteiten. Zo zouden de volgende use-cases tijdens de ontwikkeling van deze pilot technisch uitgewerkt worden:

- Bekijken van technische MaDo gegevens (3);
- Bekijken van commerciële MaDo gegevens (4);
- Aanpassen van MaDo gegevens (5);
- Origineel MaDo bericht weergeven (6);
- Bekijken van MaDo grafieken (7);
- Tonen van afwijkende en laatst ontvangen berichten (8);
- Wijzigen van scheepsserie of status (11);
- Autoriseren per gebruiker (12);
- Autoriseren per serie (13);
- Aanpassen van grenswaarden (14);
- Aangeven van afwijkingssignalering (15);
- Signaleren van afwijkingen (16).

Het nummer achter een use-case betreft het use-case nummer uit de definitiestudie. Op basis van deze gegevens en het pilotplan uit de definitiestudie is een planning gemaakt voor de volgorde waarop de ontwikkeling plaats zou vinden.

7.2 Structuur en eisen bij schermontwerp

Een belangrijk onderdeel bij een applicatie is het opzetten van de layout ervan. Zo kan een nog zo goed programma met bijvoorbeeld een slechte plaatsing van knoppen, minder prettig werken dan een slecht programma met een goede plaatsing.

Om hierop voorbereid te zijn, zijn delen van het hoofdstuk “User interface design” [Ian Sommerville – Software engineering] doorgenomen. Een belangrijk punt hierin is bijvoorbeeld dat het belangrijk is om de gebruikers niet te “verrassen” met een scherm. Zo moet ervoor gezorgd worden dat er structuur in zit, niet de ene keer een wit scherm en de andere keer weer een blauw scherm. De beschreven punten zijn handig om te onthouden, echter zou in eerste instantie de Spliethoff stijl gevolgd moeten worden. Enkele eisen waaraan de applicatie zou moeten voldoen:

- De applicatie moet in de Engelse taal gemaakt worden;
  {Spliethoff is een internationale organisatie, hierdoor kan het voorkomen dat er in de toekomst Engels sprekende gebruikers zullen zijn}
- De applicatie moet goed te zien zijn bij een schermresolutie van 800x600.
  {Nog een redelijk aantal systemen binnen het bedrijf hebben nog een schermresolutie van 800x600, zo ook de toekomstige gebruiker}

7.3 Schermontwerp voor de applicatie

Voordat er begonnen werd met het ontwerpen van de schermen voor de applicatie, is eerst gekeken naar de reeds bestaande applicatie. Erg veel informatie is hieruit niet gekomen aangezien die applicatie in Access gemaakt is, en niet uitgebreid is qua het aantal schermen. Op basis van de in de definitiefase verworven eisen, is te zien dat de gebruiker de technische gegevens van een schip zou moeten zien:

"De gebruiker moet de technische MaDo gegevens kunnen bekijken van een geselecteerd schip."

Om hiervoor verschillende vensters te ontwerpen is omslachtig, de ernst van een eventuele hoge waarde van een gegeven zou pas goed beoordeeld kunnen worden als deze in contrast gezien wordt met andere gegevens. Hierdoor was het idee om de gegevens in een lijst te plaatsen. Het exacte hoe en wat was echter nog niet bepaald.
Omdat de gebruikers ook gegevens moeten kunnen wijzigen, moet hiervoor de gelegenheid gegeven worden. Het is misschien mogelijk om de gebruiker de gegevens direct in het scherm te laten wijzigen, waar deze ze ook kan bekijken. Eventueel is het mooier om een apart scherm voor deze wijzigingsmogelijkheden te maken. Met deze punten is met de hoofdprogrammeur gesproken wat de beste mogelijkheid is. Deze gaf aan dat de toekomstige gebruiker niet veel zou moeten klikken met de muis. Hierdoor is besloten om de kijk en wijzig functionaliteiten op te nemen in één en hetzelfde scherm. Verder zouden ook bij de overige te ontwerpen schermen de kijk en wijzig functionaliteiten zoveel mogelijk samengevoegd moeten worden.

7.4 De ontwikkeling van het detailscherm

De kernfunctionaliteit van de te ontwikkelen applicatie is de gebruiker de mogelijkheid te bieden om de prestatiegegevens te bekijken en te bewerken. Zoals in paragraaf 7.3 besproken is, is er voor gekozen om deze functionaliteiten in één enkel scherm mogelijk te maken. Om aan de Spliethoff ontwikkeleisen te voldoen is met de hoofdprogrammeur gesproken of hier een standaard Delphi component voor gebruikt moet worden. Hierop is aangegeven dat de gegevens het beste getoond kunnen worden met behulp van een DBGrid. Om duidelijk te maken hoe de werking hiervan is, heeft de hoofdprogrammeur een voorbeeld hiervan laten zien binnen een applicatie die hij op dat moment aan het ontwikkelen was. Een voorbeeld van een DBGrid binnen de ontwikkelde applicatie is weergegeven op de volgende pagina op Figuur 12.

7.4.1 Het tonen van de prestatiegegevens

Voor het tonen van de gegevens is na advies van de hoofdprogrammeur dus gebruik gemaakt van een DBGrid. In zo'n component kan je gegevens tonen, maar wordt de gebruiker ook de mogelijkheid geboden deze te wijzigen. Verder kunnen de afwijkende waarden zoals deze geconstateerd zijn tijdens de analyse van een MaDo bericht duidelijk aangegeven worden.

Date	Rack	Pitch	Turbo			Scavenge	Exhaust	Pressure	Speed		Draft		Miles	Weather
			Pressure	RpmA	RpmB				Current	Ground	For	Aft		
24-05-2004	36.0	90.0	1.90	23000	23000	58	414	0.120	0.50	14.06	810	860	741	GOED, DEINING OOST/LA
20-05-2004	36.0	92.0	1.85	22500	22800	60	418	0.120	-1.00	13.88	810	860	2112	GOED, DEINING OOST/LA
03-05-2004	36.0	90.0	1.90	22800	23100	62	418	0.120	-0.50	12.48	840	860	404	GOED, DEINING NO/MATI
29-04-2004	36.0	87.0	1.90	22400	22900	59	410	0.120	-0.50	12.36	840	860	1593	GOED, DEINING NW/MATI
26-04-2004	36.0	91.0	1.95	22500	22800	60	407	0.120	-0.50	12.48	840	860	2328	GOED, DEINING WEST/MA
19-04-2004	36.0	92.0	2.00	22900	23000	58	403		-1.00	11.82	820	850	461	SLECHT TOT MATIG,ZW 7
29-03-2004	36.0	94.0	2.00	22800	23200	58	403		-0.50	11.60	745	890	286	MATIG,NE 8 BFT VOOR
25-03-2004	36.0	95.0	2.00	22800	23200	58	408		0.50	13.40	750	900	1508	GOED,NW 4 BFT SCHUIN
22-03-2004	36.0	96.0	1.90	22600	22800	56	410		0.50	13.70	750	900	2373	GOED, W 3/4 BFT VOOR
15-03-2004	36.0	96.0	1.90	22600	22800	54	412		0.50	14.30	400	510	976	GOED, SE 4 BFT VOOR
11-03-2004	36.0	96.0	1.90	22500	22700	56	415	0.110	-0.50	12.90	540	680	434	REDELIJK, NE 4-5 BFT VO
04-03-2004	37.0	96.0	1.90	22300	22600	56	405	0.100	-1.00	12.10	655	840	165	GOED, SE 4 BFT VOOR
01-03-2004	6.0	97.0	1.95	22400	22700	58	405		1.50	14.50	660	840	1077	GOED, NE 4 BFT ACHTERI
26-02-2004	35.0	91.0	1.80	22000	23000	56	405		0.00	11.00	740	820	470	MATIG, ZW 7 BFT TEGEN
09-02-2004	36.0	98.0	2.10	22900	23000	56	395		0.00	12.40	700	800	159	IJSVAART-GEEN WIND
05-02-2004	36.0	91.0	1.90	22300	22600	57	405		0.00	12.60	700	800	1494	REDELIJK,ZW 8 ACHTERI
02-02-2004	35.0	88.0	1.75	22200	22500	56	405		-0.50	12.60	700	800	2406	REDELIJK,ZW 6 SCHUIN A
29-01-2004	35.0	88.0	1.75	22200	22600	57	422	91.000	-0.50	12.30	700	800	3558	REDELIJK,ZW 6 SCHUIN A

Figuur 12: Grid met prestatiegegevens

In Figuur 12 is een grid weergegeven welke gebruikt is in het detailscherm, in samenspraak met de hoofdprogrammeur is ervoor gekozen om de afwijkende waarden in geel te tonen met rode letters. De gewijzigde waarden door een gebruiker worden met behulp van een blauw vak getoond. De reden voor de keuze van deze kleurencombinatie is dat een gele cel met rode letters snel opvalt en juist door de rode letters kenbaar maakt dat daar iets niet klopt. Blauw valt in een witte omgeving ook op, maar heeft een minder agressieve uitstraling. De opdrachtgever had aangegeven dat de naam van de huidige gebruiker en de huidige datum en tijd opgeslagen moesten worden, wanneer een gebruiker gegevens aan zou passen. Hiermee kan bij een onjuiste correctie te zien zijn door wie dit is uitgevoerd.

Voor de volgorde van de getoonde gegevens in de grid, is overleg gepleegd met de toekomstige gebruiker. In eerste instantie is de getoonde volgorde gebaseerd op hoe deze was in de oude MaDo applicatie. Van de applicatie is op dat moment een prototype gemaakt, welke aan de gebruiker is getoond. Deze heeft toen aangegeven dat gegevens als de verwachte aankomst datum en de positie van een schip niet belangrijk zijn, waardoor deze meer naar achter geschoven mochten worden.

7.4.2 Gegevensverzameling of database

Bij het gegevensoverzicht heeft de gebruiker de mogelijkheid om een tijdsbestek te selecteren waarbinnen de te tonen gegevens moeten vallen. De reden voor deze mogelijkheid is dat de gebruiker op deze wijze de selectie zo kan instellen dat alleen de gegevens verkregen tijdens een bepaalde vaart getoond kunnen worden. Zo is het van invloed op de scheepsgegevens of een schip door ijs of juist langs warmere gebieden vaart. Zo zou de uitlaat temperatuur tijdens een ijsvaart door de omgevingstemperatuur lager kunnen uitvallen.

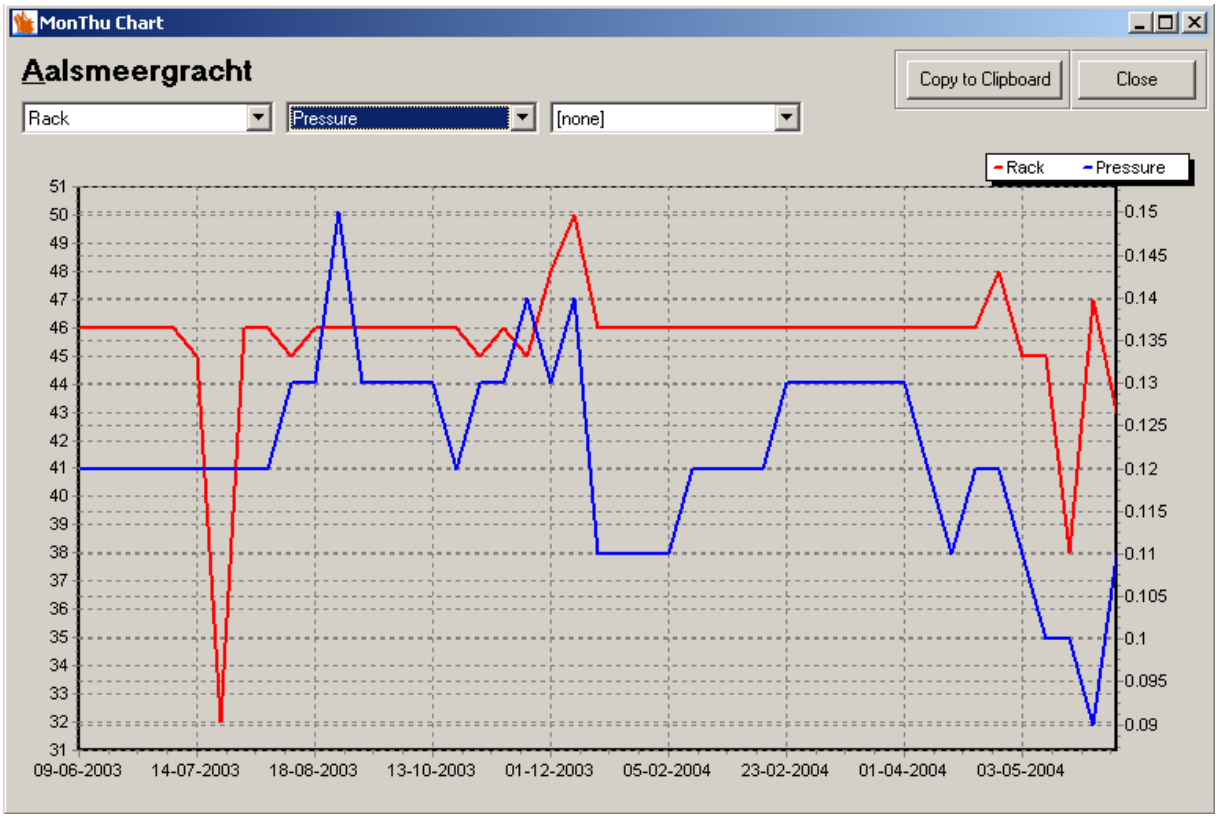
Voor het ophalen en het tonen van de gegevens moest beslist worden hoe dit in verband met de database zou werken. Zo is het inefficiënt om bij elke periode selectie opnieuw de gegevens uit de database op te halen. Hiervoor is binnen Delphi een component aanwezig welke de gegevens tijdelijk lokaal opslaat, een clientdataset. De gebruiker heeft aangegeven dat hij normaalgesproken geen gegevens zal bekijken welke ouder zijn dan één jaar. Hierdoor is besloten om tijdens het voor de eerste keer ophalen van gegevens na de selectie van een schip, dit voor één jaar te doen. Mocht de gebruiker de selectie aanpassen, dan wordt er eerst gecontroleerd of de selectie binnen de huidige aanwezige gegevens in de dataset vallen. Is dit niet het geval, dan worden de gegevens in de dataset aangevuld met de nieuwe benodigde gegevens. Valt de selectie wel binnen de aanwezige gegevens, dan worden deze uit de dataset gefilterd. Met deze manier van werken wordt het netwerk en ook de database zo min mogelijk belast.

7.5 Gegevens in grafiek vorm

Nadat de functionaliteiten van het detailscherm ontwikkeld waren, is begonnen om een andere systeemeis te verwezenlijken:

“De gebruiker moet grafieken op basis van MaDo gegevens kunnen bekijken van een geselecteerd schip.”

In de oude MaDo applicatie konden ook grafieken getoond worden, echter was de keuze van de te tonen gegevens in een grafiek niet door de gebruiker zelf aan te geven. Dit was één van de gewenste opties die de opdrachtgever graag zou willen zien. Binnen Borland Delphi bevindt zich standaard een grafiek component, welke gebruikt zou worden om een grafiek in de applicatie te tonen.



Figuur 13: Grafiek met prestatiegegevens

Verder wordt door middel van drie keuzemogelijkheden de gebruiker vrijgelaten in de keuze van de te tonen gegevens.

Het idee was om de gegevens die in de grafiek getoond zouden worden, binnen hetzelfde tijdsbestek te laten zijn als de selectie in het detailscherm. Hierop volgend leek het ook een goede manier om in plaats van alleen het tijdsbestek van de gegevensverzameling ook de data over te nemen. Hiermee werd wederom voorkomen dat onnodig een connectie naar de database gemaakt zou hoeven worden.

7.6 Ontwikkeling overige functionaliteiten

De ontwikkeling van de overige schermen van de applicatie is telkens in nauwe samenspraak met de opdrachtgever en de hoofdprogrammeur ontworpen en opgebouwd. Om de interactie op een technische manier tussen de gebruiker en het systeem weer te geven, is gebruik gemaakt van XUAN modellen.

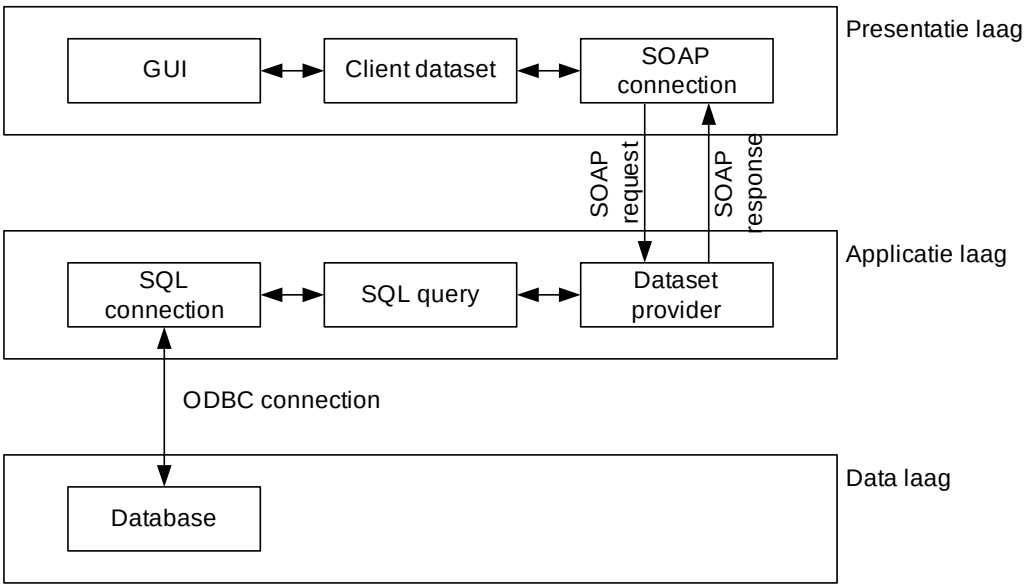
User action	*Control*	*Event*	*System action*
De gebruiker geeft aan een grafiek te willen zien van het huidige geselecteerde schip binnen de geselecteerde tijdsperiode	BtnOpenChart	OnClick	Het systeem opent het grafieken scherm (FrmChartMonThu).
		OnCreate	Het systeem toont de naam van het geselecteerde schip en geeft de gebruiker de mogelijkheid een categorie te selecteren.
De gebruiker selecteert een categorie om te tonen in de grafiek.	CmbBxChartA CmbBxChartB CmbBxChartC	OnClick OnClick OnClick	Het systeem haalt voor de geselecteerde categorie de waarden binnen het huidige tijdsbestek op en toont deze in grafiek vorm in “ChrtDetails”.

Voorbeeld 9: XUAN model in het pilotontwikkelplan

Het bovenstaande XUAN model heeft betrekking op de selectie van gegevens in het grafieken scherm. Zo is duidelijk te zien wat er gebeurt wanneer er op een specifieke knop of ander control geklikt wordt.

7.7 Netwerkcommunicatie van de applicatie

Net als voorgaande pilot wordt bij deze pilot gebruik gemaakt van een gedistribueerde netwerk omgeving met behulp van SOAP.



Figuur 14: Communicatiestructuur bij de applicatie

Op de applicatie server bevinden zich ook functies welke door de client aangesproken kunnen worden. Zo bestaat de systeemeis:

"De gebruiker moet aan kunnen geven dat MaDo gegevens van een schip opnieuw gecontroleerd worden op basis van de grenswaarden."

Omdat er in de loop der tijd vele berichten voor een enkel schip in het systeem kan staan, kan het hercontroleren van deze gegevens enige tijd in beslag nemen. Om het systeem van de gebruiker tijdens deze fase te ontlasten, is er in samenspraak met de hoofdprogrammeur voor gekozen om deze bewerkingen volledig door de applicatie server te laten doen. Het enige wat het client systeem hoeft te doen is het doorgeven van bijvoorbeeld een scheepsnummer aan de server. De client is op de hoogte van de functies op de server doordat dit met WSDL geregeld is.

7.8 Het krijgen van feedback tijdens de ontwikkeling

Omdat door de gekozen iteratiestrategie de systeemeisen per iteratie aangepast of aangevuld kunnen worden, is hier ook de mogelijkheid voor gegeven door middel van prototyping.

Achter het werkstation waar de applicatie op ontwikkeld werd, is de opdrachtgever en de hoofdprogrammeur in het begin van de ontwikkeling van een functionaliteit de werking en opbouw getoond. Met de hierop volgende feedback zijn waar nodig aanpassingen gedaan op het ontwikkelde deel en verder zijn door deze manier van prototyping nieuwe functionele systeemeisen naar boven gekomen. Een voorbeeld hiervan is:

"De hoofdgebruiker moet per serie aan kunnen geven welke gebruikers hiertoe rechten hebben."

Deze systeemeis is gebaseerd op degene waar de hoofdgebruiker de rechten per gebruiker aan zou moeten kunnen geven. Na het zien van deze functionaliteit in ontwikkeling, wilde de opdrachtgever graag ook een omgekeerde vorm van autorisatie.

7.9 Het testen van de functionaliteiten

Tijdens de ontwikkeling is veel gebruikt gemaakt van testmomenten met de opdrachtgever, en de hoofdprogrammeur. De toekomstige gebruiker is minder betrokken geweest bij de ontwikkeling, daar deze aan de hoofdprogrammeur aangegeven heeft wat er verwacht werd en verder niet veel tijd aan de ontwikkeling kwijt wilde.

De testmomenten met de opdrachtgever vonden plaats op afgesproken tijdstippen en betroffen in het begin de volledige tot dan toe ontwikkelde onderdelen. De functionaliteiten hiervan werden gezamenlijk met de opdrachtgever getest en waar nodig werden gewenste aanpassingen aangegeven. Deze gewenste aanpassingen zijn tijdens de test gelijk opgeschreven om ze op een later tijdstip te kunnen aanpassen. Deze werkwijze had verder geen negatieve gevolgen voor de planning aangezien daarin al rekening is gehouden met aanpassingen en iteraties tijdens de ontwikkeling. Enkele voorbeelden van punten die door de opdrachtgever aangedragen werden:

- In het commerciële overzicht alle schepen tonen, ook mogelijkheid tot zien van origineel bericht.
- Snelheid bij commerciële overzicht meer naar voren plaatsen (voor draft).
- Bij autorisatie schermen een omschrijving zetten waar ze voor dienen.

Bij een volgend testmoment werden eerst de punten op de lijst van de vorige test doorgenomen om te zien hoe die punten verwerkt waren, waarna de nieuw ontwikkelde functionaliteiten getest werden.

Na het testen door de opdrachtgever zijn de ontwikkelde delen ook door de hoofdprogrammeur getest. Aangegeven aanpassingen werden hierbij meteen verwerkt, dit in tegenstelling tot het eerst noteren ervan. De reden voor deze manier is dat de hoofdprogrammeur tegelijk op technisch gebied hulp kan bieden door de aanwezige programmeerkennis.

8 Pilot 3: Genereren van gegevens rapporten

Deze pilot hield in dat er met behulp van een reporting tool, rapporten op basis van de aanwezige prestatie gegevens gegenereerd moesten worden. Als eerste wordt er een omschrijving gegeven van de gebruikte tool en de kennismaking hiermee. Hierna zal de daadwerkelijke ontwikkeling beschreven worden.

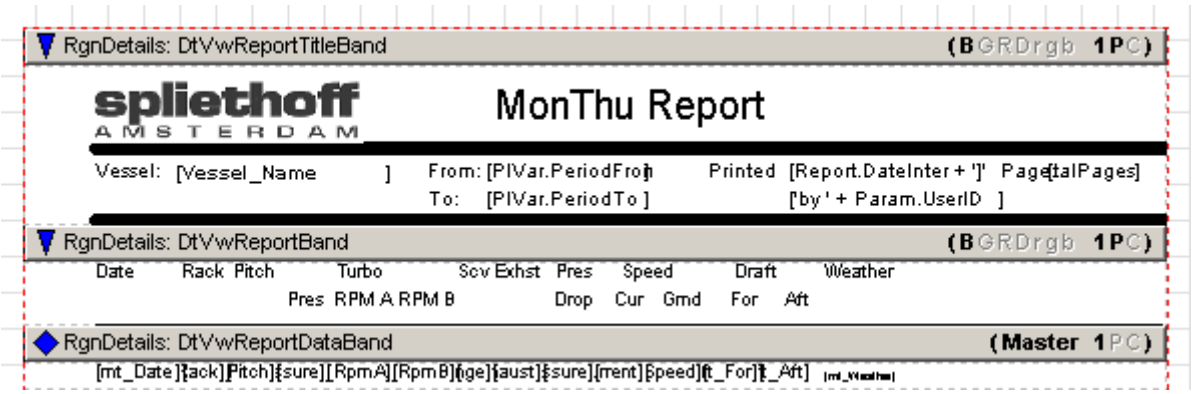
8.1 Nevrona Rave Reports, de reporting tool

Reporting tools worden gebruikt om het genereren van rapporten op basis van gegevens te vergemakkelijken. Het is mogelijk om via programmeertalen volledig via de code een tekst bestand aan te maken, maar wat is er makkelijker dan het visueel te doen? Hiervoor brengen reporting tools de uitkomst, waarvan Nevrona Rave Reports de gebruikte is aangezien deze aanwezig is binnen Spliethoff. Een ander zeer bekende reporting tool is Crystal Reports van Microsoft.

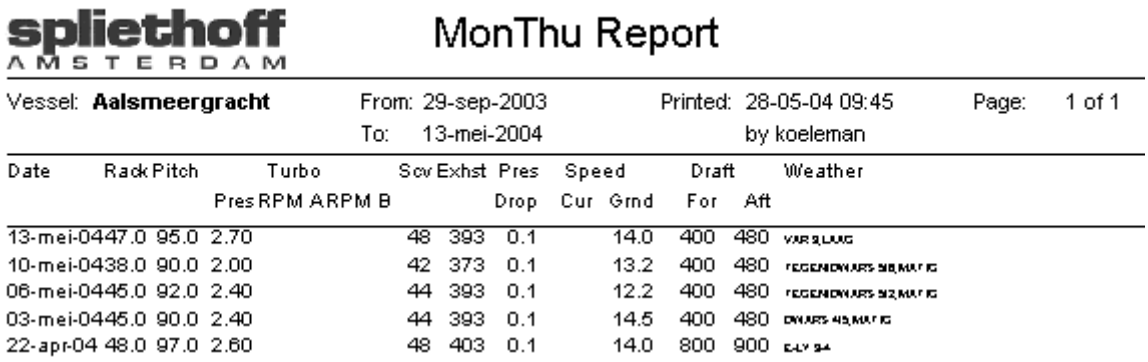
8.1.1 De werking van een reporting tool

Het idee is als volgt, je maakt via een visuele editor een rapport aan welke daarna opgeslagen wordt. Door middel van een aanroep vanuit de gebruikte programmeertaal wordt de instructie gegeven om het rapport op te bouwen en te tonen op het scherm of direct af te laten drukken.
Om ook daadwerkelijk gegevens in het rapport te laten zien, wordt tegelijk met de aanroep voor de opbouw een verwijzing naar een gewenste gegevensverzameling meegestuurd. Tijdens de ontwikkeling van het rapport is door de gebruiker aangegeven waar welke gegevens van de gegevensverzameling moeten komen te staan en in welke vorm. De gebruiker is hier helemaal vrij in, moeten het een groot lettertype zijn of juist een kleine, moet de tekst gewoon in zwart, of toch maar in een kleur?

In Nevrona Rave Reports wordt gewerkt met zogenaamde banden om gegevens in te tonen. Er zijn twee soorten te onderscheiden, te weten de “Band” en de “DataBand”. De “Band” wordt vooral gebruikt om statische gegevens in op te nemen. Denk hierbij vooral aan kop en voetteksten in het rapport. De “DataBand” wordt vooral gebruikt voor het tonen van de gegevens uit de gegevensverzameling. Op de band zelf kunnen echter geen gegevens getoond worden, hier is een ander Rave component voor nodig, de “DataText”. De “DataText” is puur bedoeld voor het tonen van gegevens uit de gegevensverzameling. Zo wordt bij elk “DataText” component op de “DataBand” aangegeven welke data getoond moet worden. Om een indruk te geven over de werking van deze banden en het data component volgen hier twee afbeeldingen.



Figuur 15: Banden in ontwikkelgedeelte



spliethoff
AMSTERDAM

MonThu Report

Vessel: **Aalsmeergracht** From: 29-sep-2003 To: 13-mei-2004 Printed: 28-05-04 09:45 by koeleman Page: 1 of 1

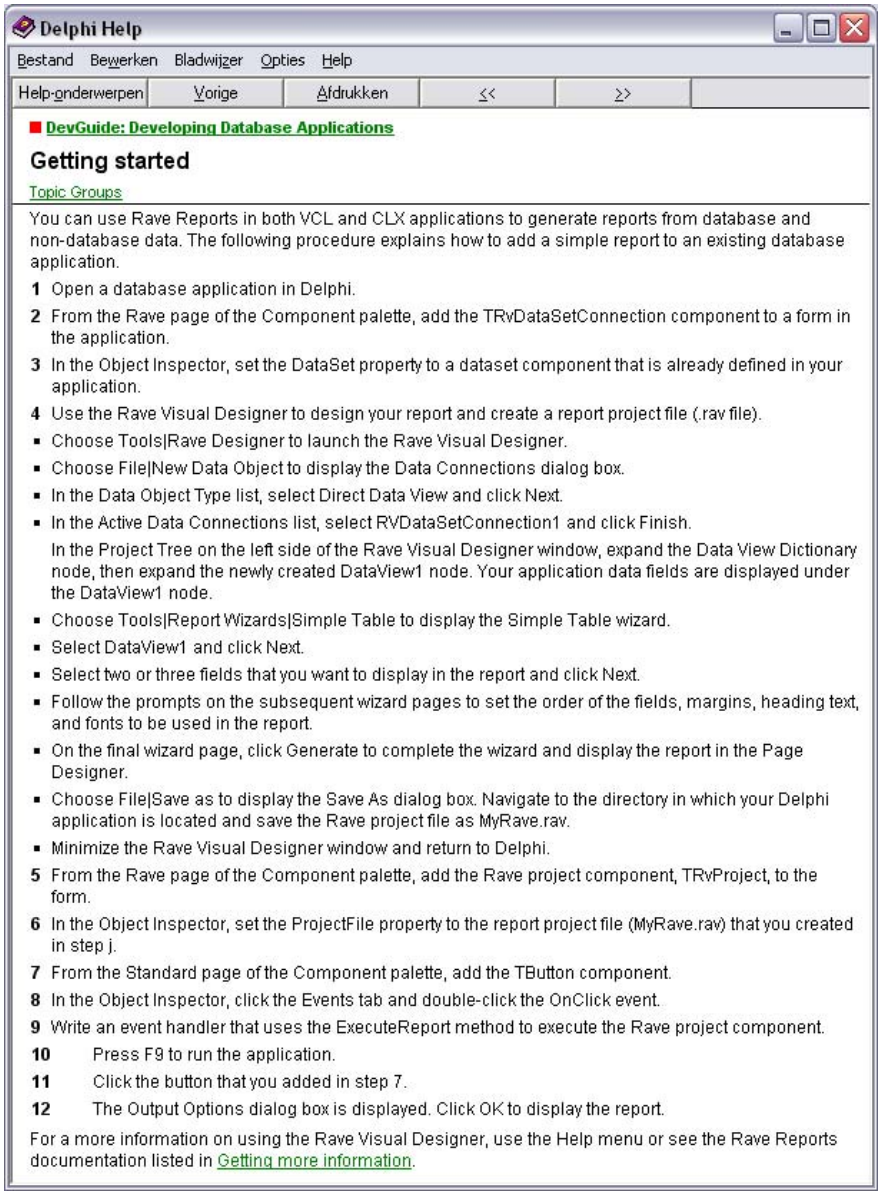
Date	Rack	Pitch	Turbo Pres	Turbo RPM	Turbo ARPM	Turbo B	Scv	Exhst	Pres Drop	Speed Cur	Speed Grnd	Draft For	Draft Aft	Weather
13-mei-04	47.0	95.0	2.70				48	393	0.1		14.0	400	480	VAR
10-mei-04	38.0	90.0	2.00				42	373	0.1		13.2	400	480	REGEN
06-mei-04	45.0	92.0	2.40				44	393	0.1		12.2	400	480	REGEN
03-mei-04	45.0	90.0	2.40				44	393	0.1		14.5	400	480	DWARS
22-apr-04	48.0	97.0	2.60				48	403	0.1		14.0	800	900	ITALY

Figuur 16: Banden in werking

In Figuur 15 worden de banden getoond in het visuele ontwikkelgedeelte, waarin de opmaak bepaald wordt. De bovenste is de “Band” met daaronder de “DataBand”. In Figuur 16 is het resultaat weergegeven nadat met behulp van de gegevensverzameling het rapport opgebouwd is. Zo is te zien dat de “Band” nog dezelfde opmaak heeft als in het ontwikkel gedeelte en dat de “DataBand” herhalend gegevens weergeeft, dus feitelijk het aantal records welke via de gegevensverzameling verkregen zijn.

8.1.2 Kennismaking met Rave Reports

Bij de start van deze pilot was nog nooit met een reporting tool gewerkt zodat eerst een kennismaking plaats moest vinden. Voordat daadwerkelijk met de tool gewerkt ging worden, is eerst bij collega’s op de IT afdeling meegekeken, deze waren op het gelijktijdige moment ook bezig met het maken van enkele rapporten. Na een korte introductie van de collega’s is zelf begonnen om de tool onder de knie te krijgen. Het gestelde doel was een zeer eenvoudig rapport te kunnen genereren met een aanroep en gegevens vanuit Delphi.
Zo is er binnen het Delphi project een nieuw test scherm gemaakt puur voor de kennismaking met Rave Reports. Als eerste is een “getting started” opdracht gevolgd welke in de Rave Reports help staat.



Figuur 17: "Getting started" met Rave

Nadat de “getting started” opdracht volbracht was en een eenvoudig rapport gegenereerd kon worden is het rapport naar eigen inzicht gewijzigd om dingen uit te proberen. Zo is geprobeerd om de aanduiding van pagina nummers en een variabele vanuit Delphi te tonen, uit te voeren. Dit werkte zeer snel zonder enige problemen waardoor besloten werd om te beginnen met de echte gegevens rapporten.

8.2 Te tonen gegevens op de rapporten

Tijdens de definitiefase is met behulp van de systeemeisen bepaald wat voor soort rapporten er gegenereerd moeten worden. Er was vanaf de gebruiker de wens om een rapport te kunnen printen met de technische van een geselecteerd schip. De opdrachtgever vond het een mooie aanvulling als er ook een rapport met grafieken op basis van de gegevens gemaakt kon worden.

8.2.1 Technische gegevens

Voordat er echt met het maken van de rapporten begonnen werd, is eerst uitgedacht welke gegevens er daadwerkelijk op het gegevensrapport getoond moesten worden. In tegenstelling tot het op het scherm tonen van gegevens, heb je op papier minder ruimte. Zo is het voor de technische dienst niet belangrijk wat de huidige positie in coördinaten van een schip is. Gekeken is naar de eerste rij gegevens zoals deze getoond worden in het technische detail scherm. Aangezien de gebruiker daar aangegeven heeft wat hij wel en niet belangrijk vindt, is dat als uitgangspunt genomen. Dit resulteerde erin dat de volgende technische gegevens op het rapport getoond zouden moeten worden:

- Scheepsnaam;
- Datum van bericht;
- Rackstand;
- Schroefstand;
- Turbo druk;
- Turbo A toerental;
- Turbo B toerental;
- Oplaadtemperatuur;
- Uitlaattemperatuur;
- Stroomsnelheid;
- Grondsnelheid;
- Diepgang voor;
- Diepgang achter;
- Weertype.

Bovenstaand zijn puur de technische gegevens, echter moest ook bepaald worden welke algemene gegevens van het rapport zelf weergegeven moesten worden. Om hierachter te komen is hierover gesproken met de opdrachtgever. Aangegeven werd dat in verhouding met de uniformiteit met andere rapporten binnen Spliethoff de volgende gegevens op het rapport moesten komen te staan, waarin in de plaatsing ruimte voor eigen initiatief gelaten is:

- Spliethoff logo;
- Periode waarbinnen de getoonde gegevens vallen;
- Huidige datum en tijd;
- Verantwoordelijke gebruiker;
- Aantal pagina’s en aanduiding ervan.

8.2.2 Drie soorten grafieken

Voor het grafieken rapport geldt dat de gedefinieerde stijl van het gegevens rapport gevolgd moet worden. Echter, welke gegevens in grafiek vorm getoond moesten worden, was nog niet beslist. Om ideeën op te doen is gekeken naar de oude MaDo Access applicatie, hier is ook de mogelijkheid aanwezig om grafieken af te drukken. Hier is het zo dat per variabele een eigen grafiek getoond wordt, om ruimte te besparen is naar voren gebracht om gegevens welke een verband hebben, in één grafiek te tonen.

Met het idee om meerdere gegevens in één grafiek te tonen, is uitgezocht welke hiervoor in aanmerking komen. Eerst is gekeken welk soort gegevens bij elkaar getoond zouden kunnen worden. Deze combinaties zijn opgeschreven en daarmee is overleg gepleegd met de opdrachtgever. Hieruit is gekomen dat er drie grafieken getoond zullen worden met de volgende samenstelling qua te tonen gegevens:

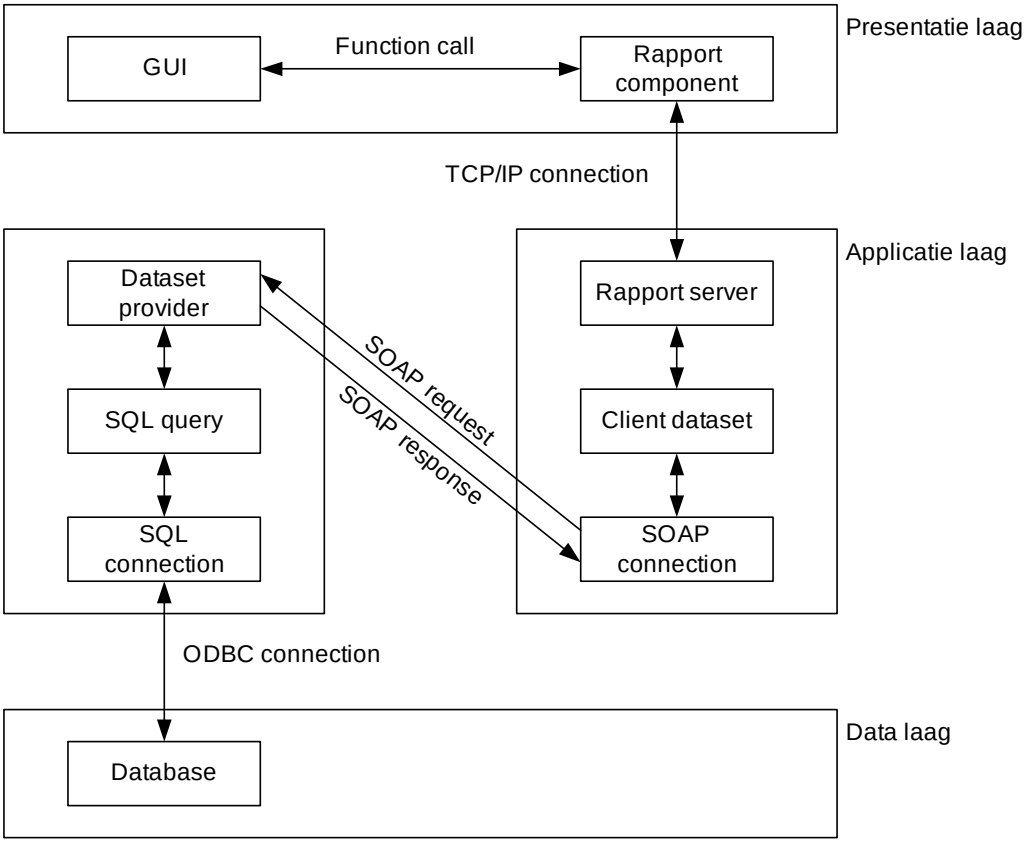
- Grafiek 1: turbo A RPM, turbo B RPM en turbo druk;
- Grafiek 2: rackstand en uitlaattemperatuur;
- Grafiek 3: diepgang voor en diepgang achter.

De opdrachtgever kwam ook met de wens dat de grafieken allen een zelfde tijdsbestek weer zouden geven. Hierbij zouden ook de aanduidingen gelijk moeten liggen, zodat de gebruiker ook het eventuele verband tussen de verschillende grafieken zou kunnen zien.

8.3 Rapporten via de client-server architectuur

Voordat er begonnen is met het ontwikkelen van de rapporten, is eerst bekeken hoe de rapporten in de productie omgeving gegenereerd zullen worden. Om hier achter te komen is gesproken met de hoofdprogrammeur. Net als bij de applicatie, vindt bij het genereren van rapporten de communicatie plaats via een centrale server. Dit is niet alleen fysiek een server, er draait software dat zorgt voor de juiste afhandeling en het aanmaken van de rapporten.

De werking hiervan is als volgt, deze server ontvangt van een client een opdracht om een bepaald rapport te genereren. Dit gebeurt op basis van een functie welke door de client met bepaalde paramaters aangeroepen wordt. Met deze parameters wordt ook de locatie van de applicatie server meegegeven. De rapport server spreekt met de gegevens de webservice aan, welke weer zorgdraagt voor het ophalen van de gegevens uit de database en deze teruggeeft aan de rapport server. Als laatste maakt de rapport server het gewenste rapport fysiek aan op de server zelf. Het aanmaken van het rapport is op dit moment volbracht, echter moet het nog terug gestuurd worden naar de client. Dit gebeurt door middel van het geven van de locatie van het nieuwe rapport op de rapport server. De client opent dan vervolgens de teruggegeven link alsof het lokaal aanwezig is. Om één en ander wat te verduidelijken, is dit weergegeven in Figuur 18.



Figuur 18: Communicatie bij rapport generatie

In het figuur wordt het proces beschreven aan de hand van het 4-tier principe. Binnen Spliethoff zijn de rapport en applicatie server fysiek dezelfde, waarop het rapport en applicatie gedeelte als twee verschillende processen draait. Mocht in de toekomst blijken dat het rapport gedeelte de server erg zwaar belast, dan zal Spliethoff ook fysiek overstappen op de 4-tier architectuur voor dit proces.

8.4 De ontwikkeling van de rapporten

Nu in kaart was gebracht welke rapporten gemaakt moesten worden en op welke wijze, kon met de echte ontwikkeling begonnen worden.
Omdat de rapport server ook in gebruik is in de echte productieomgeving, is aangegeven dat de te maken rapporten in het begin volledig lokaal gemaakt moesten worden. Mocht op de ene of andere manier een rapport verkeerd zijn en een negatieve invloed uitoefenen op de server, dan zou de productieomgeving hier geen last van krijgen.

8.4.1 De gegevensverzameling voor gebruik met de server

De gebruikte rapport server kan technisch maar overweg met één bepaalde SQL aanroep via de SOAP connectie. Omdat in de analyse fase bepaald is welke gegevens getoond zouden moeten worden, is met deze informatie een view op database niveau aangemaakt. Een view is op database gebied geen echte tabel, maar het kan gezien worden als een virtuele tabel, opgebouwd uit gegevens uit meerdere tabellen. Op de applicatie server is hierna een verwijzing aangemaakt om alle gegevens uit de view op te nemen.
Een andere mogelijke oplossing van dit probleem zou zijn door de SQL statement waarmee de view opgebouwd is, gewoon op de applicatie server te implementeren. Het probleem hiervan is dat wanneer er in de toekomst meer gegevens voor de rapporten nodig zijn, de functionaliteit op de applicatie server aangepast moet worden. Door de gekozen oplossing hoeft enkel de view op de database aangepast te worden.

8.4.2 Rapporten met prestatiegegevens

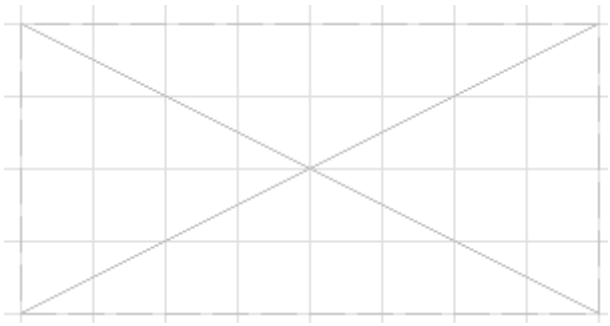
Door de eerdere kennismaking met Rave Reports in het begin van deze fase, was het geen probleem het gegevensrapport te maken. Met behulp van de visuele ontwerpomgeving is de structuur gemaakt waarin de gewenste gegevens getoond kunnen worden na het genereren van het rapport.

Nadat de eerste versie van het rapport lokaal aangemaakt was, is deze door te toekomstige gebruiker gecontroleerd door het rapport voor te leggen. De enige feedback die gekregen was, was dat enkele getoonde gegevens wat dichter bij elkaar gezet mochten worden. Verder was het rapport naar zijn wensen voldoende.
Na het doorvoeren van deze kleine wijzigingen is het rapport gecontroleerd door de opdrachtgever. Een punt wat verbeterd moest worden, was dat wanneer met een perforator gaten in het rapport gezet werden, deze door de getoonde gegevens heen zouden komen. Om dit op te lossen is het lettertype dat op het rapport gebruikt werd verkleind en is een aantal gegevens op het rapport naar rechts verplaatst. Een tweede verzoek van de opdrachtgever was de notatie van de datum, deze stond in het “dd-mm-jjjj” formaat. Om onduidelijkheden in de maand te voorkomen is deze notatie gewijzigd naar het “dd-mmm-jjjj” formaat.
Hiermee was het gegevensrapport naar de wensen van de opdrachtgever en de gebruiker.

8.4.3 De grafieken naar Rave

Daar waar de ontwikkeling van de gegevensrapporten zonder enige problemen verlopen is, zijn er wel problemen bij het ontwikkelen van het grafieken rapport naar boven gekomen. Het

bleek dat het net als bij het plaatsen van een “DataText” component, niet mogelijk was om via de visuele ontwerpomgeving een grafiek in een rapport te plaatsen.
Omdat de grafiek zoals deze bij een voorgaande pilot ontwikkeld is, naar wens was van de opdrachtgever en gebruiker, is onderzocht of het mogelijk was om zo’n Delphi component in Rave te krijgen. Het idee was om dan met Delphi de benodigde grafieken te laten genereren en deze dan op een bepaalde manier in het rapport te plaatsen.
Op de website van Nevrona is voordat er daadwerkelijk met de ideeën aan de slag gegaan werd, eerst gekeken of er een andere oplossing zou zijn. Het leuke van het geheel was dat zelfs op de officiële website van de leverancier aangegeven werd dat het bedachte idee ook de juiste oplossing was. Wanneer een grafiek binnen Delphi ontwikkeld zou zijn, moet deze via een CustomConnection naar Rave verzonden worden. Rave herkend de aangeboden gegevens van de CustomConnection net als een normale gegevensverzameling. Hierdoor kan aangegeven worden dat de waarde in de gegevensverzameling, getoond moet worden in een data component. Zo wordt bij het gegevens rapport een “DataText” gebruikt, hier wordt een “Meta” component gebruikt.



Figuur 19: Een Meta component in Rave

Een “Meta” component moet gezien worden als een aanduiding waar bijvoorbeeld een afbeelding in geplaatst kan worden, welke via een extern programma aan Rave aangeboden wordt.

Met deze oplossing is het probleem van de grafieken in Rave nog niet opgelost. De gebruikte rapport server kent hier nog geen functie voor. Daarom is de code die lokaal gebruikt is voor het genereren van de rapporten bijna volledig overgezet naar de server zelf. Door een eenvoudige aanroep maakt de server de grafieken aan en plaatst deze in de rapporten.

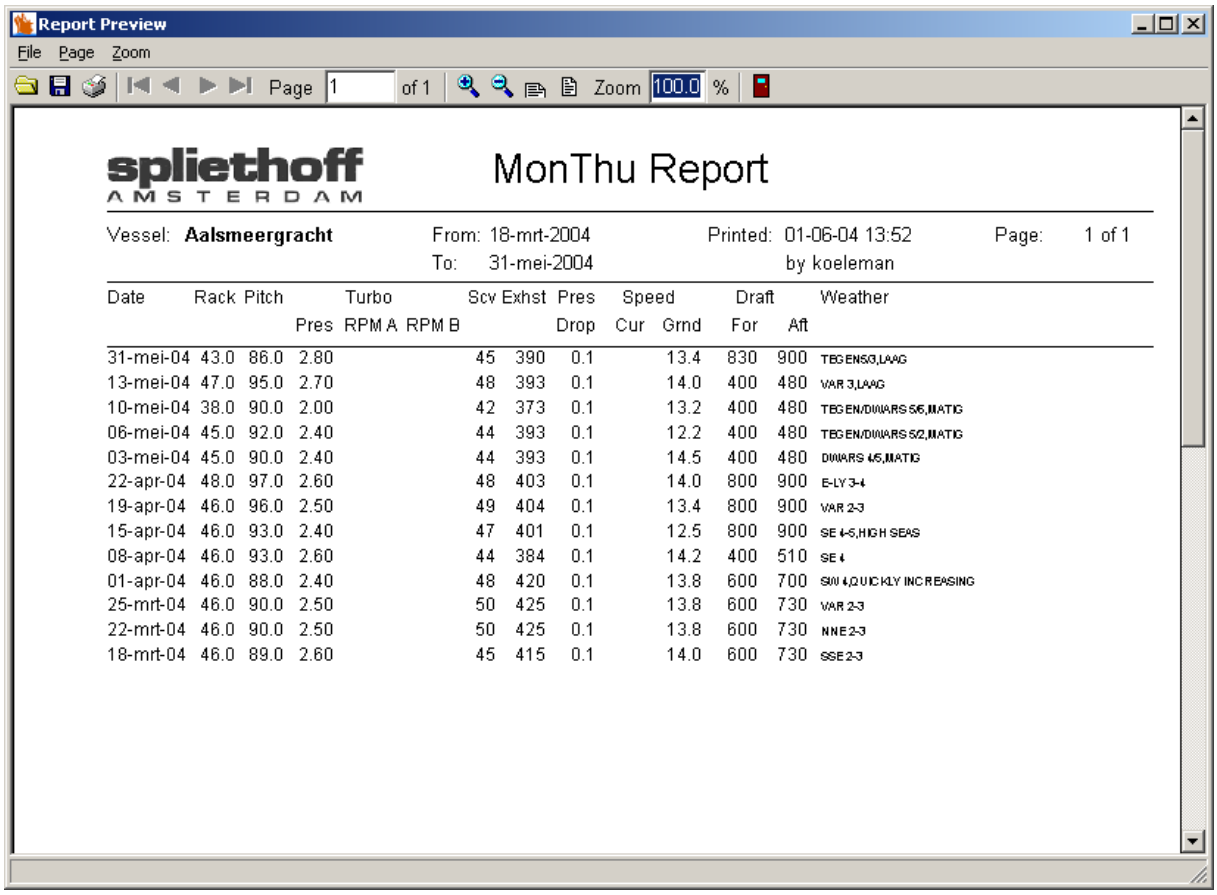
8.5 Samenvoeging met de applicatie

Aangezien het rapporten gedeelte een aanvulling is op de in pilot 2 gemaakte applicatie, zijn deze na het voltooien van de ontwikkeling samengevoegd. Dit was geen moeilijk gedeelte aangezien enkel de aanroep naar de rapporten op een ander deel in de applicatie gezet moest worden. Na deze samenvoeging zijn de rapporten ook direct overgezet naar de rapport server. Meteen bleken de rapporten goed te functioneren zonder enige verdere aanpassingen.

8.6 Resultaat van de pilot

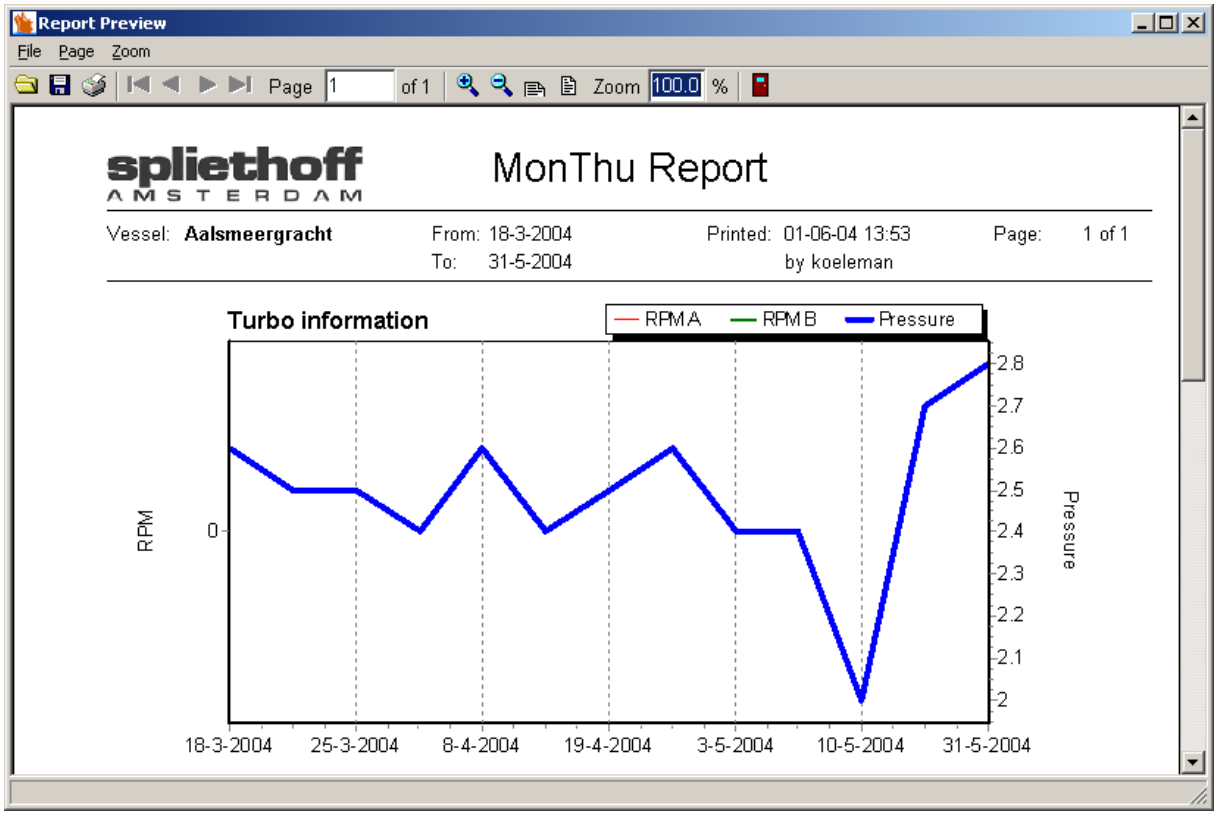
Het resultaat van deze rapporten pilot is dat het nu mogelijk is om de gegevens die digitaal waren, ook op een nette manier te kunnen printen. Hiermee kunnen de gegevens meegenomen worden bij het controleren van de prestatie van een schip.

Hieronder is het uiteindelijke resultaat van de rapporten pilot weergegeven. Bij de grafiekrapport is slechts één grafiek zichtbaar, dit is gedaan om de schermen duidelijk te houden.



Date	Rack	Pitch	Turbo Pres	Turbo RPM A	Turbo RPM B	Scv	Exhst Pres	Exhst Drop	Speed Cur	Speed Grnd	Draft For	Draft Aft	Weather
31-mei-04	43.0	86.0	2.80			45	390	0.1		13.4	830	900	TEGEN SG,LAAG
13-mei-04	47.0	95.0	2.70			48	393	0.1		14.0	400	480	VAR 3,LAAG
10-mei-04	38.0	90.0	2.00			42	373	0.1		13.2	400	480	TEGEN/DWARS S6,MATIG
06-mei-04	45.0	92.0	2.40			44	393	0.1		12.2	400	480	TEGEN/DWARS S2,MATIG
03-mei-04	45.0	90.0	2.40			44	393	0.1		14.5	400	480	DWARS 4/5,MATIG
22-apr-04	48.0	97.0	2.60			48	403	0.1		14.0	800	900	E-LY 3-4
19-apr-04	46.0	96.0	2.50			49	404	0.1		13.4	800	900	VAR 2-3
15-apr-04	46.0	93.0	2.40			47	401	0.1		12.5	800	900	SE 4-5,HIGH SEAS
08-apr-04	46.0	93.0	2.60			44	384	0.1		14.2	400	510	SE 4
01-apr-04	46.0	88.0	2.40			48	420	0.1		13.8	600	700	SW 4,QUICKLY INCREASING
25-mrt-04	46.0	90.0	2.50			50	425	0.1		13.8	600	730	VAR 2-3
22-mrt-04	46.0	90.0	2.50			50	425	0.1		13.8	600	730	NNE 2-3
18-mrt-04	46.0	89.0	2.60			45	415	0.1		14.0	600	730	SSE 2-3

Figuur 20: Gegevensrapport



Figuur 21: Grafiekenrapport

9 Invoering van de pilots in de organisatie

In dit hoofdstuk wordt het proces beschreven welke aan de orde was bij de daadwerkelijke invoering van de ontwikkelde pilots in de organisatie.

9.1 Voorbereidingen voor de invoering

Door de gekozen iteratiestrategie vindt de invoering van alle pilots in één keer plaats. Om dit goed te laten verlopen is eerst een planning opgesteld over hoe de invoering plaats moet vinden. Als eerste zou de analyse functie in de productieomgeving ingezet moeten worden waarna de applicatie operationeel gemaakt zou worden.

Nadat de analysemodule van pilot 1 ontwikkeld was, was ervoor gekozen om alle oude MaDo records uit de oude database te converteren naar de nieuwe database. Nadat die conversie voltooid was, is een backup gemaakt van de verwerkte gegevens. Nu de invoering van de applicatie voor handen was, moesten deze gegevens in de productie database gezet worden. Na dit met de hoofdprogrammeur besproken te hebben, bleek het een beter idee om in plaats van alle gegevens over te zetten naar de productie tabel, deze te verwijderen en de backup tabel als productie tabel in te stellen. Aangezien het verschil tussen beide feitelijk alleen de naam was (OPS_MONTHU en OPS_MONTHU_backup), is dit veel efficiënter dan alle gegevens ook daadwerkelijk over te zetten. De nieuwere MaDo gegevens zouden, nu de analysemodule in productie zou komen, makkelijk opnieuw aangeboden kunnen worden.

Een andere voorbereiding voor de invoering was dat de ontwikkelde rapportgeneratie functie voor de rapport server bij de productie rapport server geïnstalleerd moest worden. Deze handelingen werden uitgevoerd door de hoofdprogrammeur aangezien hij verantwoordelijk is voor de rapport server.

9.2 De analyse module in bedrijf

Voor de invoering van de analyse module zijn geen grote handelingen benodigd geweest. Het enige wat hoefde te gebeuren was het compileren van de code ervan in Borland Delphi. De DLL die hieruit voortkwam is overhandigd aan de ontwikkelaar van de message scheduler en deze heeft binnen de message scheduler kenbaar gemaakt welke berichten naar de analyse module verzonden moesten worden.
Via een andere ontwikkelde tool door de ontwikkelaar, de supervisor, kunnen de log berichten van de analyse module op een client systeem bekeken worden. Hiermee kon op de eerste donderdag dat de analyse module in werking was, gekeken worden hoe de aangeleverde berichten behandeld waren. Alle berichten werden goed herkend en daarna geanalyseerd opgeslagen in de database. Zo is de invoering van de analyse module probleemloos verlopen.

9.3 De MaDo applicatie bij de gebruiker

De invoering van de applicatie is ook zonder enige problemen verlopen. Voordat deze echt binnen de Technische Dienst in gebruik genomen kon worden, moesten eerst alle ontwikkelde functionaliteiten met Fleetcom geïntegreerd worden. Het grote voordeel hiervan is dat nu de systeembeheerder van Spliethoff niet op alle werkstations waar gebruik gemaakt mag worden van de MaDo applicatie, een versie hoeft te installeren. De hoofdprogrammeur integreert het met Fleetcom waarna via een update alle werkstations meteen de nieuwste versie hebben. Deze update vind plaats wanneer het systeem van een medewerker van Spliethoff opstart. Zo wordt er gecontroleerd of de versie van Fleetcom welke op het systeem staat anders is dan die op de server. Is dit het geval, dan wordt de versie vanaf de server naar het werkstation gekopieerd.

De hoofdprogrammeur doet de integratie door alle schermen van de applicatie vanuit het ontwikkelproject in het Fleetcom Delphi project te plaatsen. Hierna kan de hoofdprogrammeur via een autorisatie tool aangeven, welke medewerkers binnen Spliethoff het MaDo onderdeel in Fleetcom te zien en dus ook te gebruiken krijgen.

Toen de integratie voltooid was, is samen met de hoofdprogrammeur naar de toekomstige gebruiker gegaan voor een korte uitleg. Zo is laten zien hoe technische gegevens te wijzigen zijn en hoe schepen te beheren zijn qua de serie tot welke ze behoren. De gebruiker was zeer tevreden met het resultaat en vond de mogelijkheid waarmee de datum van het laatst ontvangen bericht getoond kon worden ook zeer handig.
Nadat de gebruiker de applicatie getest had heeft deze aangegeven welke overige gebruikers op welke scheepsseries rechten moesten hebben. De hoofdprogrammeur heeft deze rechten ingesteld waarna de applicatie binnen Fleetcom operationeel was.

9.4 De rapporten operationeel

De rapporten zijn tegelijk met de applicatie als onderdeel ingevoerd. Echter bleek er bij de generatie van de rapporten wel een probleem te zijn. Hoewel de grafiek functionaliteiten al wel bij de productie rapport server geïmplementeerd waren, bleek het genereren van de betreffende rapporten niet goed te lukken. Het probleem was dat de fysieke server waar de productie rapport server op draait een verouderde versie van een bepaald benodigd bestand had. De oorzaak hiervan was dat op reeds alle systemen waarmee de grafiek rapporten gegenereerd werden, ook Borland Delphi stond. Door de installatie hiervan werd een nieuwe versie van het benodigde bestand geïnstalleerd.
Dit probleem is door de hoofdprogrammeur opgelost door het nieuwe bestand op de server te plaatsen en deze te herstarten. Hierna konden de grafiek rapporten net als de gegevens rapporten goed gegenereerd en geprint worden.

10 Evaluatie

In dit hoofdstuk wordt het proces en de producten met betrekking tot het afstudeertraject geëvalueerd. Er vindt een evaluatie plaats per opgeleverd product, gevolgd door de evaluatie over het gehele proces.

10.1 Product evaluatie

Aan het einde van het afstudeertraject zijn de volgende producten met betrekking tot het afstudeertraject opgeleverd:

- Plan van aanpak;
- Definitiestudie;
- Pilotontwikkelplannen;
- Invoeringsrapport.

Deze producten zullen in de volgende paragrafen aan bod komen.

10.1.1 Plan van aanpak

Tijdens het opstellen van het plan van aanpak is duidelijk in kaart gebracht wat er tijdens het traject moest gebeuren en wat er nodig was. De planning is tijdens het traject vaak geraadpleegd, en waar nodig bijgesteld. Veel bijstelling is overigens niet nodig geweest, zo zijn later enkele testmomenten toegevoegd. Dat weinig bijstellingen benodigd zijn geweest is ook te verklaren uit het feit dat de volledige ontwikkeling onderverdeeld is, maar dit niet gedetailleerd is gespecificeerd in de planning. De afzonderlijke planningen binnen de pilotontwikkeling zijn wel op de nodige punten aangepast, maar daar wordt in het betreffende paragraaf verder ingegaan.

10.1.2 Definitiestudie

De definitiestudie is tijdens het project op vele momenten aangepast door de gekozen iteratiestrategie. Dit gebeurde vooral op het gebied van de systeemeisen, hierop hebben tijdens de ontwikkeling aanvullingen maar ook verwijderingen plaatsgevonden.
Zo was tijdens de beginfase in het traject het een wens van de opdrachtgever dat ook voorspellingen gedaan konden worden op basis van de aanwezige gegevens. Hiervoor was een aparte pilot gedefinieerd, maar het bleek dat deze pilot niet haalbaar zou zijn. Wel kon met behulp van statistiek de lineaire regressie en verwachtte waarden berekend worden, maar de echte toekomstige voorspellingen waren niet mogelijk. Dit bleek nadat met de hoofdprogrammeur een rekenmodel bekeken is, waarmee dit wel mogelijk zou moeten zijn. Er kwam naar voren dat er in de MaDo berichten te weinig gegevens aanwezig waren. Zo is bij dit soort berekeningen ook van belang wat voor soort benzine en bijvoorbeeld olie gebruikt word op een schip.
Het systeemconcept is een belangrijk deel van de definitiestudie, hierop worden de te ontwikkelen pilots gebaseerd en moeten dus duidelijk zijn. Om dit voor de opdrachtgever en hoofdprogrammeur ook te houden, zijn hiervoor use-cases gebruikt.

10.1.3 Pilotontwikkelplannen

In deze rapporten is de systeemdocumentatie opgenomen van de ontwikkelde pilots. Er is gekozen om dit te doen in de vorm van schermspecificaties in combinatie met XUAN modellen. Hiermee kan duidelijk de acties van de gebruiker weergegeven worden. In situaties waar geen schermen aan de orde zijn, zijn onder andere flowcharts en een toestandsdiagram gebruikt. Er zijn geen delen van de programmeercode opgenomen, omdat dit met behulp van de documentatie snel op te zoeken is binnen het Delphi project. Dit is verder ook aangegeven door de hoofdprogrammeur, waardoor het rapport niet gedetailleerder hoefde te zijn.

10.1.4 Invoeringsrapport

In het invoeringsrapport is beschreven wat gedaan moet worden om de applicatie operationeel te maken binnen de organisatie. Echt ingewikkelde procedures staan hier niet in aangezien de invoering eenvoudig via Delphi te doen is. Het MaDo project is zo opgesteld dat alleen de schermen, en dus automatisch de code, gekopieerd hoeven te worden naar een ander project, in dit geval van Fleetcom. Zo ook de organisatorische gevolgen, deze zullen niet groot zijn, aangezien het in eerste instantie een applicatie is voor de Technische Dienst. Het zal dienen als een ondersteunend hulpmiddel en niet iets gaan vervangen.

10.1.5 De applicatie en bijbehorende onderdelen

De opdrachtgever en gebruiker zijn tevreden met de ontwikkelde onderdelen tijdens het traject. Zo werkt de analyse module zonder problemen in productie en worden de berichten goed geanalyseerd en verwerkt. De applicatie is operationeel bij de Technische Dienst, waar de gebruiker er zeer tevreden mee is. Zo hebben andere werknemers ook al aangegeven graag gebruik te willen maken van de ontwikkelde applicatie, wat betekent dat deze er ook heil in zien. Zo is het technische gedeelte via Fleetcom alleen bereikbaar voor de Technische Dienst, de commerciële gegevens zijn voor alle medewerkers voor Fleetcom bereikbaar. Hierop zijn ook al positieve berichten gekomen.

10.2 Proces evaluatie

De procesgang tijdens het afstudeertraject is positief en zonder problemen verlopen. De omgang met de collega's van de IT afdeling was goed en wanneer ergens hulp voor nodig was, is dat ook altijd gekregen en andersom ook gegeven.

Dat de communicatie met de uiteindelijke gebruiker niet hoog is geweest heeft ook geen negatieve effecten op het proces gehad. De hoofdprogrammeur en de opdrachtgever waren beide goed op de hoogte van de wensen en hadden ook hun eigen inbreng hierin. Dit resulteerde erin dat de aanlevering van benodigde informatie te allen tijde goed was.

De planning is tijdens het proces goed gevolgd en mede door het vervallen van een vierde pilot ook gehaald. Wanneer de vierde pilot voor het detecteren van toekomstige afwijkingen wel ontwikkeld zou kunnen worden, zou naar alle waarschijnlijkheid de planning niet gehaald zijn. In dit geval had er waarschijnlijk besloten moeten worden of enkele features van de applicatie achterwege gelaten zouden moeten worden of dat voor de vierde pilot alleen voorbereidend onderzoek gedaan zou worden. In het huidige proces is de applicatie wel ontwikkeld en is alleen uitgevonden dat er voor de toekomstige voorspelling meerdere gegevens nodig zijn, welke dit zijn is verder niet uitgezocht.

De gebruikte IAD strategie heeft ook duidelijk gewerkt in verband met de steeds wijzigende systeemeisen. Doordat er interactief en iteratief ontwikkeld werd, kregen de opdrachtgever

en hoofdprogrammeur bij het zien van pilotdelen in vele gevallen nieuwe ideeën wat in nieuwe systeemeisen zou kunnen resulteren.

Het gebruik van de vooraf gedefinieerde technieken is ook goed verlopen. Met behulp van het UML onderdeel use-cases konden de systeemeisen goed omgezet worden naar leesbare beschrijvingen van de gewenste functionaliteiten. Meer technische aspecten konden met bijvoorbeeld een flowchart eerst goed uitgedacht en getekend worden voordat met de daadwerkelijke ontwikkeling begonnen werd.

Wat niet gelukt is, is het ontwikkelen van een pilot voor de detectie van toekomstige afwijkingen. Met behulp van statistiek zijn hier wel berekeningen op uitgevoerd, waarna het leek dat het zou lukken. Echter werd door de hoofdprogrammeur aangegeven dat er gewoonweg te weinig gegevens aanwezig zijn om dit echt te berekenen. Door de technische achtergrond die hij heeft en oud lesmateriaal heeft hij kunnen aangeven waarom het niet zou lukken. Om hier verder niet veel tijd aan te verliezen is besloten om het onderdeel te laten voor wat het is. De opdrachtgever is wel van plan om dit in de toekomst als eventueel losstaand project op te zetten.

De begeleiding is tijdens het traject vanuit het bedrijf goed geweest. De opdrachtgever toonde duidelijk interesse in het project en wilde graag weten wat de status op bepaalde momenten was. Ook heeft hij mij in de gelegenheid gesteld voor het werken aan dit eindverslag tijdens werktijd, waarvoor dank.
De technische begeleiding door de hoofdprogrammeur was van goed niveau. Wanneer er een vraag was met betrekking tot het programmeren of de netwerkcommunicatie tussen bepaalde modulen, werd dit toegelicht. Leuk om te zien was dat de hoofdprogrammeur ook op punten van mij en andere afstudeerders geleerd heeft. Wanneer hij bij bepaalde situaties een probleem had, informeerde hij altijd of er toevallig al een oplossing voor was.

Afkortingen

DBMS	Database Management Systeem
IAD	Iterative Application Development
MaDo	Maandag/Donderdag
OBA	Orders van Blijvende Aard
SOAP	Simple Object Access Protocol
UML	Unified Modeling Language
WSDL	Web Services Description Language
XML	eXtensible Markup Language
XP	eXtreme Programming
XUAN	eXtended User Action Notation

Figuurlijst

Literatuurlijst

Literatuur:

R.J.H. Tolido
IAD Het evolutionair ontwikkelen van informatiesystemen
Academic Service
1e druk 1996

J. Warmer
Praktisch UML
Addison-Wesley
1999

Prof. Dr. J.A. Vandenbulcke
Databasesystemen voor de praktijk
Kluwer
6e druk 1997

I. Sommerville
Software Engineering
Addison-Wesley
6e druk 2001

Spliethoff BV
OBA Booklet
Spliethoff Amsterdam
2001

W.J. de Schipper
Syllabus Programmeren 1
Hogeschool InHolland Diemen
3e editie 2001

Internet:

Borland: http://www.borland.com

Spliethoff BV: http://www.spliethoff.com

World Wide Web Consortium: http://www.w3.org

Externe bijlagen

De volgende documenten zijn als externe bijlage meegeleverd met dit document:

- Plan van aanpak
- Definitiestudie
- Pilotontwikkelplannen
- Invoeringsrapport