
Stageverslag



HSB-Bus-Analyser

Rink Springer

Versie 1.2.1
7 juni 2004

**STAGEVERSLAG VOOR FONTYS HOGESCHOOL
INFORMATICA TWEEDE STAGE**

Gegevens Student:

Naam: Springer, R.P.W. (Rink)
Studentnummer: 2016014
Opleiding: Hogere Informatica
Voltijd

Stageperiode: 02/02/2004 - 18/06/2004

Gegevens Bedrijf:

Naam bedrijf: Delem B.V.
Afdeling: Development
Plaats: Eindhoven, Nederland
Naam bedrijfsbegeleider: M. Scholte

Gegevens Docentbegeleider:

Naam docentbegeleider: J.B.H.M. van Heumen

Gegevens verslag:

Titel Stageverslag: HSB Bus Analyzer
Datum uitgifte stageverslag: 07/06/2004

Getekend voor gezien door bedrijfsbegeleider:

Datum:

De bedrijfsbegeleider,

Voorwoord

Dit stageverslag beschrijft mijn 2^e stage bij Delem BV te Eindhoven en is zo informatief mogelijk opgezet.

Voor mensen die niet bekend zijn met de HSB bus of met de opdracht in het algemeen is het raadzaam om eerst de achtergrondinformatie op pagina 25 en de projectbeschrijving op op pagina 26 door te nemen.

De hoofdzaak van deze opdracht was het ontwerp en de implementatie van een analyse- en kwalificatieprogramma voor modules op de Delem HSB bus. De uiteindelijke opdracht is na te lezen in hoofdstuk 2 op pagina 7. De technische kant van de geproduceerde applicatie zal dan ook niet uitgebreid beschreven worden in dit verslag.

Dit verslag is bestemd voor iedereen die geïnteresseerd is in het verloop van mijn stage, wat vooral van toepassing zal zijn bij mijn bedrijfsbegeleider, de heer Scholte en mijn docentbegeleider, de heer van Heumen.

Ik wil de volgende mensen bedanken die mij hebben geholpen tot het succesvol afronden van mijn stage: de heren Marcel Scholte, Bas Kluiters, Bart Meeuwissen en Eddie Draaisma van Delem en de heer Hans van Heumen en mevrouw H el ene Geldof van Fontys Hogeschool Informatica. Tenslotte zou ik mijn (voormalig) mede-studenten Cliff Albert, Tom Broumels, Bart Cortooms en Thijs Hodiamont willen bedanken voor het delen van hun stageverslagen, die de vormgeving en inhoud van dit verslag zeer positief beïnvloed hebben en alle mensen die het stageverslag gelezen hebben en commentaar erop geleverd hebben.

HOOFDSTUK 0. VOORWOORD

Inhoudsopgave

Voorwoord	iii
Samenvatting	vii
Summary	ix
Verklarende woordenlijst	xi
1 Inleiding	1
2 Proces	3
2.1 Projectmanagement	3
2.2 Besprekingen	4
2.3 Initiële Planning	5
2.4 Definitieve Resultaat	6
3 De opdracht	7
4 Onderzoek	9
4.1 CAN Hardware	9
4.2 Definitie	10
4.2.1 Overzicht	11
4.3 Uitbreidbaarheid	11
5 Implementatie	13
5.1 Ontwerp	14
5.2 Documentatie	16
6 Conclusie	17
7 Bronvermelding	19
A Plan van Aanpak	21
A.1 Inleiding	22
A.2 Delem	22

INHOUDSOPGAVE

A.2.1	Afdeling Development	22
A.2.2	Organigram	24
A.3	Achtergrond	25
A.3.1	De pers	25
A.3.2	De modules	25
A.4	Het project	26
A.4.1	Informatie vooraf	26
A.4.2	Organisatie	26
A.4.3	Probleemstelling	26
A.4.4	Resultaat	27
A.4.5	Randvoorwaarden	27
A.4.6	Risico's	27
A.4.7	Fasering	27
A.4.8	Hulpprogramma's	29
A.4.9	Analyses	30
A.5	Planning	31
A.5.1	Overzicht	32
A.5.2	Toelichting	33
A.6	Beheersaspecten	33
A.6.1	Geld	33
A.6.2	Organisatie	34
A.6.3	Kwaliteit	34
A.6.4	Informatie	34
A.6.5	Tijd	34
A.7	Conclusie / Aanbevelingen	34
B	Communicatieplan	35
B.1	Betrokkenen	35
B.1.1	Student	35
B.1.2	Bedrijf	35
B.1.3	Fontys Hogescholen Informatica	35
B.2	Algemeen	36
B.3	Te ontvangen documenten door docentbegeleider	36
B.4	Te ontvangen documenten door stagiair	36
C	Gebruikte software	37

Samenvatting

Dit verslag biedt een overzicht van het verloop van mijn werkzaamheden tijdens mijn 2^e stage te Delem BV. Het verslag beschrijft het projectproces, de onderzoeken, de implementatie en eindigt in een aantal conclusies die het generale verloop van de stage samenvatten.

De lezer van dit verslag wordt verwacht bekend te zijn met HSB modules. Mocht deze kennis ontbreken, dan is het verstandig eerst de achtergrondinformatie op pagina 25 en de projectomschrijving op pagina 26 te lezen.

Het project om een HSB Analyzer te maken is tot stand gekomen om aan de steeds hogere kwaliteitseisen te kunnen voldoen. Aan de hand van terugkoppelingen met de ontwikkelaars is er uiteindelijk een programma geheel op maat voor het bedrijf gemaakt.

Na afloop van deze stage, kan er gezegd worden dat ik uitgebreide kennis van XML opgedaan heb. Deze technologie is uitvoerig in deze stage gebruikt, zoals pagina 10 verder zal toelichten. Verder heb ik ook een goed inzicht gekregen in de algemene gang van zaken bij de firma Delem met betrekking tot projectmanagement en kwaliteitscontrole.

De stage is zeer geslaagd; Delem is tevreden met het opgeleverde programma, waarmee tijdens de ontwikkeling ervan al een aantal zaken aan het licht gekomen is met betrekking tot de communicatie tussen de besturing en de HSB modules.

HOOFDSTUK 0. SAMENVATTING

Summary

This report provides an in-depth view of my second internship at Delem. The result is a program, which I expect to be very useful for Delem. The intention was to create this program as extensible and maintainable as possible. This result has fully been accomplished.

The reader of this report is expected to be familiar with HSB modules. It is advisable to read the background information on page 25 as well as the project description on page 26 if this knowledge is absent.

During this internship, I've spent about one day a week on other activities. A good example of this is testing an existing application VBend¹, which would soon be released. During this activity, I was fully integrated within the project team itself and this experience has given me a pretty good indication on 'normal' projects at Delem. The usage of PR and CR²'s was also greatly illustrated here.

I've also worked quite a bit with XML, which could prove useful in the future. Furthermore, my Windows programming wasn't quite optimal (I hardly use Windows at home), so I learned quite a bit there as well.

Delem has given me quite some freedom while carrying out this internship, which I attribute to the fact that I've been employed at Delem for over 2 years now. However, I believe this internship has given me the opportunity to learn the ins and outs of projectmanagement at Delem, as well as the company itself.

Finally, I believe this has been a very good internship. The project has been well documented and the program is quite functional. It has already been used to find several issues, which may be addressed in the future. A good result, which my supervisor at Delem confirmed.

¹Virtual Bend, application which is capable of determining and simulating an optimal bending sequence testing an existing bend sequence

²Problem Report/Change Request, refer to page 30 for more information

HOOFDSTUK 0. SUMMARY

Verklarende woordenlijst

API	Application Programming Interface, bibliotheekfuncties bedoeld voor communicatie met systeemfuncties.
Besturing	Onderdeel van de drukpers dat de aansturing van de modulen regelt.
CAN	Controller Area Network, origineel door Siemens ontwikkelde bus voor betrouwbare datacommunicatie.
CCB	Change Control Board, zie pagina 30 voor meer informatie.
CR	Change Request, verzoek om iets in de software te implementeren/veranderen.
CVS	Concurrent Version System, uitbreiding op RCS.
Deadlock	Toestand waarin twee of meer stukken code op de zelfde hulpbron wachten, zodat ze elkaar blokkeren.
HSB	Hoge Snelheid Bus, zie CAN.
L ^A T _E X	Een typesetting systeem, waarmee boeken en dergelijke gemaakt kunnen worden door middel van een markup taal
MFC	Microsoft Foundation Classes, API van Microsoft om in C++ (redelijk) eenvoudig Windows programma's te maken.
Module	Regelt de aansturing van (onder andere) motoren. Zie pagina 25 voor meer informatie.
Milestone	Mijlpaal, periode in de fasering. Zie pagina 27 voor meer informatie.
Mutex	Afkorting van mutual exclusion, zorgt ervoor dat een hulpbron niet door meerdere stukken code tegelijk gebruikt kan worden.
Increment	Periode van 2 weken. Zie pagina 3 voor meer informatie.
PDF	Project Definition, vergelijkbaar met een Plan van Aanpak.
PR	Problem Report, verzoek om een fout in de software te verbeteren.
RCS	Revision Control System, systeem om versies van tekstbestanden (bijvoorbeeld broncode) te archiveren.
UML	Universal Modelling Language, notatie gebruikt om objectklassen te beschrijven.

HOOFDSTUK 0. VERKLARENDE WOORDENLIJST

URD	User Requirements Document, beschrijft de initiële bedoeling van een project.
XML	eXtensible Markup Language, taal om data in te beschrijven die eenvoudig uitbreidbaar is.
XSLT	eXtensible Stylesheet Language Transformations, taal waarmee XML bestanden omgezet kunnen worden naar tekstbestanden met een mogelijk geheel andere opbouw.

Hoofdstuk 1

Inleiding

Vlakbij Eindhoven Airport, aan de Luchthavenweg zit de firma Delem. Dit is een bedrijf dat zich richt op het ontwerpen en produceren van besturingen van drukpersen ten behoeve van metaalbewerking. Van februari tot juni 2004 heb ik stage gelopen bij de afdeling Development.

Er werken ongeveer 60 mensen bij Delem, waarvan ongeveer 30 in de afdeling Development. De overige werknemers werken bij de productie, sales en ondersteunende afdelingen.

Gezien Delem zich in een zeer gespecialiseerde markt bevindt, is kwaliteit afleveren zeer belangrijk. Maar minstens zo belangrijk is het bewaken van deze kwaliteit. Dat is in principe de essentie van deze opdracht, die in hoofdstuk 2 op pagina 7 verder beschreven zal worden.

Daarna zullen vervolgens de onderzoeken, het proces, de implementatie en uiteindelijk de conclusie volgen. Elk hoofdstuk geeft een kijk op een ander deel van de stage, waarbij de conclusie vanzelfsprekend een afsluitende kijk op het geheel geeft.

HOOFDSTUK 1. INLEIDING

Hoofdstuk 2

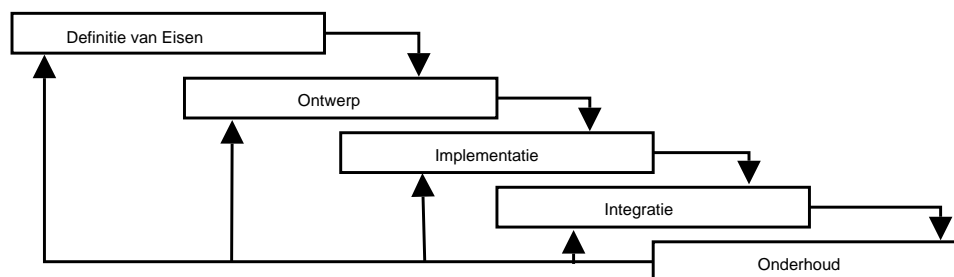
Proces

Dit hoofdstuk beschrijft het ontwikkelproces van deze stage. Hierbij zal de manier van werken centraal staan, alswel het verloop.

2.1 Projectmanagement

Zoals in het Plan van Aanpak vermeld staat (in het de paragraaf fasering op bladzijde 27), is binnen Delem elk project opgedeeld in zogenaamde *milestones*. Voor dit project is elke milestone weer opgedeeld in zogenaamde *increments*, een periode van twee weken. Hierbij stond elke increment een hoofdfeature centraal. Dit zal in de overzichten op de pagina's hierna te zien zijn.

Dit is te vergelijken met het standaard waterval model:



Het grootste verschil met het standaard waterval model, is dat nu het model meerdere keren doorlopen wordt (namelijk één waterval per increment). Verder wordt aan het begin van elke increment naar de requirements gekeken, in tegenstelling tot eenmalig aan het begin van het project. Tenslotte is de terugkoppeling met de gebruikers iets waarin het 'standaard' watervalmodel helemaal niet voorziet.

2.2 Besprekingen

Na een gesprek met de bedrijfsbegeleider, besloten we elke maandag een uur te vergaderen over het project. Hierbij stonden de volgende zaken centraal:

- Voortgang
Wat is er van de zaken die vorige keer besproken zijn terecht gekomen?
- Toekomst
Wat is de bedoeling voor volgende weken?

Deze aanpak is gekozen, omdat de uiteindelijke requirements van het project (dus een concrete lijst wat het moet kunnen) niet beschikbaar was. Vanuit Delem was het idee om ontwikkelaars het programma te laten gebruiken en aan de hand van het commentaar erop het te laten groeien. Zo kwam er een programma precies volgens de wensen van de ontwikkelaars (dit is prima omdat het voor intern gebruik is)

Na elke increment is er gekeken naar wat we nu hadden en wat de bedoeling was om eraan toe te voegen. Zo kwam er op de uiteindelijke planning naar voren dat er halverwege increment 6 weinig meer aan toe te voegen was, omdat er geen nieuwe requirements waren die geïmplementeerd moesten worden.

Om het verloop zo goed mogelijk weer te geven, is er op de volgende bladzijde de initiële planning te zien en op de bladzijde daarna het definitieve resultaat van de planning.

Hoofdstuk 3

De opdracht

De opdracht van deze stage was het ontwerpen en implementeren van een applicatie die verkeer op de CAN Bus van Delem besturingen kan analyseren en kwalificeren. De exacte invulling van deze opdracht werd tijdens de stage zelf ingevuld, aan de hand van feedback van de ontwikkelaars.

Dit had tot gevolg dat de tool zo breed en uitgebreid mogelijk opgezet diende te worden, zodat er altijd meer bijgebouwd kon worden. Een eis was wel, dat bestaande uitvoer van de Warwick X-Analyzer tool gelezen moest kunnen worden, alsmede real-time analyse door middel van een Softing CANusb en CANcard device.

Een groot probleem was het verzuimen tot updaten van documentatie. Steeds als er nieuwe commando's op de bus bijgevoegd werden, werd vaak de documentatie niet netjes up-to-date gebracht. Na verloop van jaren was er dus vrij veel niet meer duidelijk.

Om dit verschijnsel tegen te gaan, moest er een makkelijk uitbreidbare definitie van commando's bedacht worden. De bedoeling was om deze definitie zo makkelijk mogelijk door ontwikkelaars zelf te laten aanpassen, terwijl vanuit deze definitie de benodigde structuren gegenereerd werden die het programma begreep. Een voorstel voor het formaat hiervan was XML¹.

Daarnaast werd het idee geopperd, om aangezien er toch een duidelijke definitie bestaat, ook direct documentatie te genereren vanuit deze definitie. Er bestond een Microsoft Word document, maar na verloop van tijd was dit hopeloos verouderd. Het idee achter het genereren van documentatie vanuit de definitie is: als het niet in het gegenereerde document staat, herkent de applicatie het commando ook niet en bestaat het commando logischerwijs niet.

¹Extensible Markup Language, een markup taal met zelf definiëerbare elementen

HOOFDSTUK 3. DE OPDRACHT

Het kwalificerende deel van de applicatie moet in staat zijn om mee te lopen tijdens het testen van de besturingssoftware. Mochten er dan problemen optreden tijdens het testen, moet er aan de hand van de log van de applicatie uitgezocht kunnen worden of er vreemde of ongeldige berichten op de HSB bus geweest waren. Deze logs moeten dus leesbaar in de applicatie zijn, zodat eventuele problemen snel geanalyseerd kunnen worden.

De initiële bedoeling was om ook een cyclus viewer te maken, die elke fase van het persen in beeld kon brengen. Nadat dit aan de ontwikkelaars voorgelegd was, bleek dat de besturing deze functionaliteit al had en het absoluut niet nuttig zou zijn om dit te implementeren.

Tenslotte was het ook de bedoeling, om ontwikkelaars zo makkelijk mogelijk zaken te laten toevoegen aan de applicatie. Mocht er een zeer specifiek probleem zijn om te debuggen, dan was het zeker niet de bedoeling om de hele applicatie opnieuw te moeten bouwen. Om hierin te voldoen is er een plugin structuur bedacht, waar door middel van DLL files extra functionaliteit toegevoegd kan worden.

Hoofdstuk 4

Onderzoek

Zoals in de opdrachtschrijving op pagina 7 te zien is, was het noodzakelijk om een aantal onderzoeken uit te voeren, aan de hand van welke besluiten genomen konden worden. In dit hoofdstuk komen al deze onderzoeken aan bod, met hun uitvoering en conclusies.

4.1 CAN Hardware

Een van de eerste onderzoeken was een analyse van de huidige beschikbare CAN hardware, die de verbinding tussen de CAN bus en de PC verzorgen.

Een eis was wel, mocht er ooit besloten worden andere hardware aan te schaffen, het mogelijk moest zijn hier makkelijk een driver voor te maken. Dit is opgelost door een generieke architectuur te ontwerpen, die later toegelicht zal worden.

Op internet heb ik de specificaties opgezocht van een aantal leveranciers van CAN hardware en hiervan een duidelijk overzicht gemaakt. Ook de API wordt hierbij onder de loep genomen, om er zeker van te zijn dat software daadwerkelijk aansluit bij de hardware.

De uiteindelijke conclusie hier was, dat de huidige Softing CANusb dongel prima geschikt was voor de Delem toepassingen. Er waren veel luxere en minder luxe te krijgen, maar omdat er hier al enkelen van in het bezit van Delem zijn, is er besloten deze te gebruiken.

4.2 Definitie

Zoals in de opdrachtoomschrijving te vinden is, was een verdere eis het omschrijven van de HSB commando's in een eenvoudig formaat. Dit formaat moest makkelijk uitbreidbaar zijn, zonder in de knoop te raken met bestaande definities.

Om dit te realiseren werd door een ontwikkelaar XML voorgesteld. Om te kijken of die geschikt was, is er eerst gekeken wat voor zaken er gespecificeerd dienden te worden om alle commando's eenduidig te kunnen identificeren.

Nadat dit bekend werd, is er een voorbeeld XML bestand gemaakt met wat commando's erin en via XSLT omgezet naar C structuren, die door een preprocessor module toegepast word op de inkomende gegevensstroom.

De versie die dit ondersteunde is omgedoopt tot prototype en is uitvoerig besproken met de bedrijfsbegeleider. Deze was dusdanig enthousiast dat er besloten is op deze manier door te gaan.

Hierna is er gekeken om deze definities ook naar HTML te converteren, als documentatie. Toen later bleek dat je met een HTML bestand en een HHC¹ file een CHM file kan maken is ervoor gekozen om dit als help formaat binnen de applicatie te gebruiken, zodat commando's makkelijker op te zoeken zijn.

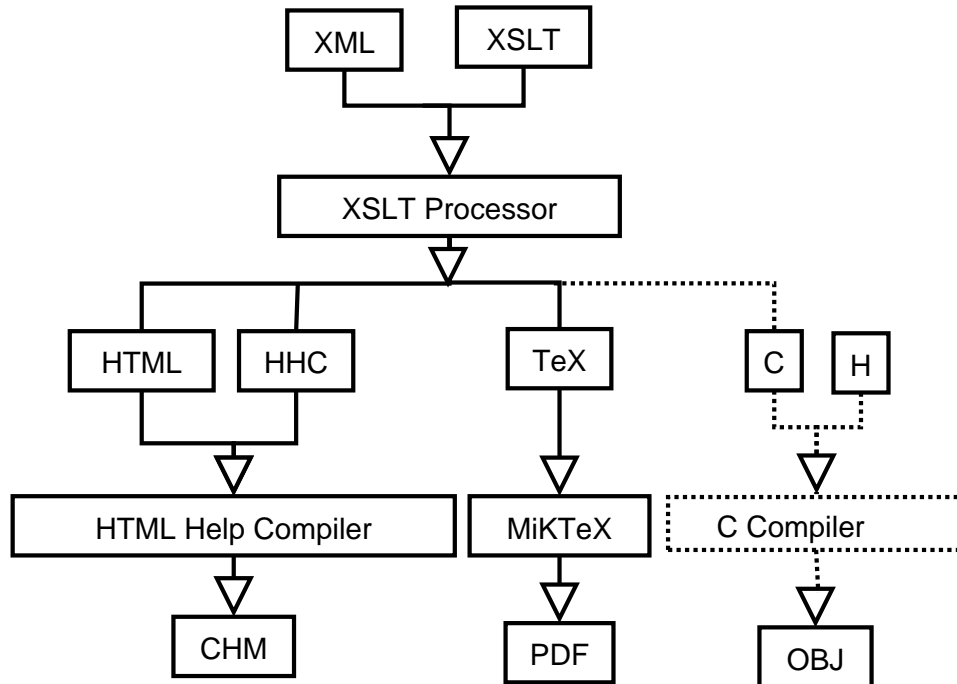
Het was uiteindelijk ook de bedoeling om het bestaande document met commando's te kunnen genereren vanuit de definitie. Dit overzicht zal aan het bestaande document toegevoegd worden, als een appendix. De definities worden weer door XSLT geprocessed, maar dan naar L^AT_EX-formaat. Dat laatste wordt door L^AT_EX-uiteindelijk geprocessed naar een PDF bestand, dat netjes uit te printen is.

Later is er ook gekeken naar de mogelijkheid om deze definities rechtstreeks uit de XML file zelf te lezen. Na een korte vergelijking tussen MSXML en libxml2 is er gekozen voor de laatste, aangezien MSXML zeer omslachtig in het gebruik is. Verder is de documentatie van libxml2 veel beter en kon er meteen mee aan de slag worden gegaan. Het resultaat is dat je definities voortaan kunt aanpassen zonder het programma opnieuw te moeten compileren.

¹Inhoudsopgave bestand voor een CHM file

4.2.1 Overzicht

Uiteindelijk ziet het overzicht van het XML-transformatie proces er als volgt uit:



In dit overzicht is het C-gedeelte met stippellijnen aangeduid, omdat dit stuk later kwam te vervallen.

4.3 Uitbreidbaarheid

Aan het einde van het project bleek dat er een extreem grote behoefte was tot het eenvoudig ontwikkelen van plugins, die ervoor zorgen dat de huidige functionaliteit eenvoudig uit te breiden is. Er werden twee mogelijkheden bekeken: scripts en DLL bestanden.

Het voordeel van scripts was dat ze erg makkelijk te maken en onderhouden zijn. Helaas was de verwachting dat het realtime gedrag niet meer gegarandeerd kan worden als er voor een dusdanige oplossing gekozen zou worden. Daarom is deze oplossing verder dus niet verder onderzocht.

DLL bestanden leken ideaal: het is gewoon C++ code, je kunt bestaande code recycelen. Nadat er dan ook een goede API ontworpen was, die later flink uitgebreid is, is dit dan ook gepresenteerd aan de ontwikkelaars. Deze API werkt prima en er zijn geen verschillen bemerkbaar in snelheid.

HOOFDSTUK 4. ONDERZOEK

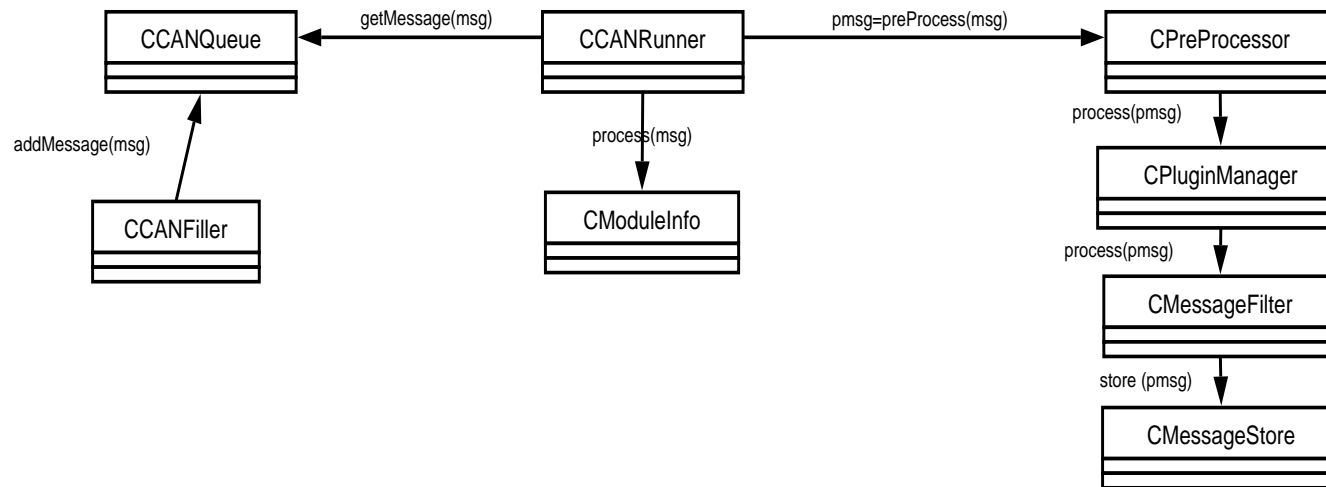
Hoofdstuk 5

Implementatie

Dit hoofdstuk presenteert een overzicht van de implementatie van de applicatie. Hierbij komt de interne opzet van het programma naar voren. De gebruikte software is opgenomen als een bijlage die te bekijken is op pagina 37.

5.1 Ontwerp

Aangezien de applicatie in principe een datastroom moet analyseren en eventueel weergeven, moet het ontwerp beschrijven hoe data door de applicatie heen stroomt. Zie het volgende plaatje:



Omdat het hier alleen de datastroom beschrijft en verder geen directe interfacing, is er gekozen voor een eenvoudigere representatie dan UML. De insteek is om de dit diagram ook leesbaar te laten zijn voor mensen die geen kennis van de zogenaamde use-cases hebben.

HOOFDSTUK 5. IMPLEMENTATIE

De berichten worden in principe opgebouwd door een *CCANFiller* superklasse. Deze klasse zorgt ervoor, dat berichten van een bron (zoals de CAN hardware, maar bijvoorbeeld ook berichten uit een bestand) opgeslagen worden in een zogenaamde *CCANQueue*. Om ervoor te zorgen dat het ontvangen, bewerken en weergeven synchroon van elkaar gebeurt, bevindt de *CCANFiller* zich in een eigen thread.

Alle te behandelen berichten komen dus in de *CCANQueue*. Deze klasse wordt door middel van mutexes beschermd tegen deadlocks en dergelijke, gezien hij door meerdere threads tegelijk wordt gebruikt.

De *CCANRunner* superklasse wacht net zo lang tot de *CCANQueue* een bericht teruggeeft dat behandeld dient te worden. Zoals reeds vermeld gebeurt het afhandelen in een aparte thread. Het enige dat de *CCANRunner* doet is de berichten door alle subsystemen heen voeren.

Tijdens het verwerken worden de berichten eerst door de *CModuleInfo* klasse gevoerd. Deze kijkt of er modules op de HSB bus toegevoegd of aangepast moeten worden. Dit gebeurt voor de daadwerkelijke verwerking op berichtniveau, aangezien bepaalde typen berichten (de zogenaamde sequencer berichten) pas geïdentificeerd kunnen worden als het moduletype bekend is. Aangezien deze klasse nooit het bericht uitbreidt, maar alleen de interne module administratie, is dit als een aparte pijl getekend in het diagram op de vorige pagina.

Nu komt de klasse die daadwerkelijk de berichten behandelt: de *CPreProcessor* klasse. Deze klasse breidt de bestaande, rauwe bericht-data uit aan de hand van de definities in de ingelezen XML file. De uitvoer van deze klasse bevat dan ook het complete rauwe bericht met daarin ook de commandoinformatie, de argumenten, de bijbehorende module enzovoorts.

Deze informatie wordt aan de *CPluginManager* klasse doorgegeven, die eventuele plugins de mogelijkheid geeft om de data verder te analyseren en gegevens ten behoeve van logging eraan toe te voegen. De berichtenstroom kan normaliter niet veranderd worden (tenminste, niet zonder er speciale moeite voor te doen). De plugins van de *CPluginManager* klasse kunnen aangeven dat iets geforceerd gelogd moet worden en met welke melding dat moet.

Aangezien nu alle behandeling gereed is, rest het de *CMessageFilter* klasse om te kijken of berichten gelogd moeten worden. Als de *CPluginManager* of *CPreProcessor* iets specifiek te melden hebben, dan wordt het bericht altijd gelogd.

Wanneer een bericht geschikt is bevonden om te loggen, dan wordt het uiteindelijk aan de *CMessageStore* toegevoegd. Dit is een buffer die berichten kan bewaren. Het gebruikersinterface staat rechtstreeks in verbinding met

deze klasse, zodat berichten niet dubbel opgeslagen hoeven te worden.

5.2 Documentatie

Sinds het begin van het project heb ik aangegeven eigen ideeën te hebben over hoe het project gedocumenteerd kon worden. Dit is dan ook met mijn bedrijfsbegeleider besproken, waarna we uiteindelijk tot de volgende opzet kwamen:

- Code documentatie via Doxygen
Met behulp van het reeds genoemde Doxygen programma is er aan de hand van speciale constructies in de code documentatie gemaakt. Hiermee was snel op te zoeken wat een klasse of functie doet, welke in- en uitvoer die heeft, enzovoorts.
- Architectuur Model in Microsoft Word
In het architectuur model staat het ontwerp van het programma uitgelegd. Dit lijkt erg veel op het ontwerp op bladzijde 14, maar is veel meer toegespits op de code. De nadruk ligt dan ook op de link tussen functionaliteit van het programma en de code die het verzorgt.

Het architectuur model is later gereviewd door ontwikkelaars, waarna er uiteindelijk een duidelijk document uitkwam.

Hoofdstuk 6

Conclusie

Dit verslag bied een overzicht van het verloop van mijn werkzaamheden tijdens mijn 2^e stage te Delem BV. Na het gehele verloop van de stage is er een programma gemaakt, waarvan ik hoop dat Delem er veel aan zal hebben. De opzet was het programma zo uitbreidbaar en onderhoudbaar mogelijk te maken, iets wat goed gelukt is.

Tijdens deze stage heb ik, behalve aan de stageopdracht te werken, ook ongeveer één dag per week aan andere activiteiten besteed. Een goed voorbeeld hiervan is het testen van een bestaande applicatie VBend¹, waarvan binnenkort een nieuwe versie beschikbaar zou komen. Hierbij ben ik volledig in het projectteam opgenomen en heb ik goed kennis leren maken met de gang van zaken tijdens 'normale' projecten. Ook kwam het gebruik van PR's en CR²'s erg goed naar voren.

Verder heb ik tijdens deze stage heel uitgebreid kennis gemaakt met XML, wat wellicht later nog van pas kan komen. Ook merkte ik dat mijn Windows programmeerkennis niet meer optimaal was (ik gebruik zelf thuis bijna geen Windows), daarin heb ik mezelf ook een stuk meer verdiept.

Ik vind dat ik veel vrijheid heb gehad tijdens het uitoefenen van de stage, ik verwacht dat dit is omdat ik inmiddels al 2 jaar bij de firma Delem werkzaam ben. Hierbij ben ik van mening dat ik nu pas het bedrijf echt goed heb leren kennen, gezien ik eigenlijk pas nu het gevoel heb dat ik weet hoe de projecten daar werken .

Tenslotte denk ik dat dit een geslaagde stage was. Het is goed gedocumenteerd, er is een goed functionerend programma opgeleverd en er zijn al een aantal problemen mee gevonden die wellicht in de toekomst aangepakt

¹Virtual Bend, applicatie die een optimale buigvolgorde kan berekenen en simuleren

²Problem Report/Change Request, zie pagina 30 voor meer informatie

HOOFDSTUK 6. CONCLUSIE

zullen worden. Al met al een prima resultaat, wat ook door mijn projectbegeleider bij Delem bevestigd is.

Hoofdstuk 7

Bronvermelding

Hoogland, W., *Rapport over Rapporteren*. 3^e druk, Groningen, 1998

Nederlands Computerwoordenboek, <http://computerwoorden.nl>

HOOFDSTUK 7. BRONVERMELDING

Bijlage A

Plan van Aanpak

Deze bijlage bevat het plan van aanpak. Hierbij is de woordenlijst komen te vervallen, sinds het stageverslag een uitgebreidere woordenlijst bevat. Deze is te vinden op pagina xi.

Samenvatting

Dit verslag behandelt de opzet van de HSB-Bus-Analyser stage opdracht.

De achtereenvolgende hoofdstukken gaan in op verscheidene zaken met betrekking tot de opdracht, zoals:

- Bedrijfsprofiel van het stagebedrijf, de firma Delem
- Gedetailleerde beschrijving van het project
- Een gefaseerde aanpak van de stage

De belangrijkste conclusie is dat aan de hand van analyses veel functionaliteit verder gespecificeerd zal gaan worden. Uiteindelijk zal dit plan van aanpak dan ook met deze bevindingen uitgebreid worden.

A.1 Inleiding

Goede kwaliteit afleveren is voor ieder bedrijf erg belangrijk. Voor grote, complexe producten wordt dit steeds lastiger, vooral als deze uit meerdere sub systemen bestaan.

Delem besturingen functioneren door modules opdrachten te geven, dit gebeurt via berichten over de HSB¹-bus. Om aan de steeds groter wordende vraag om kwaliteit de voldoen, is het zeer nuttig een HSB Bus Analyser te hebben, die het mogelijk moet maken om op een makkelijke en goede manier snel problemen met betrekking tot de HSB bus te analyseren.

In hoofdstuk 3 op pagina 26 zal het project compleet beschreven worden, evenals de aanpak ervan.

A.2 Het bedrijf Delem

Delem werd in 1976 opgericht door de heer H.J.M.M. van Doorne. Op het moment is dit bedrijf de technologische leider op de wereldmarkt in het gebied van besturingen voor metaalbewerkingsmachines. Dit succes komt tot stand door het gebruik van de meest recente techniek in de producten. Delem levert uitsluitend aan OEM²-ers (dus de bedrijven die metaalbewerkingsmachines maken) en niet aan eindklanten.

Er zijn ongeveer 60 personen werkzaam bij Delem. Ongeveer 30 hiervan zijn ICT-ers met gemiddeld een WO niveau.

Delem is opgedeeld in een aantal afdelingen, waarbij de voornaamste development, solutions, productie en sales zijn. Uiteraard is er ook nog ondersteunend personeel aanwezig, maar hier wordt verder niet op ingegaan. Een organigram is te vinden op pagina 24.

In de volgende paragrafen staat een overzicht van de afdelingen die tijdens de stage van belang zijn:

A.2.1 Afdeling Development

Deze afdeling is verantwoordelijk voor het ontwikkelen van nieuwe hard- en software (voor zowel de besturing als voor een PC) en het onderzoeken van nieuwe mogelijkheden voor de besturingen. Gezien de aard van de stage wordt deze dan ook bij deze afdeling uitgevoerd. Zoals tijdens de planning

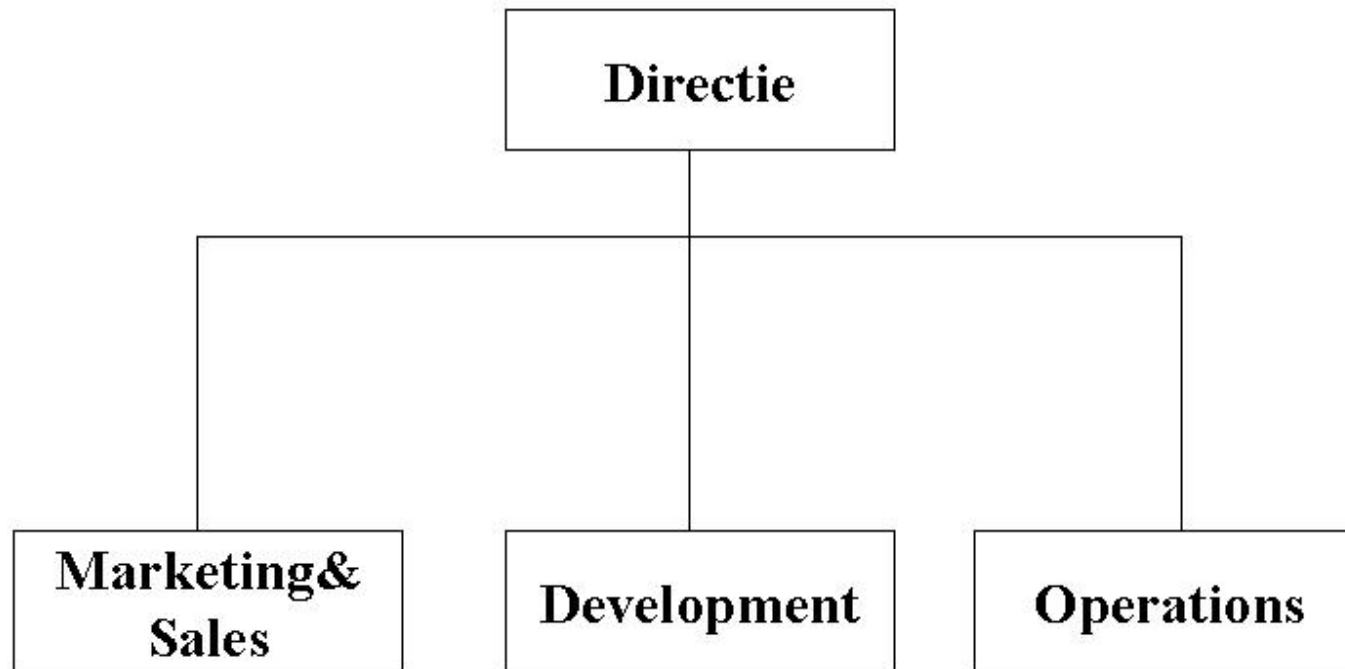
¹Hoge Snelheid Bus, ook wel bekend als CAN bus

²Original Equipment Manufacturer

BIJLAGE A. PLAN VAN AANPAK

op bladzijde 33 te zien is, is er ook tijd gereserveerd voor assistentie voor de andere activiteiten van deze afdeling.

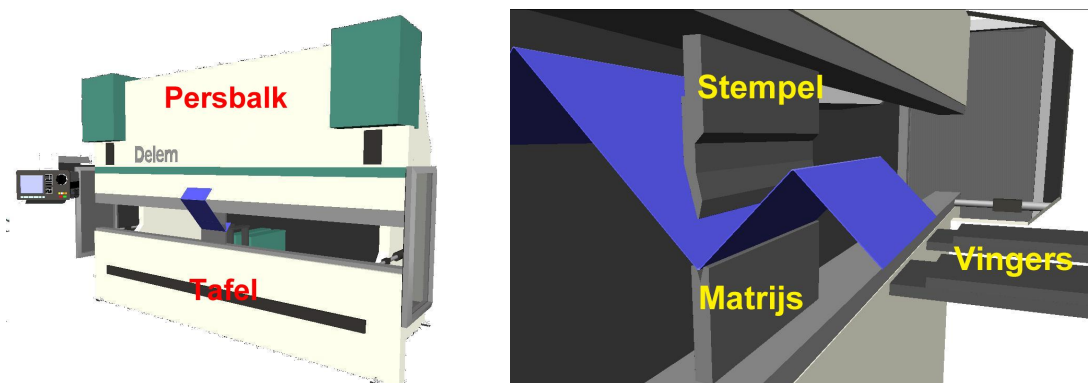
A.2.2 Organigram



A.3 Achtergrondinformatie

A.3.1 De pers

Om een goede indruk van een drukpers te krijgen, zijn hieronder wat plaatjes te zien van een pers met daarin een metalen plaat die gebogen wordt:



Zoals in de plaatjes boven te zien is, wordt een product³ gebogen doordat de *persbalk* naar beneden beweegt. Hierdoor komt de *stempel* tegen het product aan, die door de *matrijs* tegengehouden wordt. De plaat hiertussen verbuigt daardoor, hetgeen resulteert in een buiging. De *vingers* ondersteunen hierbij het product, zodat het op zijn plaats blijft.

A.3.2 De modules

De persbalk en de vingers worden aangestuurd door zogenaamde modules, die via een CAN⁴ door de besturing⁵ worden aangestuurd.

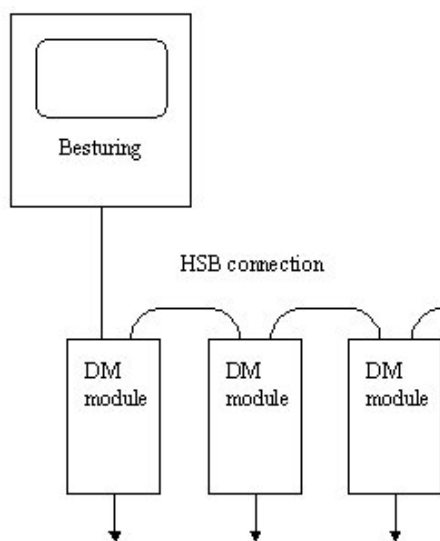
Een overzicht van de besturing met de modules:

Deze stageopdracht richt zich op de DM modules, en niet op de besturing zelf. De modules zelf sturen bijvoorbeeld motoren aan, of staan in verbinding met sensoren.

³In de plaatjes de blauwe plaat

⁴Controller Area Network, Ook wel bekend als HSB-bus

⁵Op de vorige pagina het kastje uiterst links



A.4 Het project

A.4.1 Informatie vooraf

Delem besturingen communiceren met de zogenaamde CAN-bus naar modules. Een module kan bijvoorbeeld een motor aansturen, die er bijvoorbeeld voor zorgt dat de persbalk naar een bepaalde positie gaat.

A.4.2 Organisatie

De opdrachtgever van het project is de firma Delem, die als stagebegeleider de heer M. Scholte heeft aangewezen.

De opdrachtnemer is de heer R. Springer, student van Fontys Hogescholen te Eindhoven. De begeleidende docent is de heer H. van Heumen.

A.4.3 Probleemstelling

Op het moment is er geen manier waarop modules specifiek getest kunnen worden. Om nu een module te testen wordt er soms een compleet systeem gebouwd met besturing en motoren.

Aangezien dit niet praktisch is, hebben engineers zelf oplossingen bedacht om toch zoveel mogelijk lokaal te kunnen testen. Hierbij werd vooral de scripttaal Perl⁶ gebruikt, samen met een hele hoop tooltjes.

⁶www.perl.com

A.4.4 Resultaat

De bedoeling is een programma te ontwikkelen dat modules kan testen. Aangezien er nog onduidelijk is wat er het beste getest kan worden en welke CAN PC hardware het beste gebruikt kan worden, zal er eerst een analyse uitgevoerd worden om te kijken welke hardware het beste voldoet.

De functionaliteit die er absoluut in moet zitten, is het leesbaar weergeven van berichten op de CAN bus, alsmede de mogelijkheid om zelf berichten naar modules te kunnen sturen. Dit kan niet met conventionele programma's, aangezien Delem zelf berichten toegevoegd heeft die niet in de CAN specificatie staan. Extra functionaliteit zal na de analyse en in de loop van het project toegevoegd worden.

A.4.5 Randvoorwaarden

De randvoorwaarden van dit project zijn als volgt:

- Alle toegezegde middelen blijven beschikbaar tot het eind van de stageperiode.
- Belanghebbenden van het project maken genoeg tijd beschikbaar voor ondersteuning van de stagiair.

A.4.6 Risico's

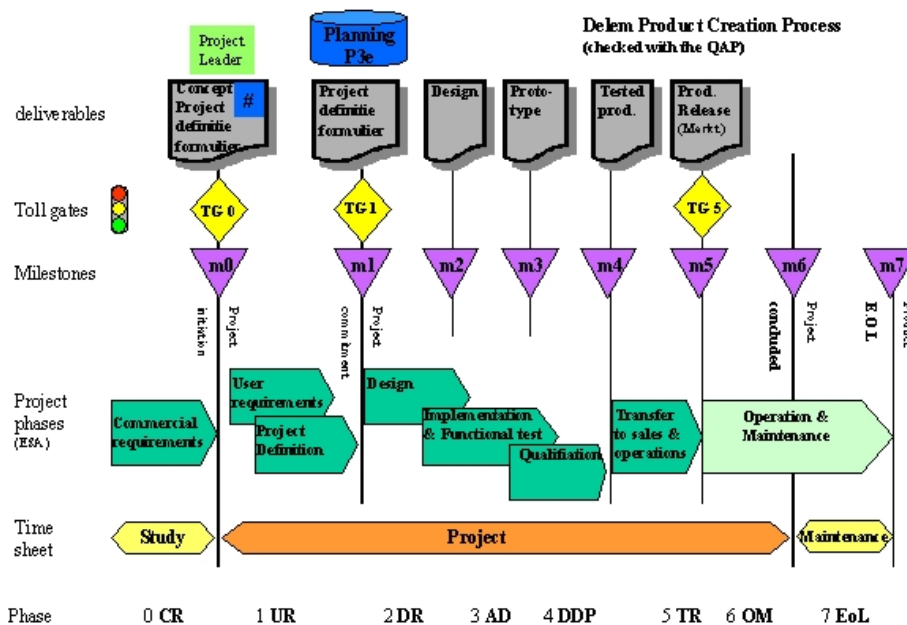
Zoals bij elke stageopdracht is er het risico dat er niet genoeg kennis bij de stagiair aanwezig is, wat als resultaat heeft dat het project niet op tijd af komt. Aan de hand van de ervaring van de bedrijfsbegeleider en de kennis van de aanwezige ICT-ers bij het bedrijf en de stagiair zelf is de kans dat dit gebeurt klein.

A.4.7 Fasering

Bij Delem wordt de doorlooptijd van projecten in zogenaamde *milestones* verdeeld. Milestones worden weer onderverdeeld in zogenaamde *increments*, een periode van 2 weken. Hieronder is een overzicht te vinden van de milestones, inclusief hun rol met betrekking tot dit project:

BIJLAGE A. PLAN VAN AANPAK

Overzicht



Toelichting

Hieronder volgt een toelichting van de fasering, zoals die in het plaatje hierboven te zien is:

- Milestone 0: Studie
Het doel hier is het project te definiëren. De uitkomst hiervan is normaal gesproken een commerciële beschrijving, zoals wat er nu in SAIS⁷ te vinden is. Dit wordt een CRS⁸ genoemd.
- Milestone 1: Voorbereiding
Tijdens deze fase wordt het product formeel voorbereid. Hierbij worden vooral zaken geanalyseerd. Het hoofdstuk planning op bladzijde 33 in dit Plan van Aanpak wordt uitgebreid aan de hand van de bevindingen hiervan. Het directe resultaat is het Plan van Aanpak. Dit wordt een URD⁹ genoemd.

⁷Fontys' Stage- en Afstudeer Informatie Systeem, zie <http://intranet.hi.fontys.nl/sais>

⁸Commercial Requirements Specification

⁹User Requirements Document

- Milestone 2: Ontwerp
In deze fase wordt het product echt ontworpen. Als het een hardwareproduct is worden de schema's ontwikkeld, bij softwareproducten worden hier bijvoorbeeld de UML¹⁰ schema's getekend.
- Milestone 3: Implementatie en testen
Nu wordt het product daadwerkelijk gemaakt. De bedoeling is dat aan het eind van deze milestone een prototype van het product af is.

Het product wordt tijdens het implementeren ook getest door de ontwikkelaars zelf. Dit resulteert in CR's en PR's, hier zal later op worden ingegaan.
- Milestone 4: Product test release
Hier wordt het product vrijgegeven voor interne release, waar het ook uitgebreid getest gaat worden door de testafdeling. Dit resulteert in nog meer CR's en PR's.

Aangezien dit een product voor intern gebruik is, zijn milestone 5: Markt release, milestone 6: Project sluiting en milestone 7: onderhoud weggelaten. De uiteindelijke planning is te vinden op bladzijde 33.

Op de volgende pagina is een overzicht te zien van de projectplanning zoals deze bij Delem gehanteerd wordt:

A.4.8 Hulpprogramma's

Er wordt bij Delem gebruikt gemaakt van een drietal programma's, waarmee de bronnen in een project en de status ervan goed bewaakt kunnen worden. Deze programma's zijn:

- Primavera Project Manager
Met dit programma kunnen alle activiteiten omtrent een project ingepland worden. Via Primavera Progress Reporter kunnen mensen die aan het project werken uren boeken op deze activiteiten. Zo is meteen te zien hoeveel uur ergens voor ingepland werd en hoeveel tijd er daadwerkelijk aan besteed is.
- PVCS Tracker
Dit programma slaat de CR's en PR's op in een database, waarop gezocht kan worden. Elke CR en PR heeft een eigenaar, die verantwoordelijk is voor de request. Opmerkingen en dergelijke kunnen bij de CR en PR gevoegd worden, zodat er altijd een goed overzicht is wat de status en geschiedenis van de request is.

¹⁰Universal Modelling Language

BIJLAGE A. PLAN VAN AANPAK

- PVCS Version Manager
Dit programma is vergelijkbaar met CVS¹¹ of diens voorganger RCS¹². Alle projectbestanden worden hiermee onder versiebeheer geplaatst. Dit zorgt ervoor dat alle veranderingen tot de projectbestanden, of dat nu broncode, documentatie of wat dan ook is, altijd gearchiveerd word.

CR's en PR's

Zoals in de vorige paragraaf al naar voren kwam, zijn PR¹³'s en CR¹⁴'s erg belangrijk. Deze twee zaken kunnen worden gecreëerd en opgezocht worden met behulp van de PVCS Tracker.

Het doel hiervan is een duidelijk overzicht te hebben van de taken die voor het project uitgevoerd moeten worden. Verder kan er met behulp van de Progress Reporter aangegeven worden hoeveel tijd er daadwerkelijk gespendeerd is aan een taak, zodat er meteen een duidelijk overzicht is.

Change Control Board

CR's en PR's worden beheerd door middel van een CCB¹⁵. Als een CR/PR is ingeschoten, dan besluit normaal gesproken het CCB (dit is normaal gesproken een team van de projectleider, projectleden en sales/management personen) of die CR/PR wel of niet gehonoreerd wordt.

A.4.9 Analyses

De volgende analyses zullen worden uitgevoerd:

- Hardware analyse
Het doel van deze analyse is om uit te zoeken wat de beste hardware is om met de modulen te communiceren.
- Functionaliteit analyse
Deze analyse concentreert zich op de functionaliteit die het programma moet krijgen. Om dit te doen zullen de mensen die veel met DM modulen te maken hebben, naar hun mening hierover gevraagd worden. Aan de hand hiervan zal er samen met de bedrijfsbegeleider

¹¹Concurrent Version System, zie www.cvs.org

¹²Revision Control System

¹³Problem Report

¹⁴Change Request

¹⁵Change Control Board

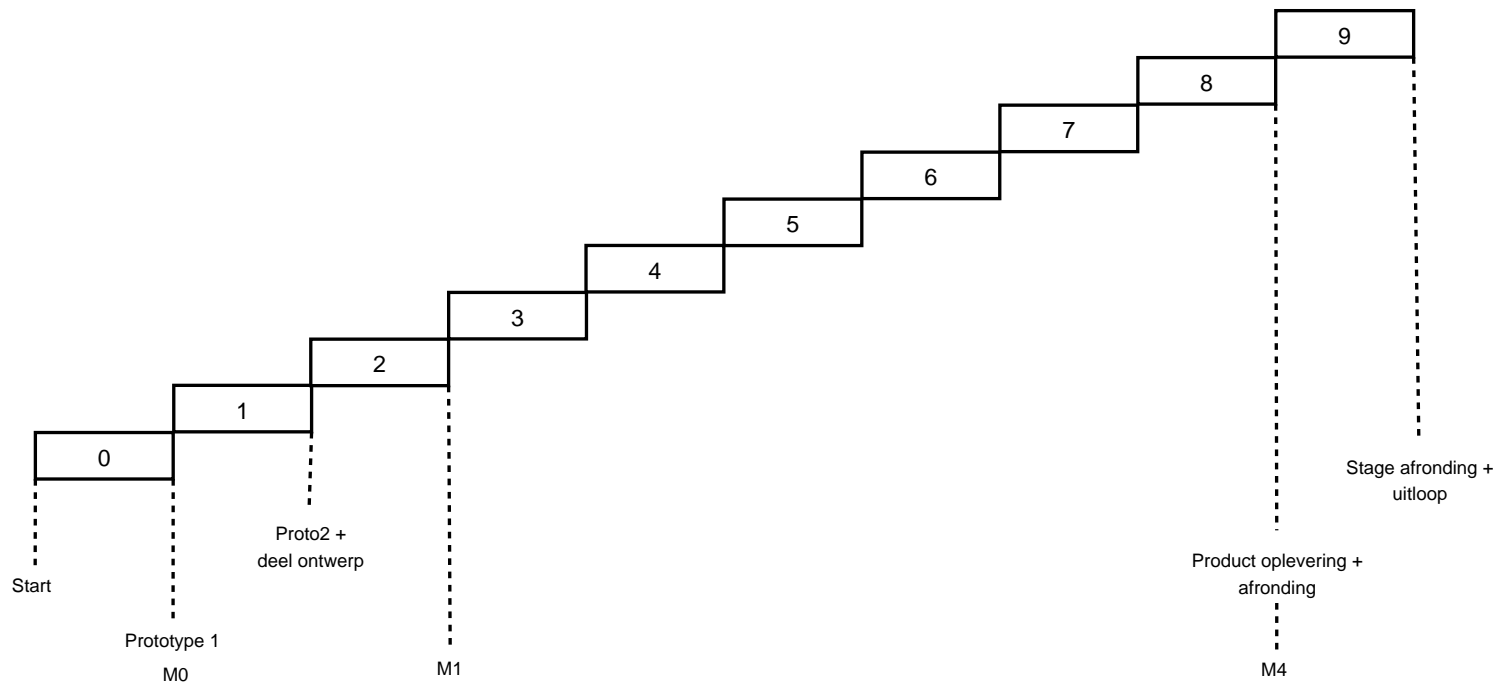
BIJLAGE A. PLAN VAN AANPAK

een overzicht gemaakt worden van functionaliteit die erin moet zitten, functionaliteit die gewenst is en functionaliteit die niet noodzakelijk is.

A.5 Planning

Dit hoofdstuk zal een overzicht van de planning geven. Op de volgende bladzijde is een overzicht van de planning te zien, dat daarna toegelicht zal worden.

A.5.1 Overzicht



In dit schema geven de rechthoekige genummerde vakken de increments aan. De gestippelde lijnen geven de gepland afgeronde activiteiten aan en de geplande milestones.

BIJLAGE A. PLAN VAN AANPAK

A.5.2 Toelichting

Zoals in de fasering op bladzijde 27 besproken werd, wordt het project ingedeeld in milestones. Deze zijn zelf weer opgedeeld in één of meerdere increments.

Increment	Datum	Activiteit
0	2 - 13 Februari	Opstellen en aanpassen Plan van Aanpak, analyses uitvoeren, prototype1
Milestone 0: Studie		
1	16 - 27 Februari	Prototype2 + deel ontwerp
2	1 - 12 Maart	
Milestone 1: Voorbereiding		
3	15 - 26 Maart	Afronden en opleveren product
4	29 Maart - 9 April	
5	12 - 23 April	
6	26 April - 7 Mei	
7	10 - 21 Mei	
8	24 Mei - 4 Juni	
Milestone 4: Product test release		
9	7 - 18 Juni	Stage afronden, presentatie houden, stageverslag maken

Zoals tijdens de probleemstelling (zie pagina 26) te zien is, zal al lopende het project extra functionaliteit aangedragen worden. Deze planning zal dan ook aangepast worden.

Het faseringsoverzicht op pagina 28 laat zien dat de milestones door elkaar heen lopen. Daarom zijn milestone 2: ontwerp en milestone 3: implementatie en testen niet in deze planning opgenomen.

Tenslotte wil Delem dat ik maximaal één dag in de week reserveer voor zaken die niet direct in verband staan met de HSB Bus Analyser, maar waar ik bijvoorbeeld Solutions werk ga doen. Dit omdat er tijdens het werken in een bedrijfssituatie regelmatig iets tussendoor komt wat hogere prioriteit heeft.

A.6 Beheersaspecten

A.6.1 Geld

Delem stelt een maandelijkse stagevergoeding beschikbaar, deze bedraagt €406,92 bruto per maand. Verder zal er ook de mogelijkheid om in overleg benodigde zaken te kopen.

BIJLAGE A. PLAN VAN AANPAK

A.6.2 Organisatie

Zie projectorganisatie op pagina 26.

A.6.3 Kwaliteit

In de eerste instantie zal het product zoveel mogelijk door de stagiair zelf getest worden tijdens het ontwikkelen, dit is gebruikelijk binnen Delem. Als de eerste release van het programma vrijgegeven wordt, zullen de gebruikers van het programma hun bevindingen melden, en er indien nodig een PR of CR van maken.

A.6.4 Informatie

Er zal om de week een overleg zijn met de bedrijfsbegeleider, alswel een tweewekelijkse rapportage naar de docentbegeleider.

De software voor het te maken project zal geschreven worden in Microsoft Visual C++ .NET, en moet werken op Windows 2000 en XP machines.

A.6.5 Tijd

De stage duurt tot 18 juni 2004. Op dit tijdstip moet het project afgerond zijn en aan alle formaliteiten voldaan zijn. In geval van ziekte en andere onvoorziene zaken is er eventueel één week extra.

A.7 Conclusie / Aanbevelingen

Ondanks het sterk informatieve karakter van dit verslag, is er een belangrijke conclusie te trekken.

Zoals tijdens de probleemstelling op pagina 26 al naar voren kwam, is het belangrijk dat er eerst door middel van een analyse een beeld ontstaat van de mogelijkheden van de hardware. Verder zal ook een overzicht van gewenste functionaliteit toegevoegd worden.

Na de eerste milestone zal het pas duidelijk zijn wat er uiteindelijk opgeleverd zal worden. Deze zal binnen het project aan de hand van CR's en PR's geregistreerd worden.

Bijlage B

Communicatieplan

Deze bijlage bevat het communicatieplan, zoals uiteindelijk met de docent-begeleider overeen gekomen. Uiteindelijk bleek het stageverslag eerder af te zijn dan gepland en word daarom ook eerder opgestuurd.

B.1 Betrokkenen

B.1.1 Student

R. P. W. Springer
Blaarthemseweg 35
5654 NR Eindhoven
Telefoon: 040-2528162
Email: rink.springer@delem.com

B.1.2 Bedrijf

M. Scholte
Luchthavenweg 42
5657 EB Eindhoven
Telefoon: 040-2552969
Email: marcel.scholte@delem.com

B.1.3 Fontys Hogescholen Informatica

J. B. H. M van Heumen
Rachelsmolen 1

BIJLAGE B. COMMUNICATIEPLAN

5600 AH Eindhoven
Telefoon: 0877-878550
Email: H.vanHeumen@fontys.nl

B.2 Algemene Informatie

- Voortgangsrapportage
De stagiair zal de docentbegeleider elke twee werkweken een voortgang van zijn stageactiviteiten geven.
- Vragen
Als de stagiair of de docentbegeleider een vraag heeft, is email de voorkeursmanier om contact op te nemen. Voor dringende zaken kan telefonisch contact gezocht worden.
- Veranderingen
Bij veranderingen stuurt de stagiair het vernieuwde communicatieplan per email op naar de docentbegeleider. Deze zal binnen één werkweek laten weten of hij met het nieuwe communicatieplan akkoord gaat.

B.3 Te ontvangen documenten door docentbegeleider

Wat?	Wanneer?	Hoe?
Communicatieplan	4 februari	Per email
Brief met uitnodiging bezoek Bijlage: Plan van Aanpak Bijlage: Routebeschrijving	11 februari	Per email
Stageverslag inhoudsopgave	8 juni	Per email
Stageverslag, eerste versie	10 juni	Per email
Stageverslag, tweede versie	16 juni	Per email
Stageverslag, laatste versie	18 juni	Per post
Voortgangsrapportage	Elke twee werkweken	Per email

B.4 Te ontvangen documenten door stagiair

Vragen en opmerking met betrekking tot de opgestuurde documenten of andere zaken, door middel van email.

Bijlage C

Gebruikte software

Hieronder volgt een overzicht van alle gebruikte software, in alfabetische volgorde:

- ActiveState Perl (<http://www.activestate.com>)
Perl is een bekende scripttaal. Deze is gebruikt om snel scripts te bouwen om dingen te controleren en zaken om te zetten.
- dia (<http://www.gnome.org/projects/dia/>)
Dia is een open source diagram editor, die ook onder Win32 werkt. Deze is zeer eenvoudig in het gebruik en kan gewoon als PNG-files diagrammen exporteren.
- Doxygen (<http://doxygen.org>)
Doxygen is een programma dat aan de hand van commentaar in de sourcecode documentatie van functies en klassen en dergelijke kan genereren. Ik heb het optionele GraphViz (<http://www.research.att.com/sw/tools/graphviz/>) programma ook geïnstalleerd, zodat het ook netjes grafische diagrammen genereerde van de klassen.
- GIMP (<http://gimp.org>)
The GIMP (GNU Image Manipulation Program) is een opensource programma waarmee plaatjes eenvoudig bewerkt kunnen worden. Dit is dan ook gebruikt voor de (zeer minimale) aanpassingen van illustraties in de documentatie.
- libxml2 (<http://xmlsoft.org>)
Deze open source bibliotheek is oorspronkelijk ontwikkeld voor het Gnome project¹. Gelukkig is hij buiten Gnome ook bruikbaar als een stabiele en snelle XML parser. Verder is ook de XSLT processor

¹www.gnome.org, een desktop omgeving voor *NIX

BIJLAGE C. GEBRUIKTE SOFTWARE

hiervan, xsltproc.exe, gebruikt om door middel van XSLT de definities om te zetten.

- MikTeX (<http://www.miktex.org>)
Dit is een Win32 L^AT_EX typesetting programma. Het is gebruikt voor het Plan van Aanpak en dit stageverslag, maar ook de PDF van de gedefinieerde commando's.
- Microsoft HTML Help Workshop
Dit programma is gebruikt om van de HTML uitvoer een CHM helpfile te maken.
- Microsoft Office 2000
Aangezien niemand bij Delem L^AT_EX gebruikt en het de bedoeling was om de architectuur documenten te kunnen onderhouden, is er voor gekozen dit in standaard Word te doen, dat onderdeel van Microsoft Office is.
- Microsoft Visual Studio .NET 2003
Deze welbekende C++ omgeving is gebruikt om het programma in te implementeren, met behulp van MFC².
- Merant Version Manager³
Dit programma is gebruikt als versie beheer systeem. Alle broncode en documenten staan hierin opgeslagen, evenals oudere versies.
- Merant Tracker
Dit programma is gebruikt om PR-en en CR-en bij te houden.
- vim (<http://www.vim.org>)
VIM is VI iMproved, een open source editor die erg op vi⁴ lijkt, maar een hele hoop verbeteringen heeft.
- Quintessential CD v1.27 (<http://www.quinnware.com>)
Deze Win32 CD speler is erg compact, en vermaakte mij tijdens het uitoefenen van mijn stageactiviteiten.
- zsh (<http://www.zsh.org>)
De Z Shell is een open source shell, die werkt op *NIX en Win32 machines. Deze is veel krachtiger dan de standaard Microsoft shell en werd daarom gebruikt in plaats van de laatste.

²Microsoft Foundation Classes

³Voorheen bekend onder de naam Primavera Version Manager, deze naam werd nog in het plan van aanpak gebruikt

⁴Visual edItor, een zeer standaard UNIX editor

Index

- Aanbevelingen, 34
- Afdelingen, 22
 - development, 22
 - productie, 22
 - sales, 22
- Analyse, 27
 - functionaliteit, 30
 - hardware, 30
- API, xi, 9

- Besprekingen, 4
- Besturing, xi, 25
- Bronvermelding, 19

- CAN, xi, 7, 26
- CCB, xi, 30
- Communicatieplan, 35
- Conclusie, 17, 34
- CR, ix, xi, 17, 30
- CVS, xi

- Deadlock, xi
 - deadlock, 15
- Delem, 22
- dia, 37
- doxygen, 37
- Drukpers, 25

- Fasering, 27
 - overzicht, 28
 - toelichting, 28

- Gebruikte software, 37
- GIMP, 37

- HSB, xi, 22
- Hulpprogramma's, 29

- Implementatie, 13
- Increments, xi, 3, 27, 33
- Inleiding, 1, 22

- LaTeX, xi, 38

- Matrijs, 25
- MFC, xi
- MikTeX, 38
- Milestones, xi, 3, 27, 33
- Module, xi, 25, 26
- Mutex, xi, 15

- Ontwerp, 14
 - opdracht, 7
- Opdrachtgever, 26
- Opdrachtnemer, 26
- Organigram, 24

- PDF, xi
- Perl, 26, 37
- Persbalk, 25
- Planning
 - Initiële, 5
 - overzicht, 32
- Planning,toelichting, 33
- PNG, 37
- PR, ix, xi, 17, 30
- Primavera
 - Progress Reporter, 29
 - Project Manager, 29
- Proces, 3
- Project, 26
 - informatie, 26
 - probleemstelling, 26
 - resultaat, 27
- projectmanagement, 3

INDEX

PVCS

Tracker, 29

Version Manager, 29

Quintessential CD v1.27, 38

Randvoorwaarden, 27

RCS, xi

requirements, 4

Resultaat

Definitieve, 6

Risico's, 27

SAIS, 28

Samenvatting, vii, 21

Stempel, 25

Summary, ix

UML, xi, 29

URD, xii, 28

Verklarende woordenlijst, xi

vim, 38

Vingers, 25

Voorwoord, iii

waterval model, 3

XML, vii, xii, 7, 10

XSLT, xii, 10

zsh, 38