“Ipulse Hardware Manager”

assortimentbeheersysteem ter ondersteuning
van het orderverwerkingsproces



Afstudeerverslag

Afdrukdatum:
9 juni 2004

Afstudeerperiode:
9 februari 2004 t/m 11 juni 2004

Afstudeerbedrijf:
Interpulse Automatisering B.V.

Bedrijfsmentoren:
Dhr. A. Meester
Dhr. M. Ghaoui

Examinatoren:
Dhr. G.J. de Graaf
Mevr. A.M.J.J. Lousberg-Orbons

Differentiatie:
Informatie Voorziening en Informatie Technologie (IVIT) richting Ontwikkeling van Software en Technische Infrastructuren (OSTI).

Auteur:
A.J. Maagdelijn

Referaat

Maagdelijn, A.J.

Afstuderen
Ontwerp en implementatie van een assortimentbeheersysteem ter ondersteuning van het orderverwerkingsproces in opdracht van Interpulse Automatisering B.V..

Dit rapport beschrijft de werkzaamheden uitgevoerd in het kader van het afstuderen. Dit project is uitgevoerd in het kader van de opleiding Informatica en Informatiekunde aan de Haagse Hogeschool.

DESCRIPTOREN

Interpulse Automatisering B.V.
Iterative Application Design (IAD)
Extensible Markup Language (XML)
Borland Delphi
King
Tech Data

Voorwoord

Voor u ligt het verslag van Arjan Maagdelijn. Dit document is een tekstuele beschrijving van de uitgevoerde werkzaamheden bij de ontwikkeling van de Ipulse Hardware Manager. Tijdens het project heb ik de op school geleerde theorie in de praktijk kunnen brengen en me kunnen verdiepen in nieuwe technieken.

Ten eerste wil ik een paar mensen bedanken voor de totstandkoming van dit verslag. Ik heb met erg veel plezier aan het project bij Interpulse Automatisering B.V. te Leiden gewerkt. Ik wil daarom alle medewerkers van Interpulse bedanken voor hun medewerking. In het bijzonder wil ik Mansour Ghaoui en Arjan Meester voor de goede begeleiding bedanken. Ik heb een hoop van ze geleerd. Ik kijk zeker met plezier terug op deze afstudeerperiode bij Interpulse.

Daarnaast wil ik graag mijn dankwoord uitspreken naar de begeleiding vanuit school. Mijn dank gaat hierbij uit naar mevrouw Lousberg-Orbons en de heer de Graaf. Ze hebben me erg geholpen met hun kritische beoordeling van mijn conceptverslag. Met deze beoordeling heb ik mijn afstudeerverslag tot een hoger niveau kunnen bijstellen.

Arjan Maagdelijn
8 juni 2004, Leiden

Inhoudsopgave

1. INLEIDING 1

2. ORIËNTATIE 2

2.1 Organisatie 2

2.2 Opdracht Ipulse Hardware Manager 3
2.2.1 Aanleiding 3
2.2.2 Probleemstelling 3
2.2.3 Doelstelling 4

2.3 Methoden, technieken en software 4
2.3.1 IAD 4
2.3.2 UML 6
2.3.3 Borland Delphi 7
2.3.4 SQL Server 2000 7
2.3.5 King 8

2.4 Werkzaamheden, planning en op te leveren producten 9
2.4.1 Te verrichten werkzaamheden 9
2.4.2 Planning 10
2.4.3 Op te leveren producten 10

2.5 Voorbereidende opdracht 11
2.5.1 Waarom dit project? 11
2.5.2 Opdracht 11
2.5.3 Uitgevoerde werkzaamheden 12
2.5.4 Leereffecten 13

3. DEFINITIESTUDIE 14

3.1 Uitvoeren pilotplan workshop 14

3.2 Analyseren huidige situatie 15

3.3 Opstellen systeemeisen 17

3.4 Opstellen systeemconcept 19
3.4.1 Klassendiagram 20
3.4.2 Use case diagram 21
3.4.3 GUI ontwerp 23

3.5 Opstellen pilotplan 24

3.6 Bouwen prototype 26

4. PILOTONTWIKKELING 28

4.1 Aanpak pilotontwikkeling 28
4.1.1 Pilotontwerp 28
4.1.2 Bouw 29
4.1.3 Beoordeling en test 29

4.2 Pilot 1: Database 30
4.2.1 Pilotontwikkeling 30
4.2.2 Bouw 37
4.2.3 Beschrijving iteratieslagen 38

4.3 Pilot 2: Gegevensimport uit Tech Data en King 39
4.3.1 Pilotontwerp 39
4.3.2 Bouw 45
4.3.3 Beoordeling en test 47
4.3.4 Beschrijving iteratieslagen 49

4.4 Pilot 3: Overzicht en beheer 49
4.4.1 Pilotontwerp 50
4.4.2 Bouw 51
4.4.3 Beoordeling en test 54
4.4.4 Beschrijving iteratieslagen 55

4.5 Pilot 4: Export 55
4.5.1 Pilotontwerp 56
4.5.2 Bouw 57
4.5.3 Beoordeling en test 57
4.5.4 Beschrijving iteratieslagen 57

4.6 Algehele en multi-user test 58

5. INVOERING 59

5.1 Acceptatietest 59

5.2 Overdracht en beheer 59

6. EVALUATIE 60

6.1 Productevaluatie 60
6.1.1 Plan van aanpak algemeen 60
6.1.2 Ontwerprapport en systeem Grafi Magazijn 60
6.1.3 Definitiestudie 61
6.1.4 Pilotontwerprapportages 61
6.1.5 Systeem Ipulse Hardware Manager 61

6.2 Procesevaluatie 62
6.2.1 Communicatie met opdrachtgever en eindgebruikers 62
6.2.2 Ontwikkelmethode en strategie 62
6.2.3 Gehanteerde technieken 63
6.2.4 Planning 63
6.2.5 Doelstellingen 64

VERKLARENDE WOORDENLIJST 65

LITERATUURLIJST 66

1. Inleiding

The ability to consolidate, standardize, and analyze corporate data provides enormous opportunity to increase the competitiveness of organizations.
bron: www.datamirror.com

Het combineren van gegevens uit verschillende databases heeft de toekomst. Steeds meer bedrijven stellen hun databases ter beschikking om geïnteresseerde gewenste gegevens te laten ophalen. De afstudeeropdracht omvat het ontwerp, de bouw en de invoering van een systeem, waarmee overzicht wordt verkregen in het artikelassortiment van verschillende leveranciers en uit verschillende systemen.

Het doel van dit verslag is u inzicht te geven in de uitgevoerde werkzaamheden tijdens de afstudeerperiode. Deze werkzaamheden worden in de volgende hoofdstukken beschreven:

In hoofdstuk 2 wordt de oriëntatiefase beschreven. In dit hoofdstuk wordt de aanloop naar het project beschreven. Zo kan in dit hoofdstuk informatie over de organisatie, afstudeeropdracht en voorbereidende opdracht worden teruggevonden.

In hoofdstuk 3 wordt de definitiefase beschreven. In dit hoofdstuk wordt het proces beschreven, waarmee de opdracht gedetailleerd is vastgesteld.

In hoofdstuk 4 wordt de pilotontwikkelfase beschreven. In deze fase is informatie terug te vinden over hoe het systeem is ontwikkeld, gebouwd en getest.

In hoofdstuk 5 worden de activiteiten die in de invoeringsfase zijn uitgevoerd beschreven. Hierbij staat de overdracht en de acceptatietest centraal.

In het laatste hoofdstuk wordt de evaluatie op de afstudeeropdracht beschreven. Hierbij worden de opgeleverde producten en het uitgevoerde proces geëvalueerd.

2. Oriëntatie

In dit hoofdstuk is informatie te vinden over de organisatie en opdracht. Als eerste wordt een korte schets gegeven van de organisatie, waarbinnen de opdracht is uitgevoerd. Vervolgens zal de opdracht worden beschreven.

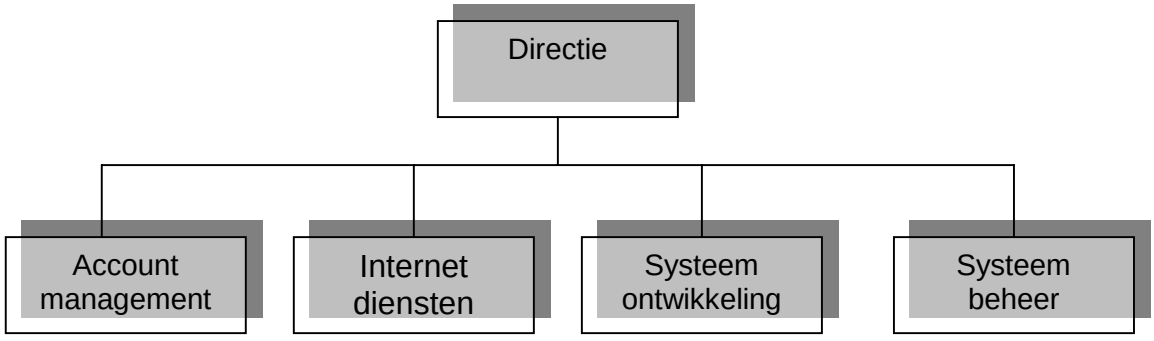
2.1 Organisatie

Interpulse Automatisering BV is een jong en dynamisch bedrijf, gevestigd in een oud Leids pand op de Hoge Rijndijk 42. Het bedrijf is eind 1999 opgericht als VOF (Vennootschap Onder Firma) door een zevental enthousiaste IT-ers. In 2001 zijn zij verder gegaan als BV (Besloten Vennootschap). In totaal zijn er op dit moment dertien mensen werkzaam bij Interpulse.

Interpulse Automatisering B.V. is een full-service dienstverlener op het gebied van Informatie & Communicatie Technologie (ICT). Interpulse biedt haar diensten vooral aan in de Leidse regio en richt zich primair op middelgrote bedrijven. De kracht van Interpulse ligt op het aanbieden van een zeer uitgebreid pakket aan diensten (advies en beheer) en systemen (software en hardware). Hierdoor hoeft de klant voor automatisering niet meerdere bedrijven aan te spreken, maar wordt Interpulse gebruikt als vast aanspreekpunt.

Interpulse houdt zich bezig met installatie, reparatie en configuratie van soft- en hardware. Tevens kan Interpulse een ander bedrijf van dienst zijn met integratie en implementatie van software- en hardwareoplossingen. De diensten die Interpulse aanbiedt kunnen globaal worden verdeeld in de volgende diensten: Internet diensten, Systeembeheer, Systeemontwikkeling en Consultancy. Interpulse levert ook hard- en software, maar dat is vooral in combinatie met een dienst van de overige afdelingen.

De organisatie structuur ziet er als volgt uit. Allereerst is er de directie, deze bestaat uit drie oprichters van de BV. Onder de directie zitten vier afdelingen: Accountmanagement, Internetdiensten, Systeemontwikkeling en Systeembeheer.



Figuur: Organogram Interpulse Automatisering B.V.

Accountmanagement:
Het accountmanagement communiceert met klanten en maken de wensen van de klant duidelijk aan de desbetreffende afdeling. De accountmanagers fungeren dus als tussenpersoon tussen de klant en de afdeling die de wens van de klant realiseert. Het accountmanagement is drie man groot.

Internetdiensten:
Internetdiensten, bestaande uit een drietal medewerkers, verzorgt het ontwerp en de ontwikkeling van complete websites en content management systemen (CMS). Daarnaast biedt de afdeling degelijke hosting faciliteiten en snelle domeinregistratie.

Systeemontwikkeling:
Ik ben tijdens de afstudeerperiode werkzaam geweest op deze afdeling. De afdeling is drie man groot. Systeemontwikkeling ontwikkelt voornamelijk applicaties, zoals informatie systemen en aansturingsystemen. Tevens kan Systeemontwikkeling de afdeling Internetdiensten de helpende hand bieden bij complexe aspecten van webapplicaties.

Systeem Beheer:
De afdeling Systeem beheer zorgt er voor dat werkstations en servers van klanten blijven werken. Zij geven ook advies over netwerken, benodigde hardware en software. Deze afdeling is vier man groot.

2.2 Opdracht Ipulse Hardware Manager

In deze paragraaf wordt de opdracht beschreven. Het doel van deze paragraaf is inzicht te geven in de inhoud van de opdracht.

2.2.1 Aanleiding

Interpulse Automatisering B.V. is een full-service automatiseerder. Naast software en diensten levert Interpulse ook hardware aan klanten. Interpulse is zelf geen fabrikant, importeur of groothandel op het gebied van hardware, maar fungeert als tussenpersoon tussen leverancier en klant. Interpulse verkrijgt informatie over hardwareproducten via de websites van de leveranciers. De belangrijkste leveranciers zijn Tech Data en Copaco.

2.2.2 Probleemstelling

Om klanten een actueel overzicht te kunnen geven en zelf inzicht te hebben in de huidige hard- en softwareprijzen, moet er handmatig zoekwerk worden verricht op de websites van de leveranciers. De gevonden artikelen worden vervolgens handmatig op een offerte geplaatst en na goedkeuring van de klant in het boekhoudsysteem ingevoerd. Het prijsverschil tussen aankoop en verkoop van hardware is niet erg groot. De kleine winstmarge die op de verkoop wordt geboekt is op dit moment nog lager, omdat er geld wordt verloren aan manuren die nodig zijn voor het opzoeken van de artikelen.

Daarnaast is het om strategische redenen van belang dat Interpulse Automatisering B.V. op de hoogte is van de laagste prijzen van de hardwareproducten. Op dit moment is het lastig om op prijsdalingen van hardwareproducten in te springen. In de huidige situatie wordt een prijsdaling min of meer toevallig opgemerkt. In veel gevallen gaat een prijsdaling aan Interpulse voorbij. Het is van belang dat Interpulse de concurrentie kan voorblijven door op de hoogte te zijn van de laatste prijzen en op die manier de klanten een voordelig hardwarepakket kan aanbieden.

2.2.3 Doelstelling

Het doel van de opdracht is het inzicht in het artikelassortiment van de leveranciers te vergroten. Inzicht in actuele prijzen en het prijsverloop van de geschiedenis per artikel is hierbij van belang. Een ander doel is dat de handmatige handelingen van het zoeken op websites en het invoeren van orders en artikelen op offertes en in het boekhoudsysteem worden weggenomen.

Deze doelstelling zal bereikt worden door een tool, genaamd Ipulse Hardware Manager, te ontwikkelen. Deze tool moet beschikken over de volgende eigenschappen:

- Importmogelijkheid voor het assortiment van leveranciers voor o.a. prijsinformatie.
- Importmogelijkheid van het boekhoudsysteem om de Interpulse voorraadartikelen te herkennen.
- Beheer-, zoek- en overzichtmogelijkheid in het artikelassortiment van de leveranciers en uit het boekhoudsysteem.
- Mogelijkheid tot orderverwerking.
- Exportmogelijkheid voor orders en artikelen in xml-formaat. Het xml-bestand kan worden geïmporteerd in het boekhoudsysteem.

2.3 Methoden, technieken en software

In deze paragraaf worden de belangrijkste methoden, technieken en softwarepakketten beschreven die gedurende het project zijn gebruikt. Het doel van deze beschrijving is u inzicht te geven wat de gebruikte methoden, technieken en software inhouden en waarom er voor deze gekozen is.

2.3.1 IAD

In de aanloop tot de opdracht heb ik met Interpulse gesproken over de te hanteren ontwikkelmethode. Interpulse liet mij hier vrij in, maar had een wens voor IAD. Binnen Interpulse wordt geen gebruik gemaakt van een echte ontwikkelmethodiek. Op dit moment wordt een opdracht door een accountmanager mondeling op de programmeur overgebracht. Deze maakt de benodigde detailspecificaties (meestal in UML) en bouwt het gewenste systeem. De gedurende het project door mij opgedane kennis over IAD wil Interpulse in de toekomst gaan gebruiken voor een eigen invulling van een IAD ontwikkeltraject.

Ik heb daarom besloten deze methodiek te gebruiken. Zelf heb ik in het verleden alleen nog met SDM 1 gewerkt, waardoor de ontwikkelmethode nieuw voor mij was. In de beginfase heb ik onderzoek naar IAD gedaan. Ik heb dit gedaan via een literatuurstudie.

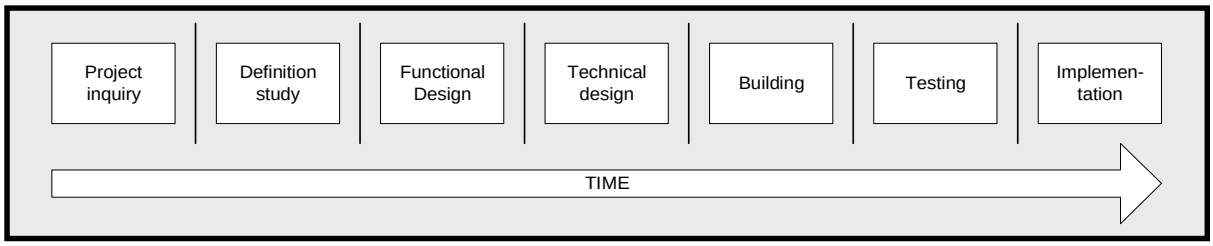
Iterative Application Development (IAD) is een uitgebreide set van activiteiten om een systeem te ontwikkelen. Iterative, vertaalt iteratief, wil zeggen dat de ontwikkelfasen verschillende malen doorlopen worden. Puntsgewijs ziet een ontwikkeltraject volgens IAD er als volgt uit:

- Definitiestudie
- Pilotontwikkeling
- Invoering

IAD versus SDM 1

Zoals eerder beschreven heb ik in het verleden slechts gebruik gemaakt van de ontwikkelmethodiek SDM 1, nader SDM te noemen. Om te onderzoeken op welke plaats IAD afwijkt van SDM heb ik de twee methodieken met elkaar vergeleken.

Het grootste verschil tussen IAD en SDM is dat IAD iteratief is en SDM lineair (waterval). Dit betekent dat het te bouwen systeem met IAD middels meerdere iteraties wordt opgebouwd. Steeds wordt een gedeelte van het systeem ontwikkeld en geïmplementeerd. Met SDM wordt een traject gevolgd zoals in onderstaand model is aangegeven.



Figuur: SDM ontwikkeltraject

SDM heeft een aantal voordelen en nadelen. Hieronder worden de voor- en nadelen van deze methodiek opgesomd:

Voordelen:

- Logisch geordende volgorde van taken
- Producten komen door stapsgewijze verfijning tot stand

Nadelen:

- Doordat alle eisen in het begin vastgesteld moeten worden is er weinig ruimte om deze lijst te herzien of aan te vullen
- Het lang uitblijven van resultaten kan het vertrouwen van de opdrachtgever ondermijnen

Hieronder worden de voor- en nadelen van IAD opgesomd:

Voordelen:

- De opdrachtgever / eindgebruiker wordt nauw bij het ontwikkeltraject betrokken. Waardoor de eisen in een vroeg stadium kunnen worden bijgesteld.
- Snelle resultaten.

Nadelen:

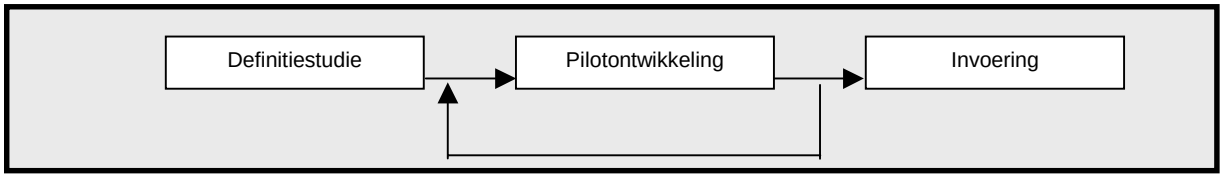
- Scope creeping. Software is nooit af. Wanneer de systeemeisen steeds worden bijgesteld loopt het project gevaar nooit afgerond te worden.

- Veel tussentijdse versies kan leiden tot problemen of verliezen bij slechte toepassing van versiebeheer.

Vooral het nauw betrekken van de opdrachtgever spreekt mij aan. In projecten, waarbij SDM als methodiek wordt gebruikt, is deze betrokkenheid van de opdrachtgever vaak minder, waardoor het kan voor komen dat het eindresultaat niet is wat de opdrachtgever voor ogen had. Dit probleem is hiermee opgelost.

Strategiekeuze

Binnen IAD zijn er verschillende strategieën, waaruit de ontwikkelaar kan kiezen. Ik heb in overleg met de opdrachtgever gekozen voor de strategie ‘incrementeel ontwikkelen’ (RAD). Dit is een strategie, waarin eerst de systeemeisen in de definitiefase volledig in kaart worden gebracht. In de tweede fase wordt het systeem d.m.v. een aantal iteraties van de pilotontwikkelfase gerealiseerd. De invoering vindt vervolgens in één keer plaats.



Figuur: IAD ontwikkeltraject Incrementeel ontwikkelen

Ik heb voor deze strategie gekozen, omdat vooraf het systeem goed kon worden gespecificeerd. Daarom was het niet nodig om voor elke pilot opnieuw de definitiefase te doorlopen. Met de opdrachtgever was afgesproken dat het systeem in één keer wordt opgeleverd. Het heeft hierdoor geen zin de invoeringsfase in het iteratieve traject te betrekken. Het iteratief doorlopen van de pilotontwikkelfase is naar mijn mening wel zinvol. Aparte systeemdelen worden in dezelfde iteratie ontworpen en gebouwd. Hierdoor wordt steeds aan één aparte bouwsteen gewerkt. Ik verwacht dat dit leidt tot betere software, omdat meteen na het ontwerp de bouwsteen wordt gerealiseerd i.p.v. dat eerst moet worden nagedacht over een andere bouwsteen.

2.3.2 UML

Binnen Interpulse wordt UML als standaard modelleringtechniek gebruikt. De opdrachtgever wenste daarom dat ik deze techniek tijdens mijn project zou gebruiken. Hiermee wordt de overdraagbaarheid vergroot. Ik ben met de wens van de opdrachtgever akkoord gegaan en heb UML als modelleringtechniek gebruikt.

UML staat voor Unified Modelling Language. UML is te gebruiken in combinatie met de ontwikkelmethode IAD. Een IAD traject is vaak iteratief. Bij een iteratief proces kunnen in een laat stadium van het ontwikkeltraject extra functionaliteiten worden toegevoegd of kan het systeemconcept worden aangepast. Bij klassieke modelleringtechnieken, zoals Yourdon, kan dit problemen opleveren. Als een wijziging moet worden doorgevoerd moeten vaak alle diagrammen uit een klassieke modelleringtechniek worden herschreven. Dit brengt veel werk met zich mee. Het aanpassen van een diagram in UML gaat vaak veel sneller dan bij een klassieke modelleringtechniek. Dit omdat diagrammen niet geheel opnieuw geschreven dienen te worden. Vaak is een kleine toevoeging aan het diagram voldoende.

Aan het begin van de opdracht was mijn UML kennis niet toereikend om de opdracht uit te kunnen voeren. Vanuit school ben ik slechts bij één module in aanraking gekomen met deze techniek. Mijn UML kennis heb ik verbeterd d.m.v. een literatuuronderzoek en een voorbereidende opdracht, waarin UML in de ontwerpfase gebruikt is.

In de ontwerpfase van de applicatie kan met UML een groot aantal diagrammen gebruikt worden. UML laat de ontwikkelaar vrij in het kiezen van de te gebruiken diagrammen. De meest gebruikte diagrammen zijn:

- Use case diagram. Dit diagram toont hoe het systeem gebruikt kan worden door externe entiteiten zoals menselijke gebruikers.
- Klassediagram. Toont de statische structuur van het softwaresysteem, weergegeven als klassen en relaties.
- Sequencediagram. Toont de volgorde in tijd van de boodschappen die in het systeem verstuurd en ontvangen worden.

2.3.3 Borland Delphi

Borland Delphi is een programmeeromgeving die in veel bedrijven wordt gebruikt. Voor de opdracht wordt gebruik gemaakt van Delphi, omdat dit de standaard ontwikkelomgeving bij Interpulse is. Interpulse maakt gebruik van Delphi 6.

Delphi is gebaseerd op de programmeertaal Borland Object Pascal. Object Pascal houdt in dat het een objectgeoriënteerde taal is. Delphi is een visuele ontwikkelomgeving. Dat wil zeggen dat het ontwerpen van de applicatie door middel van het drag-en-drop principe gaat. Standaard zijn er al een aantal componenten beschikbaar. Deze zijn opgeslagen in de Visual Component Library (VCL). Een component kan bijvoorbeeld een knop of een tekstveld zijn. De VCL kan worden aangevuld met nieuwe componenten. Nieuwe componenten kunnen zelf worden gemaakt of via externe bronnen worden geïnstalleerd.

Ik heb zelf al eerder ervaring opgedaan met deze ontwikkelomgeving. Borland Delphi is de enige object georiënteerde ontwikkelomgeving, waarin ik eerder heb ontwikkeld.

2.3.4 SQL Server 2000

Om de database te realiseren is gebruik gemaakt van SQL Server 2000. Hieronder motiveer ik mijn keuze voor deze DBMS.

In de beginfase heb ik onderzocht welk type database het beste geschikt zou zijn voor het project. Via een kort onderzoek naar XML kwam ik de term ‘XML database’ tegen. Aangezien bij de importfunctie gebruik wordt gemaakt van XML bestanden zou dit type database misschien van toepassing kunnen zijn op mijn project. Ik heb onderzocht of dit type database of de door mij bekende relationele databases geschikt is voor mijn project. Het onderzoek heb ik gedaan d.m.v. het raadplegen van een aantal websites, waarop XML databases beschreven worden.

Een XML database slaat in tegenstelling tot relationele databases geen data, maar documenten op. De structuur van de XML documenten blijft hierbij bewaart. Op de

door mij bekeken websites werd vooral discussie gevoerd of een XML database nuttig is. Ik kwam hierbij de volgende uitspraak tegen:

XML databases are ideally suited to the management of document-oriented information (unstructured or semi-structured data), they don't make sense for use for structured data.
Kevin Williams, Blue Oxide Technologies, 2001

Uit deze uitspreek lees ik af dat gestructureerde data moeilijk terug te vinden is binnen XML databases. Het terugvinden van gestructureerde data is echter belangrijk voor mijn project. Na het onderzoek heb ik een aantal essentiële nadelen aan XML kunnen ontdekken.

- In het huidige project worden niet alleen XML documenten gebruikt, maar ook andere importformaten. Hierdoor dienen extra conversiehandelingen uitgevoerd te worden.

- Mijn kennis over XML databases is niet voldoende. Het onderzoek naar XML databases zal tot tijdverlies leiden.

- Binnen Interpulse zijn er geen programma’s voorhanden, waarmee een XML database kan worden opgebouwd.

Het tweede en het derde nadeel van XML databases hebben de doorslag gegeven niet voor dit type database te kiezen. In tegenstelling tot xml databases ben ik wel bekend met relationele databases. Ik heb daarom voor een relationele database gekozen.

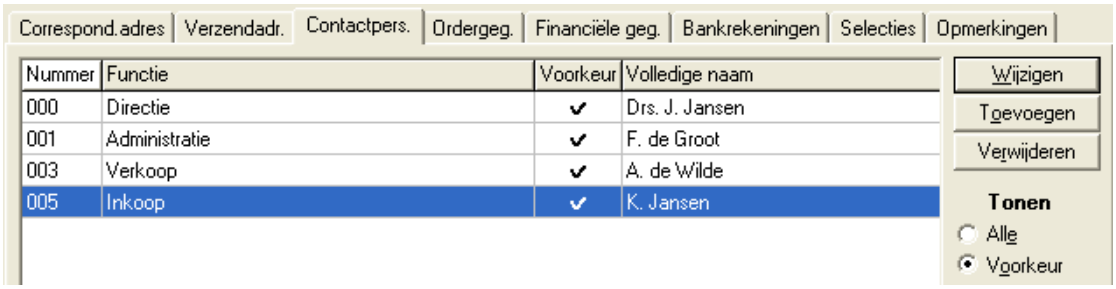
Binnen Interpulse zijn er twee database management systemen voorhanden. Om geen extra kosten aan een nieuw databasepakket te maken heb ik een keuze gemaakt uit deze pakketten. De aanwezige database management systemen zijn Microsoft Acces en Microsoft SQL Server 2000. Ik heb in het verleden al vaak met Acces gewerkt. Dit pakket is naar mijn mening relatief simpel in gebruik, maar is erg traag bij veel records. Microsoft SQL Server heeft hoge performance (ook bij veel records) als groot voordeel. Daarnaast wordt dit pakket veel gebruikt binnen Interpulse, waardoor het kennisniveau binnen Interpulse goed te noemen is.
Deze voordelen hebben de doorslag voor de keuze voor Microsoft SQL Server gegeven.

2.3.5 King

Een pakket waar gedurende het project veel gebruik van gemaakt wordt is King. De fabrikant van King is Quadrant Software BV. King is een standaard pakket voor de financiële en logistieke automatisering, gericht op het midden- en kleinbedrijf. Binnen dit boekhoudpakket kunnen een groot aantal functionaliteiten worden gebruikt zoals: orderverwerking, voorraadbeheer, budgetteren en het beheren van de klantenadministratie.

Belangrijk voor de opdracht zijn orders en artikelen uit King. Een order kan in King verschillende gedaantes aannemen. Een order in King kan worden verwerkt tot orderbevestiging, pakbon en factuur. In King zijn alleen de artikelen opgenomen die eerder in een order zijn verwerkt. Om King overzichtelijk te houden mag niet het hele assortiment van de leveranciers in King worden opgenomen.

In King kunnen handmatig gegevens worden toegevoegd. Naast het handmatig toevoegen zijn er voor de stamgegevens imports mogelijk. Artikelen, klanten en orders behoren tot deze stamgegevens. Het importbestand dient bij de King import een XML bestand met een door King vastgestelde structuur te zijn. Deze structuur is te vinden in de King helpfile. Daarnaast kunnen stamgegevens, zoals klanten, orders en artikelen, worden geëxporteerd. Ook het exportbestand is een XML bestand.
Op het gebied van gebruiksvriendelijkheid is in King veel gedaan. Hierbij zijn de duidelijke en consistente schermopbouw en de uitvoerige gebruiksdocumentatie de belangrijkste voorbeelden. Hieronder wordt hier een voorbeeld van getoond. Duidelijk te zien is het gebruik van tabbladen en de functieknoppen aan de rechterkant van het scherm.



Figuur: Schermopbouw in King

Meer informatie over King is te vinden via www.quadrant.nl.

2.4 Werkzaamheden, planning en op te leveren producten

In deze paragraaf worden de werkzaamheden beschreven die worden uitgevoerd tijdens de opdracht. Het doel hiervan is een tijdschatting voor het project te kunnen maken.

2.4.1 Te verrichten werkzaamheden

Hieronder worden de te verrichten werkzaamheden beschreven. Ik ben tot deze werkzaamheden gekomen aan de hand van voorstudie naar IAD. Per fase uit IAD heb ik bekeken welke werkzaamheden van toepassing zijn voor de afstudeeropdracht. Dit zijn de volgende werkzaamheden:

- Oriëntatiefase
  - Opstellen plan van aanpak algemeen
  - Voorstudie Microsoft SQL Server 2000
  - Voorstudie King boekhoudsysteem
  - Voorstudie UML / IAD
  - Oefenproject Grafi Magazijn

- Definitiestudie
  - Opstellen plan van aanpak definitiestudie
  - Vaststellen huidige situatie
  - Opstellen systeemeisen
  - Opstellen systeemeisen lijst
  - Opstellen systeemconcept
  - Opstellen omschrijving technische structuur
  - Opstellen pilotplan

- Pilotontwikkeling (iteratief)
  - Opstellen Pilotontwerp
  - Bouw pilot
  - Test pilot

- Invoering
  - Maken installatiebestand met InstallShield
  - Acceptatietest
  - Implementatie Ipulse Hardware Manager bij eindgebruiker

2.4.2 Planning

Aan de hand van de IAD fasering ben ik in staat geweest een overall planning te maken. Ik heb dit gedaan d.m.v. het onderstaande GANTT diagram.

Week / Onderdeel	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Inventarisatie	■	■	■															
Definitiestudie			■	■	■													
Pilotontwikkeling (iteratief)					■	■	■	■	■	■	■	■	■	■	■	■	■	■
Pilot 1: Database					■													■
Pilot 2: Importfunctie						■	■	■	■	■	■							■
Pilot 3: Beheer en overzicht											■	■	■					■
Pilot 4: Exportfunctie													■	■	■	■	■	■
Invoering																		■
Overleg met opdrachtgever	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Afstudeerverslag en voorbereiding presentatie	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■

Figuur: GANTT diagram overall planning

2.4.3 Op te leveren producten

Tijdens het project worden verschillende mijlpaalproducten opgeleverd. Deze producten zijn mede verkregen door het bestuderen van IAD literatuur. De mijlpaalproducten zijn:

- Plan van aanpak algemeen
- Ontwerprapport Grafi Magazijn
- Systeem Grafi Magazijn
- Definitiestudie
- Prototype(s)
- Pilotontwerp (per pilot)
- Systeem Ipulse Hardware Manager

2.5 Voorbereidende opdracht

Dit hoofdstuk heeft als doel u inzicht te geven in het oefenproject Grafi Magazijn. Aangezien Grafi Magazijn een extra toegevoegd project is geef ik eerst een verantwoording waarom ik dit project heb uitgevoerd. Vervolgens wordt een samenvatting van de opdracht en de uitgevoerde activiteiten beschreven. Als laatste worden de leereffecten beschreven.

2.5.1 Waarom dit project?

De eerste weken als afstudeerder bij Interpulse heb ik gebruikt ter inventarisatie. In deze periode bood Interpulse mij aan een klein project uit te voeren. Met dit project zou ik in projectvorm kennis kunnen opdoen m.b.t. het werken binnen Interpulse. Het project heeft daarnaast de volgende toevoegingen aan de oriëntatiefase:

- Verbeteren van UML kennis. In de ontwerpfase van het project worden een aantal UML modelleertechnieken gebruikt.

- Uitbreiden van Delphi kennis. In het verleden heb ik met Delphi gewerkt. Dit was echter Delphi 5. Bij Interpulse wordt gebruik gemaakt van versie 6. Dit project kan helpen bij de overschakeling naar Delphi 6. Daarnaast wordt bij Interpulse gebruik gemaakt van andere Delphicomponenten dan ik in het verleden heb gebruikt. Het project kan dienen ter inventarisatie van deze componenten. Daarnaast heeft Interpulse een eigen codestandaard. Dit project kan dienen om mijn programmeurstijl aan te passen aan de Interpulse codeconventies.

- Iteratieve ontwikkeling. Ik heb in het verleden nog geen incrementeel project gevolgd. Met dit project zou ik in een korte tijd nuttige informatie kunnen opdoen over het incrementeel ontwikkelen.

- Het project heeft enige overlapping met oorspronkelijke afstudeeropdracht. Een aantal geprogrammeerde functies kan mogelijk worden hergebruikt in de afstudeeropdracht.

Ik denk dat een kort project effectiever is dan het bestuderen van literatuur. Ik heb de opdracht aangenomen met de eis dat de doorlooptijd van het project maximaal 2 weken zal zijn. Hierdoor blijft er voldoende tijd over voor de oorspronkelijke afstudeeropdracht.

2.5.2 Opdracht

De opdrachtgever voor de opdracht is Grafi-Call. Grafi-Call is een groothandel in computer en computeronderdelen en levert computeronderdelen aan particulieren en bedrijven. De opdracht betreft het bouwen van een systeem, waarmee Grafi-Call snel een artikel kan terugvinden in haar magazijn. Op dit moment is er namelijk geen overzicht over de producten in het magazijn.

Er dienen twee applicaties worden opgeleverd. De eerste applicatie is een beheerapplicatie. Deze applicatie wordt gebruikt om artikelen toe te voegen, wijzigen of verwijderen. Aan een artikel kan een plaats in het magazijn worden toegekend.

Naast de beheerapplicatie is er ook een opvraagapplicatie. Deze applicatie wordt geplaatst in het magazijn en wordt gebruikt om snel een artikel terug te vinden. Beide applicaties zijn gekoppeld aan dezelfde database. De database wordt op een plaats in het Grafi-Call netwerk geplaatst, die door beheer- en opvraagapplicatie bereikt kan worden.

2.5.3 Uitgevoerde werkzaamheden

Om de opdracht te realiseren is er kort een ontwerpfase doorlopen. Deze ontwikkelfase begon ik met een interview met de Arjan Meester (accountmanager binnen Interpulse). Arjan fungeert tussenpersoon tussen mij en opdrachtgever. Het doel van dit interview was het vaststellen van de opdracht.

In het interview is in overleg besloten dat er een kort IAD traject wordt doorlopen. Hierbij worden alleen de essentiële onderdelen uitgewerkt. Dit om te voorkomen dat er teveel tijd aan deze kleine opdracht wordt besteed. Deze essentiële onderdelen zijn de onderdelen die Interpulse op dit moment ook uitwerkt en worden hieronder beschreven. Aan de hand van het interview is het ontwerprapport uitgewerkt. Vervolgens is een bouwfase en een invoeringsfase doorlopen. De volgende activiteiten zijn uitgevoerd:

Ontwerpfase
- Vaststellen en beschrijven functionele systeemeisen.
- Opstellen use cases.
- Opstellen database ontwerp (databasemodel).
- Opstellen applicatiestructuur.

Bouwfase
- Bouw prototype.
- Bouw database.
- Bouw/programmeren functies.
- Testen.
- Debuggen.

Invoering
- Maken installable.
- Uitvoeren acceptatietest.
- Implementatie applicaties bij eindgebruiker.

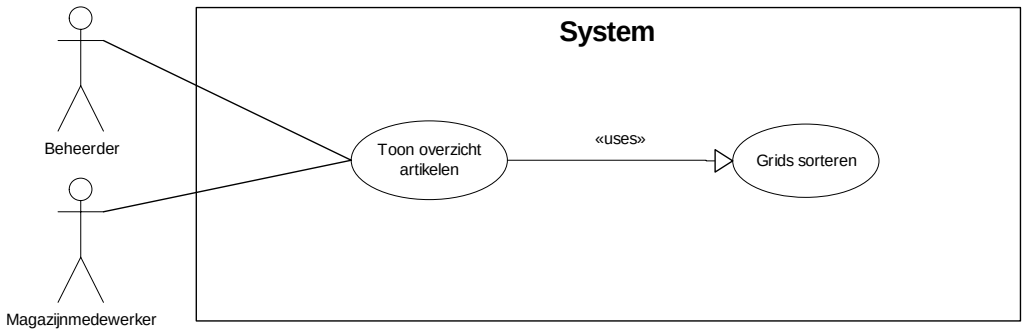
Aangezien de bovenstaande activiteiten ook worden uitgevoerd in de afstudeeropdracht, zal ik de uitwerking van deze activiteiten hier niet beschrijven. Het uitvoeren van bovenstaande activiteiten wordt uitvoerig (met uitgewerkte modellen) in de komende hoofdstukken voor de afstudeeropdracht beschreven. De in de ontwerpfase van dit project uitgewerkte modellen kunnen teruggevonden worden in de externe bijlage “ontwerprapport Grafi Magazijn”.

Het doel van het project is gehaald. Het systeem is bij de opdrachtgever geïmplementeerd.

2.5.4 Leereffecten

Het project Grafi Magazijn was voor mij een leerproject. Ik zal hieronder per punt de leereffecten beschrijven.

Als eerste heb ik mijn UML kennis verbeterd. Zoals eerder beschreven heb ik in projectvorm nog geen gebruik gemaakt van UML. In dit project heb ik gebruik gemaakt van een use case diagram. Hierbij werd het nut van dit model me pas echt duidelijk. Met het use case diagram kan snel worden gezien of de verschillende applicaties overlappende functionaliteiten hebben. Hiermee kan in de bouwfase rekening worden gehouden, zodat een functionaliteit niet tweemaal worden gebouwd. In onderstaande fragment is dit goed te zien.



Figuur: Fragment uit use case diagram Grafi Magazijn

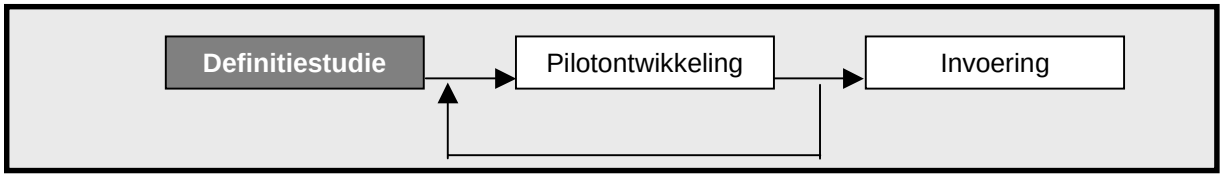
Naast het verbeteren van mijn UML kennis heb ik met dit project mijn Delphi kennis verbeterd. Mijn stagebegeleider Mansour Ghaoui heeft me een aantal programmeerconcepten uitgelegd, die ik zeker ook in de toekomst kan gebruiken. Het gebruik van bijvoorbeeld het “try – except” om foutmeldingen af te vangen heb ik als erg nuttig ervaren. Met de overstap van Delphi 5 naar Delphi 6 heb ik geen problemen ondervonden. Het verschil voor de programmeur is nauwelijks merkbaar.

Ook heb ik met dit project meer inzicht gekregen in incrementele ontwikkeling. Een nadeel wat ik hierbij heb ondervonden is, dat in een zeer laat stadium nog functies kunnen worden toegevoegd. Ik dacht een aantal keer het systeem af te hebben, maar vervolgens werd er alsnog een nieuwe functie door de opdrachtgever toegevoegd. Ik zal de functionaliteiten in het vervolg beter vastleggen, zodat het bestaande concept geen grote veranderingen ondervindt.

Als laatste punt heeft dit project nut gehad voor mijn afstudeeropdracht op het gebied van codehergebruik. Tijdens het project heb ik een aantal globale functies gebouwd, welke ook kunnen worden hergebruikt bij de afstudeeropdracht. Een goed voorbeeld hiervan is de sorteringfunctie voor overzichten.

3. Definitiestudie

Zoals eerder beschreven heb ik voor de ontwikkelmethode IAD gekozen om het eindproduct te kunnen realiseren. De eerste fase uit IAD is de definitiestudie. In de fase definitiestudie worden de doelen van het systeem geanalyseerd. Dit in nauwe samenwerking met de eindgebruiker. De definitiestudie wordt slechts eenmaal uitgevoerd.



Figuur: IAD traject

IAD is geschikt voor grote en kleinere projecten. Hierdoor kan het voor komen dat IAD een aantal onderdelen in de definitiestudie beschrijft die overbodig zijn voor mijn project. Aangezien ik nog niet eerder een IAD definitiestudiefase heb doorlopen, heb ik eerst bestudeerd welke onderdelen uit de definitiestudie van toepassing zijn op de afstudeeropdracht. Dit onderzoek is gedaan d.m.v. het lezen van IAD literatuur. Hier zijn de volgende onderdelen uit IAD uitgekomen:

- Plan van aanpak definitiestudie
- Pilotplan workshop
- Systeemeisen
- Systeemconcept
- Technische structuur
- Pilotplan
- Prototype

Een onderdeel dat ik aan dit lijstje heb toegevoegd is het beschrijven van de huidige situatie. Ik ben van mening dat dit onderdeel essentieel is om de opdracht goed te kunnen begrijpen. In de huidige situatie komen de problemen naar voren, waarop het te bouwen systeem een oplossing moet bieden. In de komende paragrafen worden bovenstaande onderdelen, op het plan van aanpak na, beschreven. Meer informatie over de definitiestudie kan gevonden worden in de externe bijlage “Ipulse Hardware Manager Definitiestudie”.

3.1 Uitvoeren pilotplan workshop

Het eerste onderdeel dat ik na het opstellen van het plan van aanpak heb uitgevoerd is het pilotplan workshop. Het doel van de workshop is het analyseren en detailleren van de opdracht.

In een IAD traject kan ook een ontwikkelscenario workshop uitgevoerd worden. In het ontwikkelscenario workshop wordt de ontwikkelaanpak besproken. Hierbij wordt de manier van opleveren (per pilot of alles tegelijk) besproken. Aangezien de tijd van de opdrachtgever beperkt is heb ik besloten het ontwikkelscenario – en pilotplan workshop samen te nemen.

Een workshop kan op verschillende manieren worden uitgevoerd. IAD beschrijft een workshop, waarin een aantal betrokkenen worden uitgenodigd. Deze betrokkenen denken mee over het te bouwen systeem. In het huidige project zijn er slechts twee betrokkenen, namelijk Arjan Meester, nader opdrachtgever genoemd, en mijzelf. Aangezien er maar twee betrokkenen zijn en de tijd van de opdrachtgever beperkt is, is er niet gekozen voor een echte workshop, waarin meerdere betrokkenen brainstormen en in discussie gaan over het te bouwen systeem. Ik heb in overleg met de opdrachtgever besloten de workshop te vervangen door interviews met de opdrachtgever. Om de opdrachtgever daarnaast nauw bij het project te betrekken is de afspraak gemaakt dat elk hoofdstuk uit de definitiestudie ter evaluatie kort met elkaar wordt besproken.

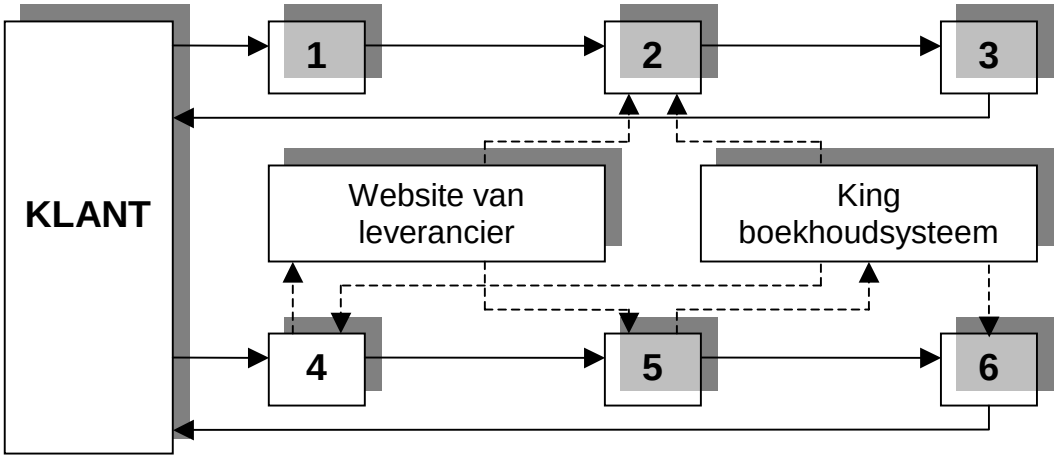
De interviews zijn voorbereid aan de hand van IAD literatuur en de eerste versie van de opdrachtomschrijving. Ik heb interviews gehouden om de huidige situatie te analyseren, de gewenste situatie te achterhalen en het te volgen ontwikkelscenario te overleggen.

De uit de interviews verkregen informatie is meteen na het interview in de definitiestudie verwerkt. De huidige en gewenste situatie wordt in de komende paragrafen beschreven. Het te volgen ontwikkelscenario is reeds beschreven bij de uitleg van de ontwikkelmethode IAD.

3.2 Analyseren huidige situatie

Zoals eerder beschreven is dit onderdeel uit de definitiestudie opgesteld aan de hand van een interview met de opdrachtgever. Het doel van het beschrijven van de huidige situatie is het vaststellen van de problemen, waarop de Ipulse Hardware Manager oplossing moet bieden.

Uit de opdrachtomschrijving is op te maken dat er problemen zijn in het verkrijgen van overzicht in het assortiment van de leveranciers. Dit probleem is betrekkelijk vaag. Tijdens het interview ben ik daarom dieper op de probleemstelling ingegaan. Uit het interview is naar voren gekomen dat het probleem ontstaat bij het verwerken van een offerteaanvraag van een klant tot een factuur. Met de verkregen informatie uit het interview heb ik, om voor mijzelf een duidelijk beeld te krijgen wat er gebeurt bij een artikelaanvraag van een klant, een schema opgesteld.



Figuur: Orderverwerking proces

De dichte pijlen geven de volgorde van de doorlopen stappen aan. De stippelpijlen geven aan welke bronnen worden geraadpleegd bij de stappen.

Stap 1: Offerteaanvraag

Een klant neemt contact met Interpulse op om een offerte te vragen voor door de klant gewenste hard- en software.

Stap 2: Opstellen offerte

Een medewerker van Interpulse maakt een offerte op in een daarvoor bestemde template in MS Word en Excel. Op de offerte zijn de gewenste artikelen opgenomen. Een door de klant gewenst artikel wordt opgezocht in King. Is het gewenste artikel niet in King aanwezig of ligt deze niet op voorraad bij Interpulse wordt de website van de leverancier voor prijsinformatie geraadpleegd. Hierbij moet rekening worden gehouden met een goede prijs – kwaliteit verhouding om zo goedkoop mogelijk voor de klant te zijn. Op de website van de leverancier wordt hier echter geen overzicht in gegeven. De belangrijkste leverancier is Tech Data. Hier plaatst Interpulse 95% van haar bestellingen. De overige bestellingen worden bij Copaco geplaatst.

Stap 3: Opsturen offerte

De offerte wordt afgedrukt en opgestuurd naar de klant.

Stap 4: Plaatsen bestelling bij leverancier

De klant heeft goedkeuring voor de offerte gegeven. Hierop wordt in King gecontroleerd of het artikel bij Interpulse op voorraad ligt. Ligt deze niet op voorraad wordt het gewenste artikel bij Tech Data besteld.

Stap 5: Invoeren artikelen in King

Bij ontvangst van de artikelen van de leverancier worden de artikelen op de ontvangstbon in het boekhoudsysteem King ingevoerd. Een order wordt in deze fase aangemaakt. Als een artikel nog niet in King is ingevoerd, wordt een op de website van de leverancier gevonden artikel handmatig in King ingevoerd. De artikelen worden toegekend aan een order.

Stap 6: Opsturen artikelen en pakbon

De artikelen op de order worden in King afgedrukt als pakbon. De door de klant bestelde artikelen worden samen met de pakbon naar de klant verstuurd. Als de pakbon door de klant wordt afgetekend wordt de order als factuur afgedrukt en naar de klant verstuurd. Na het betalen is de order afgehandeld.

Naast het orderverwerking proces ben ik tijdens het interview ingegaan op de gebruikersgroep. In dit onderdeel wordt o.a. het kennisniveau van de eindgebruikers beschreven. Dit vind ik een belangrijk punt, omdat hiermee kan worden bepaald hoe de usability van het systeem moet zijn. De eindgebruikers van de Ipulse Hardware Manager zijn de accountmanagers van Interpulse. Alle eindgebruikers uit de gebruikersgroep hebben informatica gestudeerd en zijn afgestudeerd aan de Haagse Hogeschool. Het kennisniveau betreffende het ontwikkelen van software is hierdoor goed te noemen. Door het goede computerkennisniveau heeft de usability van het systeem een lage prioriteit.

Als laatste punt is de probleemanalyse uitgewerkt. Het doel van dit onderdeel is het vaststellen van de problemen die het systeem gaat oplossen. Ik heb deze problemen opgemaakt door de eerder opgestelde probleemstelling middels het interview gedetailleerder vast te stellen. Daarnaast is de use case beschrijving geraadpleegd om probleempunten vast te stellen. De volgende problemen zijn naar voren gekomen:

Probleem 1: Geen overzicht in artikelen

Interpulse heeft op dit moment geen overzicht in het artikelassortiment van haar leveranciers. Hierdoor is het lastig om op prijsdalingen van artikelen in te springen. In de huidige situatie wordt een prijsdaling min of meer toevallig opgemerkt. In veel gevallen gaat een prijsdaling aan Interpulse voorbij. Dit probleem is gerelateerd aan stap 2 uit het bovenstaande model.

Probleem 2: Het opstellen van offertes is tijdrovend en foutgevoelig

Een artikel moet via de website van een leverancier worden opgezocht. Aangezien het aantal producten dat de leverancier aanbiedt zodanig groot is, kan het tijdrovend zijn om een gewenst artikel te vinden. Daarnaast kunnen makkelijk fouten worden gemaakt door het handmatig opstellen van deze offertes. Dit probleem is gerelateerd aan stap 2 uit het bovenstaande model.

Probleem 3: Handmatig toevoegen van artikelen en facturen aan King

Als een klant een artikel wenst dat niet is ingevoerd in de King database moet deze handmatig worden toegevoegd in King. Aangezien hardwareproducten snel verouderen komt het vaak voor dat nieuwe artikelen worden ingevoerd. Dit proces is tijdrovend en maakt het verkopen van artikelen minder rendabel. Ook het handmatig toevoegen van facturen, vooral in combinatie met probleem 2, kan erg tijdrovend zijn. Dit probleem is gerelateerd aan stap 5 uit het bovenstaande model.

Voor bovenstaande drie problemen dient een oplossing gevonden te worden. Bovenstaande problemen zijn als uitgangspunt gebruikt voor het opstellen van de systeemeisen en systeemconcept.

3.3 Opstellen systeemeisen

Na het vaststellen van de problemen ben ik in overleg met de opdrachtgever op zoek gegaan naar oplossingen. Dit doormiddel van de ‘think aloud’ techniek, waarbij beide aanwezigen worden getriggerd bij het aandragen van oplossingen. De volgende functionele eisen zijn in deze ontmoeting vastgesteld:

- De tool moet het gehele assortiment van de belangrijkste leveranciers bevatten. De belangrijkste leveranciers zijn Tech Data en Copaco. Deze leveranciers stellen per dag bestanden beschikbaar, waarmee klanten het gehele assortiment kunnen inzien. Deze bestanden zijn echter niet overzichtelijk. Met de tool worden deze bestanden geïmporteerd en moet overzicht worden verkregen in prijs- en voorraadinformatie bij de leverancier. De prijs en voorraad moet elke dag worden opgeslagen, zodat de prijs- en voorraadhistorie in grafiekvorm kan worden afgelezen. Eerder geïmporteerde artikelen worden niet opnieuw geïmporteerd. Geïmporteerde artikelen worden opgeslagen in een database.

- De reeds in King opgenomen artikelen moeten tevens in de tool kunnen worden opgenomen. In King is beschreven of een artikel op voorraad ligt bij Interpulse. Met deze import kan dit ook uit de tool worden afgelezen.

- In de tool moet een zoekfunctie voor artikelen zijn opgenomen. Hiermee kan een gewenst artikel worden opgezocht.

- Een order moet samengesteld kunnen worden. Een order bestaat uit orderregels en een klant. De klant is afkomstig uit King. King klanten moeten dus in de tool kunnen worden opgenomen.

- Stamgegevens (geïmporteerd van leverancier of uit King) moeten beheerd kunnen worden. Onder beheer valt het toevoegen, wijzigen en verwijderen van gegevens. De stamgegevens zijn:

    - Artikelen: Hardware- en softwareproducten.
    - Categorieën: Categorie van een artikel. Deze wordt gebruikt om artikelen te groeperen.
    - Fabrikanten: Fabrikanten van artikelen.
    - Leveranciers: Leveranciers van artikelen. Interpulse plaatst hier haar bestellingen.
    - Klanten: Klanten die artikelen bij Interpulse bestellen.
    - Orders: Bestelling van artikelen door een bepaalde klant.

- Nadat een order is aangemaakt moeten deze kunnen worden geëxporteerd naar King. Nog niet in King ingevoerde artikelen moeten tevens kunnen worden geëxporteerd.

- Het is wenselijk dat een order vanuit de tool direct bij de leverancier kan worden geplaatst.

Na het opstellen van de eisenlijst stond het prioriteren van de eisenlijst centraal. Het aangeven van prioriteiten is een belangrijke activiteit, omdat hiermee kan worden aangegeven welke onderdelen zeker in het systeem komen en welke onderdelen eventueel worden opgenomen. Tijdens het overleg heeft de opdrachtgever aangegeven welke eisen belangrijk waren en welke minder belangrijk. Ook ik heb dit aangegeven. In overleg met de opdrachtgever is vervolgens de volgende geprioriteerde lijst opgesteld:

Basis
- De tool moet gekoppeld zijn aan een database, waarin data kan worden opgeslagen en opgevraagd
- King artikelen kunnen worden geïmporteerd
- King klanten kunnen worden geïmporteerd
- Het Tech Data assortiment kan worden geïmporteerd
- Stamgegevens (artikelen, categorieën, leveranciers, fabrikanten, orders en klanten) worden overzichtelijk weergegeven
- Artikelen kunnen worden opgezocht
- Artikelen kunnen worden beheerd
- Artikelen kunnen worden geëxporteerd in XML formaat
- Orders kunnen worden beheerd
- Orders kunnen worden geëxporteerd in XML formaat
Comfort
- Actuele artikelinformatie van Tech Data per artikel kan worden opgehaald
- Categorieën kunnen worden beheerd

- Leveranciers kunnen worden beheerd
- Klanten kunnen worden geëxporteerd in XML formaat
Luxe
- Het Copaco assortiment kan worden geïmporteerd
- Orders kunnen rechtstreeks vanuit de tool bij Tech Data worden geplaatst
- Fabrikanten kunnen worden beheerd
- Klanten kunnen worden beheerd

Figuur: Geprioriteerde basis systeemeisen lijst

Hieruit kan worden opgemaakt dat de in de bovenstaande tabel beschreven basis-systeemeisen zeker in het systeem worden opgenomen. Opvallend uit bovenstaand diagram is dat het importeren van het Copaco assortiment als luxe eis wordt gezien. Er is tot deze keuze gekomen, omdat er waarschijnlijk binnen de afstudeerperiode niet genoeg tijd is om deze importfunctionaliteit te realiseren. Bij Tech Data wordt 95% van de bestellingen geplaatst. De rest wordt bij Copaco geplaatst. Aangezien Copaco een relatief laag percentage bestellingen heeft, heeft deze functionaliteit een lagere prioriteit dan de import van Tech Data gegevens.

Naast de bovenstaande basis systeemeisen zijn er in deze fase uit de definitiestudie in overleg ook eisen vastgesteld op het gebied van interface en performance. De interface moet de King look-and-feel hebben. Dit betekent dat stamgegevens middels tabbladen benaderd kunnen worden. Rechts zal een knoppenbalk zijn geplaatst. Links van de knoppenbalk zullen de gegevens worden getoond. De uitwerking hiervan kan later in dit hoofdstuk bij het prototype worden teruggevonden. Op het gebied van performance is de afspraak gemaakt dat de afzonderlijke imports niet langer dan 5 minuten mogen duren. Dit punt wordt bij de implementatie van de import uitgebreider beschreven. Daarnaast zijn operationele eisen vastgesteld. De opdrachtgever wenste dat de applicatie door meerdere gebruikers te benaderen is. De applicatie wordt dus een multi user applicatie. Om dit te realiseren zal een client server architectuur worden gebruikt.

De laatste eis die de opdrachtgever heeft gesteld is dat de opdracht met de middelen, die binnen Interpulse beschikbaar zijn, gerealiseerd moet worden. Er mogen geen extra kosten gemaakt worden.

Een eerder beschreven nadeel van IAD is scope creeping. Dit houdt in dat door het steeds bijstellen van systeemeisen een project het gevaar loopt niet afgrond te worden. Ik heb daarom met de opdrachtgever de afspraak gemaakt dat slechts kleine wijzigingen in de basis systeemeisen worden gerealiseerd. Grote nieuwe eisen zullen als luxe eis worden opgenomen. Een voorbeeld hiervan is de eis dat orders rechtstreeks kunnen worden geplaatst bij de leverancier. Deze eis is in een later stadium toegevoegd en dus als luxe eis opgenomen in de eisenlijst.

3.4 Opstellen systeemconcept

Het doel van het systeemconcept is het beschrijven van een oplossing voor de problemen, zoals beschreven in de huidige situatie. Het systeemconcept geeft een eerste omschrijving van het uiteindelijk te bouwen systeem.

Het invullen van het systeemconcept heb ik d.m.v. UML modellen gedaan. Ik heb hiervoor gekozen, omdat de UML modellen goed inspringen in het beschrijven van de functionaliteiten en het vastleggen van de structuur. De volgende onderdelen zijn in het systeemconcept uitgewerkt:

- Klassendiagram
- Use cases
- User Interface ontwerp

3.4.1 Klassendiagram

Het eerste model dat ik uitgewerkt heb is het klassendiagram. Het klassendiagram heeft als doel inzicht te geven in de statische structuur van het programma. Ik heb dit onderdeel in het ontwerp opgenomen, omdat het klassendiagram van dienst kan zijn bij het opstellen van het database ontwerp.

Om tot het klassendiagram te komen heb ik eerst globaal alle klassen beschreven. De objectklassen zijn verkregen door mogelijke klassen uit de systeemeisen lijst te noteren. Dit volgens de wijze zoals UML deze voorschrijft. De lijst van kandidaatklassen is vervolgens gefilterd, zodat alleen de klassen overbleven die in het klassendiagram worden opgenomen. Tijdens deze filtering viel bijvoorbeeld de kandidaatklasse ‘Tech Data’ af. Tech Data is geen klasse, maar een waarde uit de klasse leverancier.

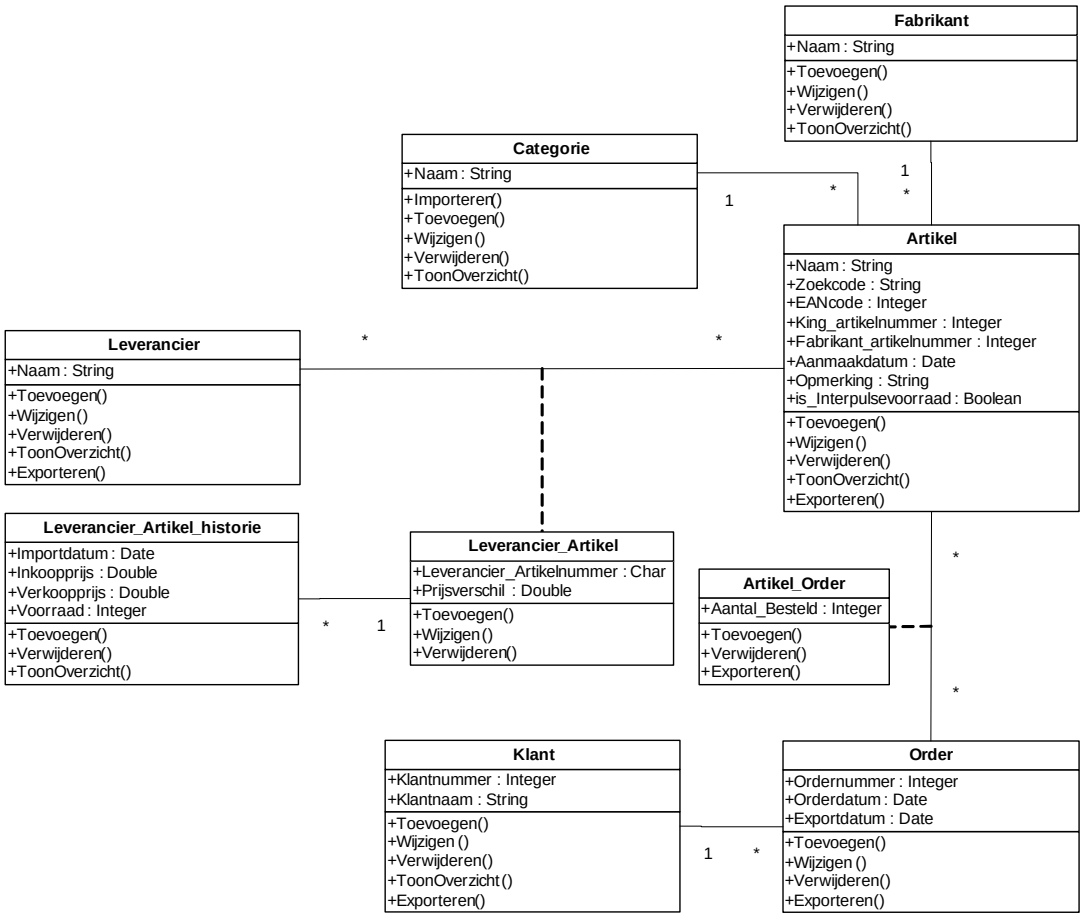
Naast de systeemeisen lijst heb ik ook gekeken naar het Tech Data importbestand om mogelijke klassen te herkennen.

152	HP	92222B	HP Cable/LDBL50 SCSI male>male 1m
642	CANON	0993A001	BJ-642/black 790sh f BJ300 330

Figuur: Fragment uit Tech Data importbestand

In bovenstaande figuur is een fragment uit het Tech Data importbestand te zien. Uit dit fragment kan een artikelnummer, fabrikant, fabrikant_artikelnummer en artikelnaam van twee artikelen worden afgelezen. Hiermee kon ik bijvoorbeeld de klassen artikel en fabrikant herkennen.

Na het opstellen van de klassen ben ik op zoek gegaan naar de attributen van de klassen. Hierbij heb ik vooral gekeken naar de importbestanden van Tech Data en King. Uit het eerder getoonde fragment van het Tech Data importbestand kan bijvoorbeeld worden afgelezen dat een artikel een artikelnummer, fabrikant_artikelnummer en een naam heeft. Volgens deze manier heb ik de klassen van attributen voorzien. Vervolgens zijn de associaties tussen de klassen aangemaakt. Hierbij werden de relaties tussen de klassen gelegd. Als laatste heb ik operaties aan klassen verbonden. Operaties zoals bijvoorbeeld toevoegen, wijzigen en verwijderen zijn opgemaakt uit de systeemeisen lijst. Het onderstaande klassendiagram is het resultaat.



Figuur: Klassendiagram

Het klassendiagram is gedurende de definitiefase door iteraties verschillende keren aangepast. Zo zijn extra attributen en klassen toegevoegd. Goede voorbeelden hiervan uit bovenstaand model zijn de klassen ‘Leverancier_artikel’ en ‘Leverancier_artikel_historie’. In eerste instantie had ik beide klassen samengevoegd. Bij het controleren of het klassendiagram een sluitende oplossing voor de systeemeisen bracht, kwam ik er echter achter dat het hiermee lastig zou worden actieproducten te herkennen. Door het splitsen van de klassen en het toevoegen van een attribuut prijsverschil kan nu worden teruggevonden of een artikel in prijs is gestegen of gedaald.

3.4.2 Use case diagram

Naast het klassendiagram heb ik ervoor gekozen om nog een UML model te gebruiken, namelijk de use cases. Use cases worden opgesteld om de functionaliteiten van het te bouwen systeem weer te geven. De use cases beschrijven hoe de gebruiker met het systeem om wil gaan. Met de use cases wordt een duidelijk beeld geschetst welke functionaliteiten een gebruiker kan gebruiken en wat deze functionaliteiten doen. Het systeem wordt hierbij gezien als een zwarte doos.

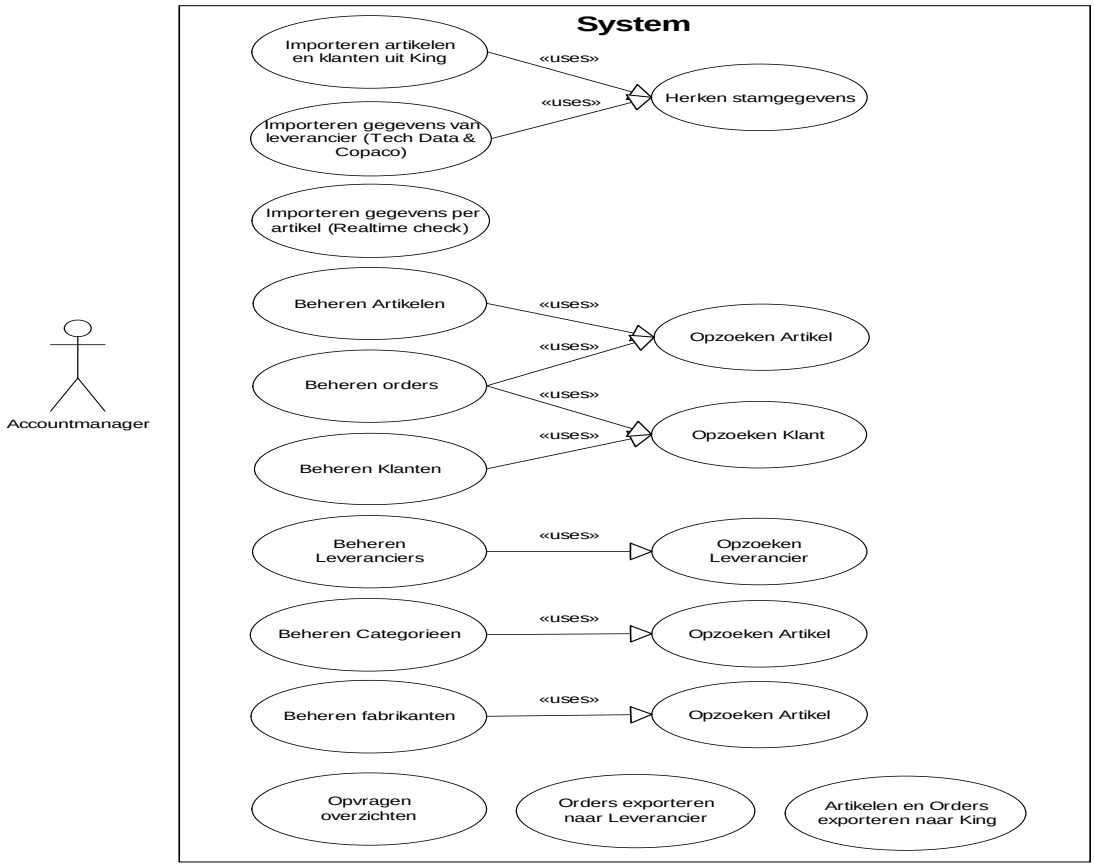
Voor elke basis systeemeis wordt een aparte use case beschreven. Dit heeft mij geholpen een beeld te vormen hoe de functionaliteit moet reageren op input van de

gebruiker. Hieronder wordt een use case beschrijving voor de import uit King weergegeven.

Naam	Importeren artikelen en klanten uit King
Samenvatting	Artikelgegevens en klantgegevens worden uit King geïmporteerd.
Actoren	Accountmanagement
Aannamen	King is op de pc geïnstalleerd
Beschrijving	(1) Een accountmanager exporteert de artikelen of klanten uit King in XML formaat. (2) Bij het opstarten van de tool controleert de tool of er een nieuw bestand uit King is geëxporteerd. Als er een nieuw bestand is gevonden wordt de gebruiker gevraagd deze te importeren. Mochten er nieuwe gegevens in het importbestand aanwezig zijn worden deze in de tool geïmporteerd. De import kan ook na het opstarten worden handmatig worden gestart. Dit kan handig zijn om gegevens tussentijds te synchroniseren. De gebruiker klikt hierbij op de importknop in het hoofdmenu.
Uitzonderingen	Er zijn foutieve artikelen in het importbestand opgenomen. Een foutief artikel is een artikel, waarin verplichte velden niet zijn gevuld. Hierdoor kan niet worden herkend of een artikel reeds in de database is opgenomen. Het artikel zal worden opgenomen in een logfile. De gebruiker krijgt hier aan het einde van de import een melding van.
Resultaat	Gegevens uit King zijn in de tool geïmporteerd en kunnen worden gebruikt in de orderverwerking.

Figuur: Use case beschrijving King import

Voor elke basis systeemeis (behalve voor de client server architectuur) heb ik een dergelijke use case beschrijving opgesteld. In het use case diagram zijn vervolgens de use cases die de gebruiker kan benaderen uitgezet. Het use case diagram geeft aan hoe het gedrag van de gebruiker is ten opzichte van het systeem. Om tekentechnische redenen heb ik geen lijnen van de actor naar de use cases toe geplaatst.



Figuur: Use case diagram

Opvallend aan het model is dat beheren orders van dezelfde subcase gebruik maakt als die van beheren artikelen en beheren klanten. Deze subcase is opgenomen om de accentueren dat bij het beheren dezelfde zoekfunctie wordt gebruikt.

3.4.3 GUI ontwerp

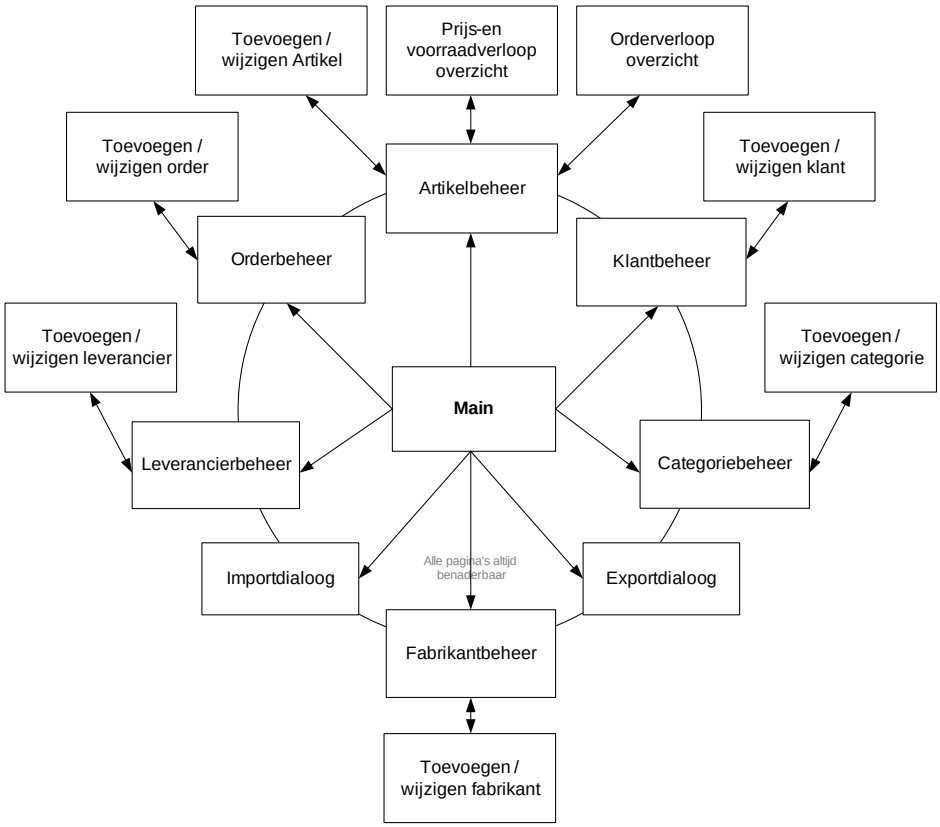
Als laatste onderdeel uit het systeemconcept is de user interface uitgewerkt. Het doel van het systeemconcept is het vastleggen van afspraken over de interface van het systeem. Een wens van de opdrachtgever is dat het systeem de King look-and-feel heeft. In dit onderdeel zijn daarom een aantal kenmerken uit de King user interface overgenomen. Zo worden bijvoorbeeld de stamgegevens (artikelen, categorieën, fabrikanten, leveranciers, orders en klanten), net zoals in King, op een apart tabblad getoond.



Figuur: King look-and-feel tabbladen

Tevens is de structuur van de applicatie uitgewerkt. Het doel hiervan is de schermindeling van het pakket vast te stellen. UML geeft geen model om de schermstructuur vast te stellen. Ik heb hierdoor de structuur uitgewerkt met een zelf bedacht model. Ik heb geen gebruik gemaakt van de boomstructuur, omdat deze naar mijn mening bij veel schermen erg onoverzichtelijk is. Het gebruikte model is

wel afgeleid van de boomstructuur. In dit model worden de lagen echter niet van boven naar beneden aangegeven, maar van binnen naar buiten. In het midden wordt het hoofdscherm aangegeven. Hoe verder de schermen naar buiten worden getoond, hoe dieper in het systeem gegaan wordt. Hieronder zal het opgestelde model worden getoond.



Figuur: Schermstructuur

Dit model is handig bij kleine systemen. In één oogopslag kan gezien worden welke schermen het systeem bestaat. Ik kwam echter tot de conclusie dat bij het toevoegen van schermen het model net als de boomstructuur steeds onoverzichtelijk wordt. Het model is echter nog zodanig overzichtelijk dat ik het model wel gebruikt heb.

3.5 Opstellen pilotplan

In het pilotplan worden de functionaliteiten ingedeeld in pilots. Een pilot is een bouwsteen. Alle bouwstenen bij elkaar vormen het systeem. Het doel van het pilotplan is het indelen van eisen in pilots, zodat het verder te volgen ontwikkeltraject kan worden ingepland.

De pilotindeling, zoals deze hieronder vermeld is, is tot stand gekomen aan de hand van de systeemeisen en het systeemconcept. Hierbij kwam vooral het use case diagram uit het systeemconcept van pas. In het use case diagram zijn op de database na alle functionaliteiten getoond, waardoor ik functionaliteiten die enige overlap of dezelfde strekking hebben kon clusteren tot een pilot. In onderstaande tabel worden de pilots beschreven met de daarbij bijbehorende systeemeisen.

Pilot	Systeemeisen	Prioriteit
1. Database	Database	basis
	Operationele systeemeisen (client/server, multi user)	basis
2. Import	Importeren artikelen en klanten uit King	basis
	Importeren gegevens van Tech Data	basis
	Performance eisen (import mag niet langer dan 5 minuten duren)	basis
	Realtime check	comfort
	Importeren gegevens van Copaco	luxe
3. Beheer en overzicht	Stamgegevens worden overzichtelijk weergegeven (totaal + detailoverzichten)	basis
	Zoekfunctie voor artikelen	basis
	Artikelen kunnen worden beheerd	basis
	Orders kunnen worden beheerd	basis
	Lay-out eisen (King look and feel)	basis
	Categorieën kunnen worden beheerd	comfort
	Leveranciers kunnen worden beheerd	comfort
	Klanten kunnen worden beheerd	luxe
	Fabrikanten kunnen worden beheerd	luxe
4. Export	Artikelen, orders en klanten exporteren in XML formaat	basis/ comfort
	Orders rechtstreeks exporteren naar/plaatsen bij leverancier	luxe

Naast het clusteren van de systeemeisen tot pilots worden in het pilotplan de pilotdelen bepaald. De pilotdelen zijn stukjes van een pilot. Door de pilots op te delen kan er een meer gedetailleerde planning worden gemaakt. Ook de planning per pilot wordt in het pilotplan opgenomen. Met deze planning wordt het te volgen traject bepaald. Bij het plannen zijn de volgende technieken gebruikt:

- Time-boxing
- Juicy bits first
- 80/20 regel

Binnen de pilotontwikkeling is er gebruik gemaakt van time-boxing. Ik heb van deze techniek gebruik gemaakt om het project te kunnen beheersen en er zorg voor te dragen dat de basis functionaliteiten met zekerheid gerealiseerd worden. Binnen het project zullen de pilots worden opgedeeld in time-boxes van variërende tijdlengtes. De tijd wordt als volgt ingedeeld. Voor de basiseisen zal 80% van de totaaltijd worden ingepland. De tijd voor comforteisen zal ongeveer 80% tot 120% van de totaaltijd zijn. De luxe eisen worden daarboven ingepland en worden alleen uitgevoerd als hier tijd voor is. Hieronder wordt ter illustratie de planning van de vierde pilot ‘export’ beschreven.

Pilotontwikkelplan	(Basis)	16 uur
Exportfunctionaliteit – Orders	(Basis)	40 uur
Exportfunctionaliteit – Artikelen	(Basis)	60 uur
Exportfunctionaliteit – Klanten	(Comfort)	40 uur
Orders rechtstreeks plaatsen bij leverancier	(Luxe)	80 uur
Totaal ingepland pilot 4: Export		236 uur
Timebox pilot 4: Export		140 uur

Figuur: Timebox Export

Tevens wordt er gebruik gemaakt van het ‘juicy bits first’ principe. Door de bouweenheden te sorteren en te categoriseren op prioriteit worden de voor de gebruiker meest interessante onderdelen als eerste ontwikkeld. Hierdoor kan de klant binnen de geplande tijd tevreden gesteld worden, zelfs wanneer het einde van de timebox is bereikt is en nog niet alles is voltooid.

De 80 / 20 regel stelt dat slecht 20% van de tijd benodigd is om 80% van de functionaliteiten van het systeem te realiseren. De overige 80% is benodigd om de overige 20% (meestal details) te realiseren. Door de pilotdelen te benoemen met een prioritering wordt in de planning rekening gehouden welke bouwstenen belangrijk zijn en welke minder.

Nadat de definitiefase is afgesloten is de planning niet meer aangepast in de definitiestudie. In het pilotontwerp van elke pilot is ook een planning opgenomen. Wijzigingen in de planning zijn alleen in deze ontwerprapporten doorgevoerd. Ik heb hiervoor gekozen, omdat er anders veel tijd verloren gaat aan het aanpassen van rapporten. De overall planning van de pilot ontwikkelfase is al eerder in de oriëntatiefase beschreven.

3.6 Bouwen prototype

Aan het eind van de definitiefase is een prototype opgesteld. Het prototype heeft tot doel de opdrachtgever een goed beeld te geven hoe het uiteindelijke systeem eruit komt te zien. Er zijn drie soorten prototypen, waaruit gekozen kan worden. Dit zijn de volgende prototypen:

- experimenteel prototype
- evolutionair prototype
- operationeel prototype

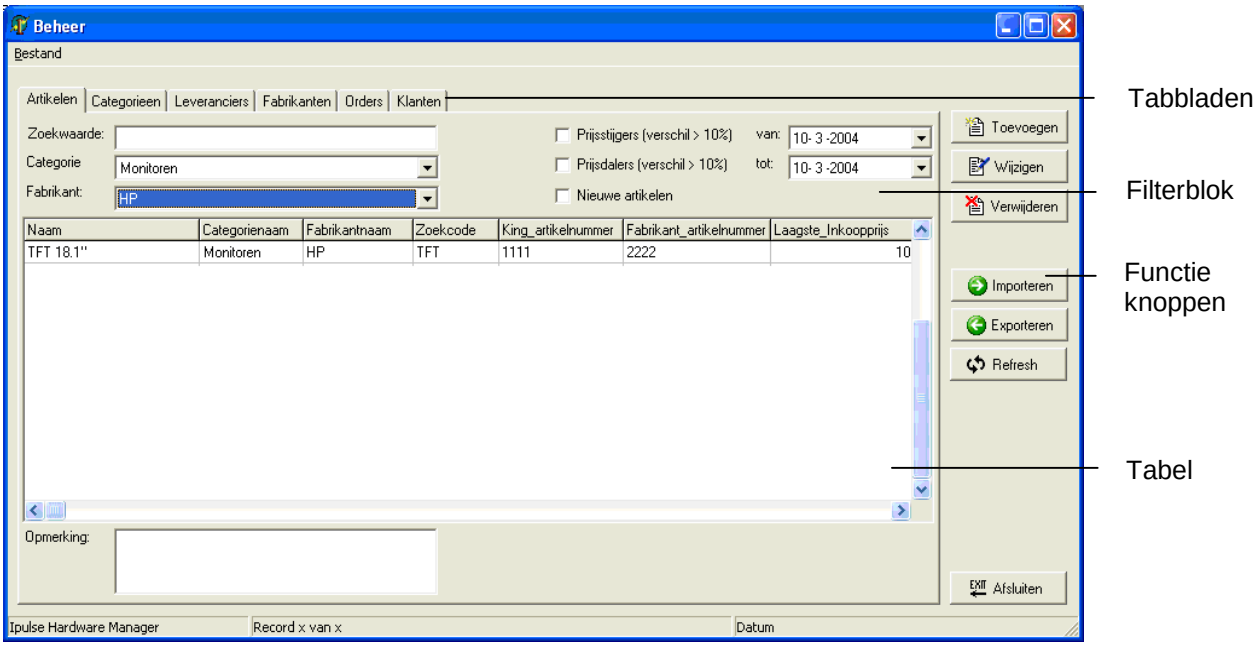
Een experimenteel prototype wordt voornamelijk gebruikt om te achterhalen of een aspect van het systeem haalbaar is. Een dergelijk prototype wordt snel opgesteld, zonder te letten op kwaliteitsnormen. Een experimenteel prototype wordt altijd weggegooid en maakt geen deel uit van het systeem.

Een evolutionair prototype wordt gebruikt om feedback van de gebruiker uit te lokken. Een dergelijk prototype is een vroege incomplete versie van het systeem. Meteen vanaf het begin wordt aandacht besteed aan kwaliteit. Het evolutionair prototype wordt steeds verder uitgewerkt tot het uiteindelijke systeem is afgerond.

Een operationeel prototype wordt gebruikt om een nieuwe systeemeis te verduidelijken. Vaak worden deze prototypen gebruikt om een uitbreiding van een pilot te specificeren en te verduidelijken.

Voor het huidige project is ervoor gekozen een evolutionair prototype te gebruiken. Het experimentele prototype viel af, omdat het mij onnodig leek tweemaal eenzelfde “schil” te bouwen. Het operationele prototype viel af, omdat het hier om een volledig systeem gaat en niet om een nieuwe systeemeis uit het systeem. Het evolutionair prototype heeft tot voordeel dat het meteen een goed overzicht geeft hoe de applicatie is opgebouwd. In het prototype zijn alleen de schermen opgebouwd, waarmee de opdrachtgever een eerste indruk van het systeem heeft gekregen. Achter de schermen is in dit stadium nog geen functionaliteit geprogrammeerd. Wel heb ik een aantal dummy waarden ingevoerd. Dit heb ik gedaan om de

opdrachtgever een beter beeld te geven hoe de applicatie eruit komt te zien. In onderstaand screenshot is het hoofdscherm van het prototype te zien.



Figuur: Prototype hoofdscherm

Het eerste prototype is na afronding met de opdrachtgever doorgenomen. Het voordeel hiervan is, is dat in een zeer vroeg stadium aanpassingen gedaan kunnen worden. Het verwijderen van bijvoorbeeld een knop is in dit stadium geen probleem. Het zou een groter probleem zijn als er bijvoorbeeld al tijd was besteed aan het plaatsen van code achter de knop. Deze tijd zou dan als verloren beschouwd kunnen worden. Een ander voordeel is dat de gebruiker vanaf het begin van de bouw een idee heeft hoe het systeem eruit komt te zien en eigenlijk al een beetje vertrouwd raakt met het systeem. Hiermee wordt bereikt dat aan het eind van het project de eindgebruiker geen totaal voor de gebruiker nieuw systeem moet gaan gebruiken. Dit systeem zou bijvoorbeeld totaal niet bij de wensen van deze gebruiker kunnen aansluiten. Dit probleem wordt voorkomen met het tonen van het prototype in de beginfase. De opdrachtgever heeft een aantal opmerkingen gegeven op het prototype. Zo wilde de opdrachtgever graag een filtermogelijkheid op nieuwe artikelen. Met de filtermogelijkheden kunnen snel artikeloverzichten worden gegenereerd. De filter is meteen (waar de opdrachtgever nog bij was) toegevoegd.

Met het prototype is de fase definitiestudie afgerond. De definitiefase wordt, zoals de gekozen strategie (RAD) voorschrijft, slechts eenmaal doorlopen. Ik heb ervoor gekozen eventuele wijzigingen in functionaliteit in de pilot ontwikkelingfase niet meer in de definitiestudie aan te passen. Dit om tijd te besparen. Dit zal ook voor een mogelijk toekomstige ontwikkelaar aan dit project geen problemen opleveren, omdat de wijzigingen wel in de pilotontwerp rapportages zijn opgenomen.

4. Pilotontwikkeling

De tweede fase uit het IAD ontwikkeltraject is de ontwikkelfase. In de ontwikkelfase wordt de IAD fase pilotontwikkeling uitgevoerd. De pilotontwikkelfase heeft als doel pilots, zoals beschreven in de definitiefase, te specificeren en te realiseren. Dit hoofdstuk heeft als doel inzicht te geven in de uitgevoerde werkzaamheden uit deze fase.



Figuur: IAD traject

Als eerste wordt de aanpak van de pilotontwikkeling beschreven. In deze paragraaf wordt op globaal niveau beschreven hoe ik deze fase heb aangepakt. Vervolgens worden de werkzaamheden aan de afzonderlijke pilots beschreven.

4.1 Aanpak pilotontwikkeling

In deze paragraaf wordt de aanpak van de ontwikkelfase op globaal niveau beschreven. In de fase pilotontwikkeling komt volgens IAD de pilot tot stand in een iteratie van drie hoofdbestanddelen. Dit zijn de volgende:

- Pilotontwerp (globaal en detail)
- Bouw
- Beoordeling en test

Bovenstaande fasen omvatten samen met de definitiestudie een compleet ontwikkeltraject. Om het ontwikkeltraject compleet te houden heb ik besloten bovenstaande drie fasen ook te doorlopen.

4.1.1 Pilotontwerp

Het pilotontwerp wordt opgesteld om het pilotplan uit de definitiestudie aan te scherpen. In het ontwerp zijn o.a. de detailspecificaties en de planning van het ontwikkeltraject voor de pilot terug te vinden. Het ontwerp wordt dus als leidraad gebruikt gedurende het ontwikkeltraject voor de pilot. Het pilotontwerp heb ik opgedeeld in de volgende terugkerende bestanddelen:

- Globaal functionele structuur
- Globaal technische structuur
- Pilotontwikkelplan
- Detailspecificaties

De ontwikkelaar wordt door IAD vrijgelaten in het invullen van deze bestanddelen. Ik heb er daarom voor gekozen aan het begin van elke pilot te onderzoeken welke onderdelen van belang zijn om tot realisatie van de pilot te komen. Dit betekent dat de inhoud (bijv. de gebruikte modellen) per pilot verschilt.

In het pilotontwerp zijn in eerste instantie de comfort - en luxe eisen (vastgesteld in de definitiestudie) niet uitgewerkt. Ik heb hiervoor gekozen om geen tijd te verliezen aan het beschrijven van eisen die wegens tijdgebrek wellicht niet gerealiseerd kunnen worden.

In elk pilotontwerp zijn de globale functionele structuur, globale technische structuur en het pilotontwikkelplan beschreven. Aangezien de onderdelen vaak globaal zijn worden deze niet verder in dit verslag beschreven. De globaal functionele structuur beschrijft globaal welke functionaliteiten de te bouwen pilot moet bevatten. Het doel van dit onderdeel is de gebruiker en ontwikkelaar een goed en ondubbelzinnig beeld te geven van de functionele inhoud van de pilot. Het doel van de globaal technische structuur is specificeren of valideren van de technische aspecten rond de ontwikkeling van de pilot. Zo wordt in dit hoofdstuk bijvoorbeeld de architectuur van de pilot en client/server beschrijving behandeld. Een terugkerend onderdeel is de beschrijving van de Delphi componenten, die worden gebruikt bij de pilot.

In het derde hoofdstuk uit het pilotontwerp wordt het pilotontwikkelplan beschreven. Dit plan bevat de planning van de pilot. Om te kunnen plannen wordt de pilot opgebroken in pilotdelen. Deze pilotdelen zijn te herkennen na het opstellen van de globaal functionele structuur. De planning uit de definitiestudie is als leidraad voor het pilotontwikkelplan gebruikt. Aanpassingen in de planning worden later in dit hoofdstuk beschreven.

Om een pilot te kunnen realiseren zijn, naast de globale informatie uit de eerste drie hoofdstukken, ook detailspecificaties opgesteld. De inhoud van de detailspecificaties wordt per pilot bepaald. Zo is het bijvoorbeeld noodzakelijk om voor de bouw van de database een databasemodel op te stellen.

Samengevat wordt in deze fase per pilot een ontwerprapport opgeleverd. In dit rapport is globale- en detailinformatie opgenomen om de pilot te kunnen realiseren.

4.1.2 Bouw

Het tweede onderdeel uit de pilotontwikkelfase is de bouw. Met de bouw wordt de functionaliteit uit de pilot gerealiseerd. De opdrachtgever is in deze fase veel op de werkvloer te vinden, om de voortgang te kunnen bespreken. Het voordeel van deze manier van werken is dat de opdrachtgever vertrouwd met het systeem raakt en het systeem ook evalueert naar iets dat de opdrachtgever echt wil hebben.

Zoals eerder beschreven is er een prototype gebouwd. Het prototype is tijdens deze fase uitgebreid. Aan de hand van het prototype en het pilotontwerp werden de functionaliteiten uit het systeem ingevuld.

4.1.3 Beoordeling en test

Het laatste onderdeel per pilotdeel is het testen van de pilot. Het doel van het testen is het traceren van eventuele bugs, zodat deze kunnen worden opgelost. Daarnaast wordt gecontroleerd of de tool aan de eerder opgestelde eisen voldoet. Ik heb besloten de volgende tests te gebruiken:

- Tussentijdse test. Deze test wordt uitgevoerd gedurende het bouwen van de applicatie. Steeds als een deeltje was afgerond wordt het onderdeel meteen

getest. Deze tests zijn door mijzelf uitgevoerd. Het doel van deze test is het controleren of de functionaliteit werkt en het snel kunnen vinden en oplossen van bugs.

- White box test. Met deze test worden alle mogelijke situaties die in het systeem kunnen voor komen nagelopen. Het voordeel van deze test is dat bugs op veel plaatsen uitgesloten kunnen worden. Een nadeel van deze test is dat het nalopen van alle mogelijke situaties tijdrovend is. Om deze reden heb ik deze test niet voor elke pilot doorlopen.

- Black box test. Bij een black box test wordt niet gekeken hoe het systeem de functionaliteit uitvoert, maar wat het resultaat is. Ik heb voor deze tests een persoon gevraagd die niet direct betrokken bij het project was. Een programmeur heeft vaak last van tunnelvisie, waarbij dingen over het hoofd worden gezien. Een persoon die niet betrokken is bij het project heeft een frisse kijk op het systeem en kan nuttige tips geven. Aan het einde van het project wordt een totale systeemtest uitgevoerd. Het kan namelijk voor komen dat elk pilotdeel afzonderlijk wel functioneert, maar functionaliteiten niet samen kunnen werken.

- Acceptatietest. Deze test wordt uiteindelijk uitgevoerd door de opdrachtgever. Met deze test bepaald de opdrachtgever of het systeem voldoet aan de eisen. De test is uitgevoerd aan het einde van het project. Over de inhoud van deze test heb ik zelf geen invloed gehad.

De testresultaten worden later per pilot in dit verslag beschreven. De acceptatietest is aan het eind uitgevoerd en wordt in het volgende hoofdstuk beschreven.

Per pilot heb ik een bijeenkomst georganiseerd, waarin ik de opdrachtgever het eindresultaat van de pilot toonde. Per pilot gaf de opdrachtgever zijn beoordeling en gaf hij aan of de pilot aan zijn wensen voldoet.

4.2 Pilot 1: Database

In deze paragraaf worden de activiteiten en keuzes beschreven die zijn uitgevoerd om de database te realiseren. De database is het hart van de te bouwen applicatie en wordt gebouwd om data op te slaan en op te vragen.

In deze paragraaf worden de belangrijkste onderdelen uit het pilotontwerp beschreven. Het pilotontwerp is opgesteld om de specificaties van de database te detailleren, zodat er genoeg informatie voorhanden is om de database te realiseren. In de subparagraaf bouw worden activiteiten bij de implementatie van deze pilot beschreven.

4.2.1 Pilotontwikkeling

Een belangrijk onderdeel bij databasebouw is het weergeven van de structuur. In dit onderdeel kan worden teruggevonden en worden nagedacht hoe de database wordt opgezet. Deze opzet is benodigd bij de implementatie van de database. Om de structuur weer te geven heb ik gekozen voor twee technieken die elk hun eigen doel hebben. Samen geven deze technieken een goed totaalbeeld van de structuur van de database. De onderdelen die ik heb uitgewerkt zijn de volgende:

- Databasemodel
- Relationeel representatiemodel

De keuze voor deze modellen wordt later in deze paragraaf gemotiveerd. In eerste instantie had ik ook een implementatiemodel ingepland. Aangezien het implementatiemodel voor het maken van de database overeenkomt met de code uit SQL Server 2000 voor de realisatie van de database, heb ik geen apart implementatiemodel opgesteld.

Databasemodel

Het doel van het databasemodel is het vastleggen van de structuur van de database. In dit model worden de tabellen en hun onderlinge relaties getoond. Het databasemodel wordt afgeleid van het klassendiagram. Voor het modelleren zijn verschillende modelleringtechnieken beschikbaar, te weten:

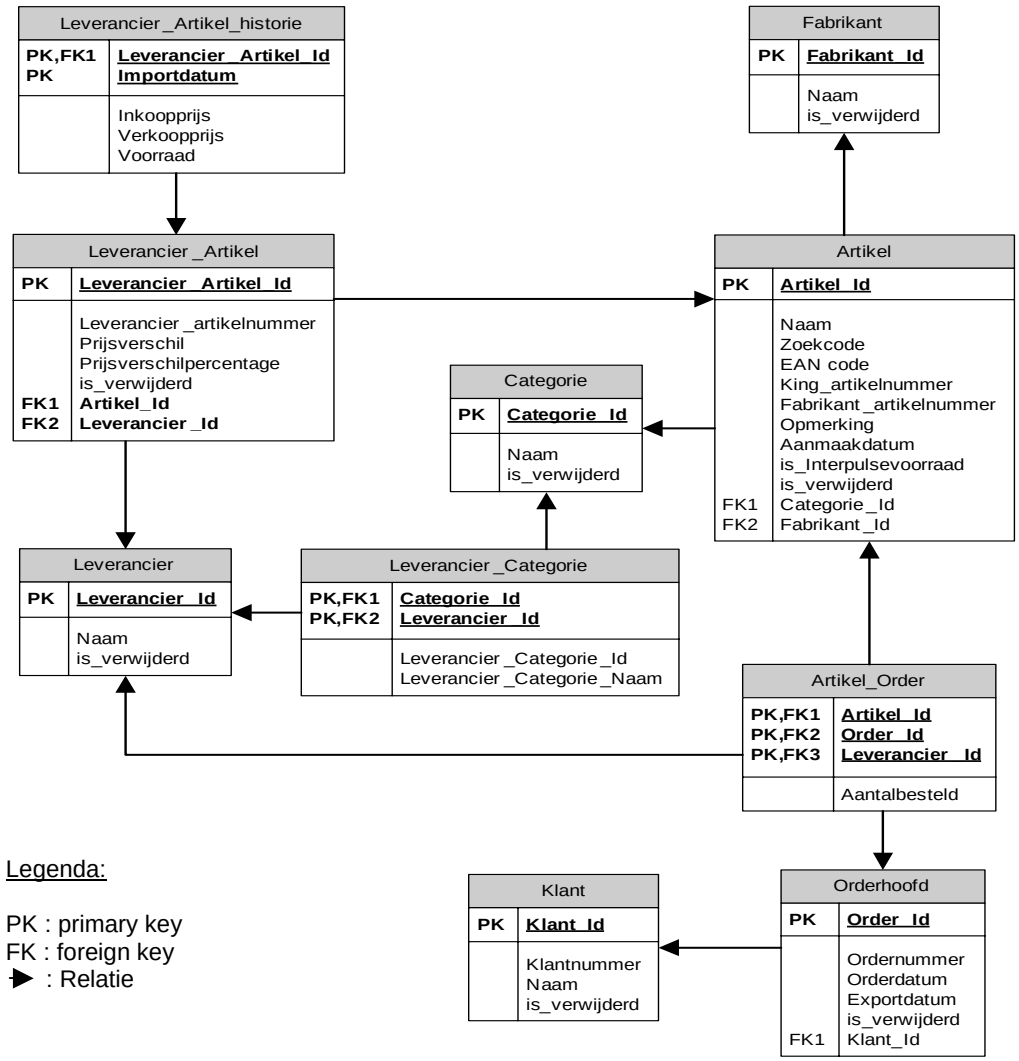
- ERD model
- Databasemodel uit Visio

Het ERD model is een klassiek model en geeft een overzichtelijk beeld van de structuur van de database. Een goed punt uit dit model is de weergave van de relaties. Naast de multipliciteit kan ook worden aangegeven of een relatie verplicht of optioneel is. Een nadeel van dit model is de aanpasbaarheid. Het aanpassen van een ERD model kan tekentechnisch veel werk kosten.

Het databasemodel lijkt meer op een klassendiagram uit UML. Ook dit model geeft een duidelijk beeld van de databasestructuur. Een nadeel aan dit databasemodel is, is dat de multipliciteit wel kan worden aangegeven in Visio, maar dat deze niet visueel in het model zichtbaar is. De aanpasbaarheid is daarentegen goed te noemen. Tevens heb ik deze modelleringwijze al in het oefenproject Grafi Magazijn gebruikt. Dit is me toen goed bevallen.

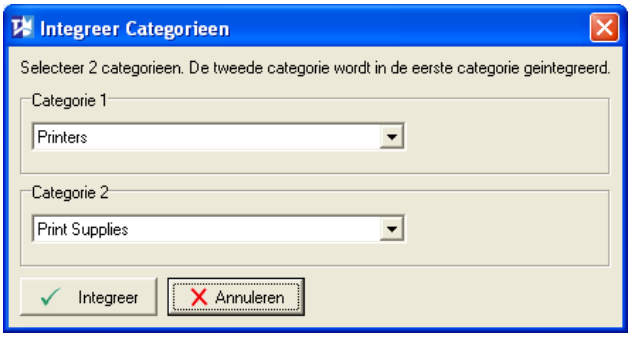
Aangezien de aanpasbaarheid goed is en ik al eerder goed met het databasemodel uit Visio heb gewerkt heb ik ervoor gekozen dit model te gebruiken. Daarnaast is het overzetten van een klassendiagram naar een databasemodel makkelijker dan het overzetten naar een ERD. Dit omdat ze tekentechnisch veel op elkaar lijken.

Het databasemodel is in eerste instantie opgebouwd door het klassendiagram één op één over te zetten. Het klassendiagram bevat nog geen primaire sleutels, dus die zijn per tabel toegevoegd. Ik heb besloten per tabel een eigen nummering bij te houden. Hiermee kan ik makkelijker vermijden dat er dubbele records in de database voor komen. Veel tabellen hebben hierdoor een nieuw id toegekend gekregen. De tussentabellen hebben een combinatie van de vreemde sleutels als primaire sleutel meegekregen.



Figuur: Databasemodel

In het databasemodel is ten opzichte van het klassendiagram nog een aantal belangrijke aanpassingen gedaan. Opvallend is de extra tabel tussen categorie en leverancier. Deze tabel heb ik opgenomen, omdat per leverancier andere categorieën kunnen worden aangehouden. Bijvoorbeeld bij Tech Data kunnen printerartikelen onder de categorie ‘printers’ vallen, maar bij Copaco kan dit ‘print supplies’ zijn. In de nieuwe situatie worden de categorieën per leverancier geïmporteerd, waarbij veel categorieën dubbel in de database voor komen. Door een integratiemodule op te nemen kunnen twee categorieën met elkaar worden samengevoegd. Deze komen dan in de leverancier_categorie tabel terecht. Twee categorieën worden dan onder dezelfde naam aan de gebruiker getoond.



Figuur: Integreer categorieën

Gedurende de pilotontwikkelfase is het databasemodel nog een aantal keer aangepast. Belangrijke wijzigingen worden in paragraaf ‘gemaakte iteratieslagen’ beschreven.

Relationeel representatiemodel

Het representatiemodel geeft een weergave van de verschillende objecten (tabellen) en attributen die in de uiteindelijke database zullen staan. Ik heb dit model opgesteld om de eerder gestructureerde database te kunnen normaliseren. De database is tot de vierde normaalvorm genormaliseerd. Dit betekent dat de relaties functioneel afhankelijk van elkaar zijn, geen overtolligheden en tegenspraak en niet meerwaardig afhankelijk zijn. Deze verwerking voorkomt problemen in de toekomst, door mogelijk inconsistente gegevens die de integriteit van de database aantasten.

Onderzoek naar Delphi databasecomponent

In het eerder beschreven interview uit de pilot ontwikkelworkshop bracht de opdrachtgever naar voren dat hij graag een client/server structuur voor de tool wil hebben. Dit betekent dat de database wordt geïmplementeerd op een server. Meerdere clients kunnen via het interne netwerk van Interpulse de database bereiken. De afzonderlijke clients kunnen de gegevens uit de database ophalen, invoeren en bewerken.

Om een client/server structuur te bewerkstelligen ben ik op zoek gegaan naar een component uit Delphi die de koppeling tussen client en server kan invullen. Ik heb een aantal databasecomponenten onderzocht. Ik heb ervoor gekozen juist deze componenten te onderzoeken, omdat deze componenten standaard in Delphi zijn opgenomen. Dit voorkomt extra kosten en installatieproblemen. De onderzochte componenten zijn:

- ADO
- BDE
- DbExpress

Bovenstaande databasecomponenten heb ik met elkaar vergeleken. Via een testapplicatie en een korte literatuurstudie heb ik onderzocht wat de eigenschappen van de afzonderlijke componenten zijn. Voor de literatuurstudie heb ik de beschikbare Delphi literatuur en diverse internetbronnen geraadpleegd. Daarnaast heb ik andere programmeurs binnen Interpulse gevraagd over hun ervaringen met de componenten. Met de volgende punten heb ik tijdens het onderzoek rekening gehouden.

- Geschikt voor client/server applicaties.
- Beschikbare kennisbronnen.
- Beveiliging

Ik heb deze drie punten gekozen om de volgende redenen. Het eerste punt is duidelijk. Als het component niet geschikt is kan er ook geen client/server architectuur mee gemaakt worden. Het tweede punt heb ik gekozen, omdat ik nog niet eerder een client/server architectuur binnen Delphi heb opgezet. Ik zal dus waarschijnlijk andere kennisbronnen nodig hebben om mijn kennis aan te vullen. Het laatste punt is de beveiliging. Ik heb dit punt gekozen, omdat binnen de tool bedrijfsgevoelige informatie is opgenomen. Het is dus niet gewenst dat een extern iemand, zomaar de database binnen kan. De resultaten uit het onderzoek zijn samengevat in onderstaande tabel.

	BDE	ADO	DbExpress
Geschikt voor client/server applicaties	--	+	++
Beschikbare kennisbronnen	+	-	++
Beveiliging	++	++	-

Figuur: Resultaat onderzoek databasecomponent

Uit bovenstaande tabel is af te lezen dat BDE totaal niet geschikt voor client/server applicaties is. Dit component is gebouwd om databases te koppelen aan stand alone applicaties. Het component bevat bijvoorbeeld geen eigenschappen voor ‘multi threading’, waardoor clients niet tegelijk met de applicatie kunnen spreken en snel deadlocks kunnen ontstaan. Hiermee valt dit component meteen af. ADO is geschikt voor stand-alone en client/server applicaties. DbExpress heeft veel features voor client/server applicaties en is daarmee uitermate geschikt.

Voor de beschikbare kennisbronnen heb ik Delphi literatuur en helpfile geraadpleegd. BDE en DbExpress zijn gefabriceerd door Borland en daarom ook goed binnen Delphi gedocumenteerd. ADO is gefabriceerd door Microsoft en is iets minder goed gedocumenteerd in Delphi. Naast de Delphi bronnen heb ik de programmeurs binnen Interpulse geraadpleegd. Hierbij heb ik de componenten aan de programmeurs voorgelegd en o.a. gevraagd welk component het meest gebruikt wordt. DbExpress wordt als standaard binnen Interpulse gebruikt, waardoor het kennisniveau hoger ligt dan voor de andere componenten.

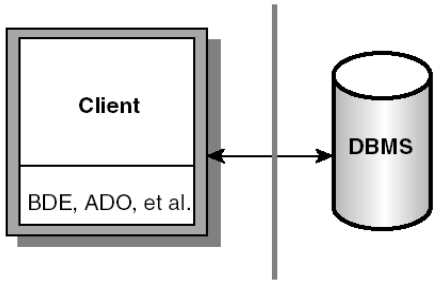
Voor informatie over de beveiliging heb ik wederom de programmeurs en mijn eigen testapplicatie geraadpleegd. DbExpress maakt gebruik van een socketconnection, waarin gegevens in pakketjes naar de database op de server worden verstuurd. Dit brengt extra overhead met zich mee, waardoor de performance iets minder is dan bij de andere componenten. Deze maken geen gebruik van socketconnections, maar verbinden met de database d.m.v. een login. Het voordeel van een socketconnection is dat de Microsoft beveiliging wordt omzeild, waardoor geen extra handelingen bij het verbinden met de database gemaakt hoeven worden. Dit pluspunt is tevens een minpunt. De socketconnection maakt verbinding via een poort op de server. Dit komt de veiligheid niet ten goede. Als de poort bij een buitenstaander bekend is kan deze relatief gemakkelijk bij de database.

Na de componenten onderzocht te hebben heb ik mijn keuze voor een component gemaakt. BDE viel meteen bij het eerste punt af, waardoor de keuze tussen ADO en DbExpress gemaakt moest worden. Mijn keuze is voor DbExpress gevallen. Dit omdat het kennisniveau binnen Interpulse hoog ligt en het component uitermate geschikt is voor client/server applicaties. Het minpunt voor beveiliging is reeds binnen Interpulse opgelost. Zo is er voor mijn tijd bij Interpulse al een encryptiemodule geschreven, waardoor de verstuurde data niet door buitenstaanders gelezen kan worden. Daarnaast is er een login op de poort van de server geplaatst.

Client/Server architectuur

Naast de eerder genoemde nadelen op het gebied van performance en beveiliging heeft DbExpress een aantal voordelen. Deze worden hieronder beschreven.

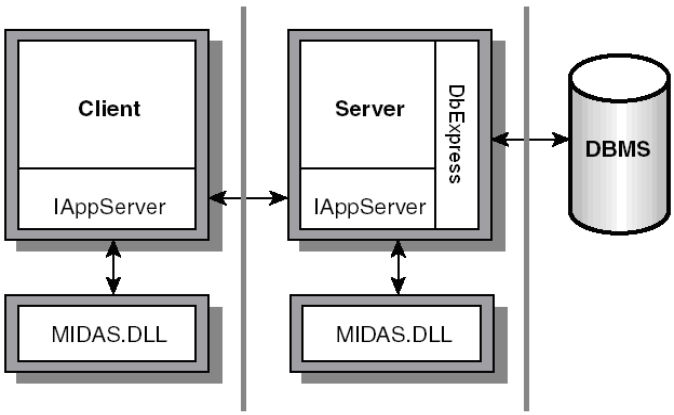
Een belangrijk verschil tussen DbExpress en klassieke client/server componenten is de architectuur. In klassieke client/server applicaties wordt een architectuur gebruikt zoals hieronder weergegeven.



Figuur: Klassieke client/server architectuur

In bovenstaand figuur is te zien dat in klassieke client/server applicaties de databasebibliotheken (dll’s) op elke client zijn opgenomen. Dit is ook meteen het zwakke punt van deze structuur. Als een wijziging in een bibliotheek wordt aangebracht dient deze voor alle afzonderlijke clients te worden uitgevoerd.

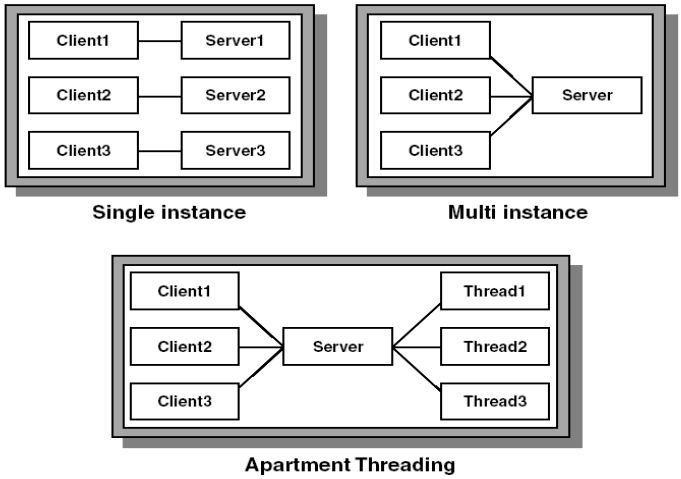
In de architectuur van een DbExpress applicatie worden de databasebibliotheken alleen in de server opgenomen. Hierdoor hoeft een wijziging slechts op één plaats te worden aangepast. Dit is een groot voordeel op bovenstaande architectuur. Een dergelijke applicatie is veel makkelijker te beheren.



Figuur: Multitier architectuur

Een belangrijk onderdeel bij het opzetten van de client/server structuur is de instantiekeuze. De instantie houdt in hoeveel clients tegelijk contact met de server kunnen hebben. Er zijn drie verschillende instantieopties binnen DbExpress voorhanden. Dit zijn de volgende:

- Single instance: Voor elke client is er een aparte server. Elke client gebruikt server bronnen voor elke server die gecreëerd wordt.
- Multi instance: Alle clients hebben contact met dezelfde server. Dit betekent ook dat van dezelfde databaseconnectie gebruik wordt gemaakt. De afzonderlijke clients kunnen echter niet tegelijk met de server praten en moeten op elkaar wachten.
- Apartment threading: Alle clients praten met dezelfde server. Met deze optie kunnen de clients tegelijk met de server praten. Hierbij moet de programmeur een synchronisatiemodule inbouwen.



Figuur: Instantiekeuze

Ik heb gekozen voor de laatste optie. Deze optie geeft het grote voordeel dat de afzonderlijke clients tegelijk met de server kunnen praten. De server maakt voor elke client een aparte thread aan, zodat de clients tegelijk op de database kunnen werken. Om dit in goede banen te leiden is het reconcile error mechanisme ingebouwd.

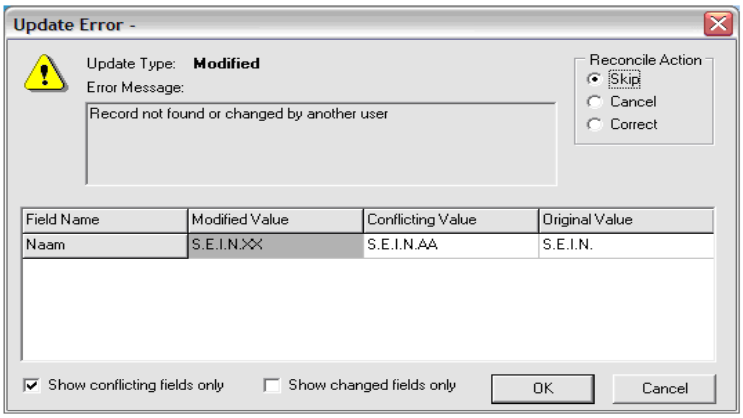
Reconcile error mechanisme

Het reconcile error mechanisme regelt het gedrag van de database als er meerdere clients op hetzelfde record werken. Het reconcile error mechanisme werkt al volgt. Een client kan meerdere wijzigingen op de database uitvoeren, bijvoorbeeld door de naam van een artikel te wijzigen. Een wijziging op de database wordt niet meteen opgeslagen in de database, maar wordt onthouden in het geheugen van de client. Als de functie ‘ApplyUpdates’ wordt aangeroepen zal de client haar wijzigingen aan de server doorgeven. Er wordt op dat moment gecontroleerd of een andere client ook al eerder wijzigingen op dat record heeft gedaan. Dit kan doordat DbExpress ook de oude waarde van het record onthoudt. Bij het wegschrijven van het aangepaste record wordt de oude waarde van het record dat bij de client vandaan komt vergeleken met die in de database. Als deze niet overeenkomen, betekent dit dat een andere client in de tussentijd wijzigingen heeft gedaan. In dit geval wordt een reconcile error gegenereerd.

Bij een reconcile error kan de applicatie verschillend reageren. De volgende reacties zijn mogelijk:

- De gehele update wordt afgebroken. Alle wijzigingen van de betreffende client gaan verloren.
- De update van het record, waarop de error wordt gegenereerd wordt overgeslagen. De andere updates kunnen wel doorgaan.
- De update overschrijft de waarden. De update van de voorgaande client wordt hiermee teniet gedaan.

Ik heb in overleg met de opdrachtgever ervoor gekozen de gebruiker zelf de keuze te geven welk gedrag de database moet aanhouden. Hierbij krijgt een gebruiker een melding, waar de gebruiker kan kiezen wat het systeem moet doen.



Figuur: Reconcile error

Dit is geen gebruikersvriendelijke manier, maar wel veilig. Om dit meer gebruikersvriendelijk te maken moet per tabel uitgezocht worden hoe het systeem zich moet gedragen. Dit is erg veel werk. Aangezien de veiligheid goed is, heeft dit punt een lage prioriteit gekregen.

4.2.2 Bouw

Zoals eerder beschreven is de database in Microsoft SQL Server geïmplementeerd. Bij de implementatie is gebruik gemaakt van een script. In dit script wordt de database en bijbehorende tabellen gecreëerd. Ik heb in dit script aangegeven van welke type de velden uit een tabel zijn en wat de referentiele integriteit bij de vreemde sleutels is. Ter verduidelijking is hieronder is het implementatiescript van de artikeltabel beschreven.

```
CREATE TABLE Artikel (
            Artikel_Id int IDENTITY (1, 1)
                        CONSTRAINT PK_Artikel_Id PRIMARY KEY,
            Naam varchar(400)                   NOT NULL,
            Zoekcode varchar(250),
            EANcode varchar(20),
            King_artikelnummer varchar(20),
            Fabrikant_artikelnummer varchar(20),
            Opmerking text,
            Aanmaakdatum datetime               NOT NULL,
            is_Interpulsevoorraad bit           NOT NULL,
            is_verwijderd bit                   NOT NULL,
            Categorie_Id int
                        CONSTRAINT FK_Categorie_Id
                              REFERENCES Categorie(Categorie_Id)
                                    ON DELETE NO ACTION
                                    ON UPDATE NO ACTION,
            Fabrikant_Id int
                        CONSTRAINT FK_Fabrikant_ID
                              REFERENCES Fabrikant(Fabrikant_Id)
                                    ON DELETE NO ACTION
                                    ON UPDATE NO ACTION )
```

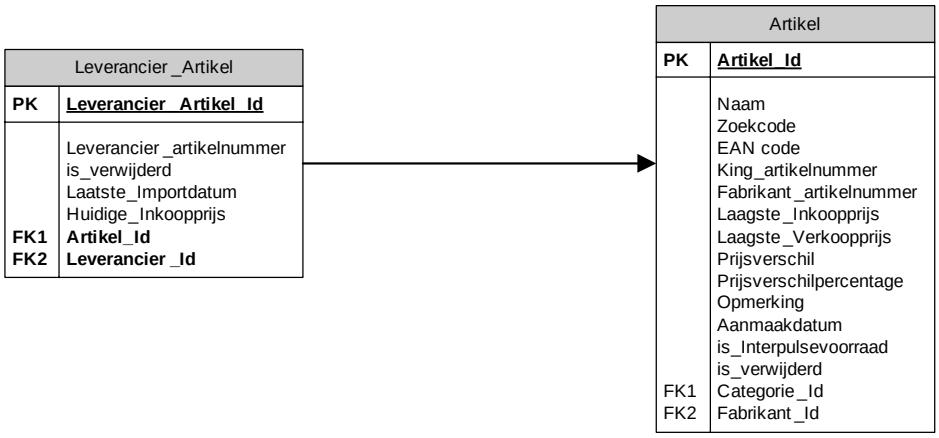
Figuur: Creëren artikeltabel in SQL Server 2000

De database is in eerste instantie op een testserver opgebouwd. Hiermee kon ik gedurende het project zonder problemen de database wijzigen en testen.
Na het bouwen van de database is het prototype gekoppeld aan de database. Bij de koppeling heb ik eerst de serverkant van het prototype uitgewerkt. Aan deze kant zijn de DbExpress SQL queries geplaatst die de data uit de database halen. Aan de client kant zijn componenten geplaatst die van deze queries data krijgen aangereikt. De client- componenten geven vervolgens weer de data door aan de overzichtcomponenten (grids, comboboxen, etc.). Na deze activiteit werd er kort getest of de koppelingen goed functioneerde en was de eerste pilot ‘afgerond’.

4.2.3 Beschrijving iteratieslagen

De hierboven genoemde activiteiten zijn in één iteratieslag uitgevoerd. De pilot is echter wegens tussentijdse aanpassingen pas echt afgerond aan het einde van het project. Een aantal belangrijke aanpassingen worden hieronder beschreven.

In de tweede pilot wordt importfunctionaliteit gerealiseerd. Hierbij is vooraf de performance-eis gesteld dat de import binnen 5 minuten moet zijn afgerond. Deze performance-eis werd niet gehaald. Eén van de manieren om de import sneller te maken is het veranderen van de database. Een belangrijke reden van het traag worden was de historie tabel. Deze tabel wordt bij elke import gevuld, maar eigenlijk nooit geleegd. Dit om geen historiegegevens kwijt te raken (wens van de opdrachtgever). Om deze tabel in bepaalde situaties niet te hoeven openen is een veld ‘huidige_inkoopprijs’ en ‘laatste_importdatum’ in de tabel leverancier_artikel opgenomen.



Figuur: Relatie tussen Leverancier_Artikel – Artikel na iteraties

De oplettende lezer merkt ook op dat de artikeltabel is uitgebreid met een aantal velden. Bij de derde pilot worden overzichten gegenereerd. In het artikeloverzicht is het prijsverschil en prijsverschilpercentage van belang. Om het overzicht te genereren moesten hierdoor gegevens uit twee tabellen worden gehaald. Om de snelheid te verhogen heb ik besloten het prijsverschil en het prijsverschilpercentage in de artikeltabel op te nemen, waardoor de gegevens uit slechts één tabel gehaald kunnen worden.

Een laatste iteratieslag op de database werd in de laatste pilot, export, gemaakt. In deze pilot wordt de export naar King gerealiseerd. Om artikelen in King te kunnen importeren is de categorie vereist. De categorie wordt door de import van King herkend aan het King categorienummer. Deze moet in King bestaan. Het King

categorienummer was echter niet aanwezig in de Ipulse Hardware Manager database. Ik heb hier een extra veld voor opgenomen. De eindgebruiker dient nu eenmaal bij elke categorie uit de Ipulse Hardware Manager database een King categorienummer in te voeren.

4.3 Pilot 2: Gegevensimport uit Tech Data en King

De importfunctionaliteit zorgt ervoor dat gegevens van het assortiment van Tech Data en artikel- en klantgegevens uit King in de database van de Ipulse Hardware Manager kunnen worden geïmporteerd. Met deze gegevens kunnen overzichten en orders worden opgesteld.

Voor de importpilot heb ik drie pilotdelen uitgevoerd. Het vierde pilotdeel, de import van Copaco bestanden, is wegens tijdgebrek niet uitgevoerd. Dit was een luxe eis en heeft dus verder geen gevolgen voor het slagen van dit project. Hieronder worden de uitgevoerde pilotdelen met de bijbehorende activiteiten beschreven:

- King – parsen en importeren (*Basis*)
    - Onderzoek XML
    - Opstellen detailspecificaties (artikelen en klanten afzonderlijk)
    - Opstellen testplan (voor white box test)
    - Bouw functies
    - Testen

- Tech Data – downloaden, parsen en importeren (*Basis*)
    - Onderzoek importbestanden Tech Data
    - Onderzoek downloaden bestanden via Tech Data
    - Opstellen detailspecificaties
    - Uitbreiden testplan
    - Bouw functies
    - Testen

- Realtime check Tech Data (*Comfort*)
    - Onderzoek Tech Data webservice
    - Opstellen detailspecificaties
    - Bouw functies
    - Testen

De pilotdelen zijn in een iteratief proces ontwikkeld. Elk pilotdeel is in een aparte iteratie ontwikkeld. Aangezien de manier waarop het detailspecificaties zijn opgesteld en de bouw is uitgevoerd per pilotdeel nauwelijks van elkaar verschillen worden de activiteiten hieronder gezamenlijk beschreven. Het testplan voor de King import is in de externe bijlagen opgenomen. Het Tech Data testplan heeft weinig toegevoegde waarde. Om de omvang van de externe bijlagen te beperken heb ik deze daarom niet opgenomen.

4.3.1 Pilotontwerp

Het pilotontwerp is middels een aantal activiteiten opgesteld. De activiteiten om het pilotontwerp op te stellen worden hieronder beschreven. Meer informatie over het pilotontwerp kan gevonden worden in de externe bijlage “pilotontwerp 2: import”.

Onderzoek naar XML

Bij de King import wordt gebruik gemaakt van XML importbestanden. XML was voor mij een nieuw begrip. Het doel van dit onderzoek is te begrijpen wat XML is en hoe ik een XML bestand kan uitlezen. Via een korte literatuurstudie (via het internet) heb ik onderzocht wat XML is. Om te ontdekken hoe een King XML bestand (voor artikelen en klanten) is opgebouwd heb ik de King helpfile geraadpleegd. Hieronder geef ik een korte samenvatting van de door mij opgedane informatie.

XML staat voor 'eXtended Markup Language' en is een taal, waarmee documenten gestructureerde informatie kunnen bevatten. XML lijkt sterk op HTML. In XML wordt net als in HTML gebruik gemaakt van ‘tags’. Een belangrijk verschil tussen XML en HTML is echter dat de tags in XML door de auteur zelf mogen worden bepaald. Daarnaast worden XML-tags niet gebruikt om een bepaalde vormgeving aan te duiden, maar zeggen ze iets over de inhoud.

Hieronder word aan de hand van een voorbeeldje uitgelegd hoe het artikel importbestand in King is opgebouwd. Het onderstaande importbestand is verkleind om de uitleg eenvoudiger te maken.

```
<KING_ARTIKELEN>
      <ARTIKEL>
            <ART_NUMMER>983204830</ART_NUMMER>
            <ART_ZOEKCODE>Mon17</ART_ZOEKCODE>
            <ART_OMSCHRIJVING>17” Monitor</ART_OMSCHRIJVING>
      </ARTIKEL>
      <ARTIKEL>
            <ART_NUMMER>919019181</ART_NUMMER>
            <ART_ZOEKCODE>Scan</ART_ZOEKCODE>
            <ART_OMSCHRIJVING>Scanner</ART_OMSCHRIJVING>
      </ARTIKEL>
</KING_ARTIKELEN>
```

Figuur: King artikel XML bestand

In bovenstaand voorbeeld wordt een King importbestand beschreven dat twee artikelen bevat. Een King artikelimportbestand begint altijd met de tag <KING_ARTIKELEN> en wordt afgesloten met de tag </KING_ARTIKELEN>. Een artikel kan meerdere keren voor komen. Een item wordt steeds met een begin en een eindtag gedefinieerd. Binnen een XML bestand wordt deze opbouw steeds aangehouden.

Om een XML bestand uit te kunnen lezen in Delphi is een component benodigd, waarmee een extern XML bestand geopend kan worden en worden benadert door de te bouwen tool. Om dit te bewerkstelligen ben ik op zoek gegaan naar een standaardcomponent uit Delphi. Dit om extra kosten en installatieproblemen te voorkomen. Om een geschikt component te vinden heb ik de Delphi helpfile en internetbronnen geraadpleegd. In Delphi 6 is standaard slechts één component aanwezig, waarmee XML bestanden kunnen worden geopend en worden uitgelezen. Ik heb dus ook voor dit component gekozen. De artikelomschrijving, zoals opgenomen in het bovenstaande voorbeeld XML document, wordt als volgt uitgelezen:

```
ArtikelLijst.ChildNodes['ART_OMSCHRIJVING'].Text
```

Figuur: Uitlezen XML bestand

Opstellen detailspecificaties

In tegenstelling tot de vorige pilot bestaat deze pilot uit meer programmeren. Voor deze pilot zijn dan ook andere detailspecificaties benodigd. De onderstaande detailspecificaties zijn voor alle pilotdelen apart uitgewerkt.

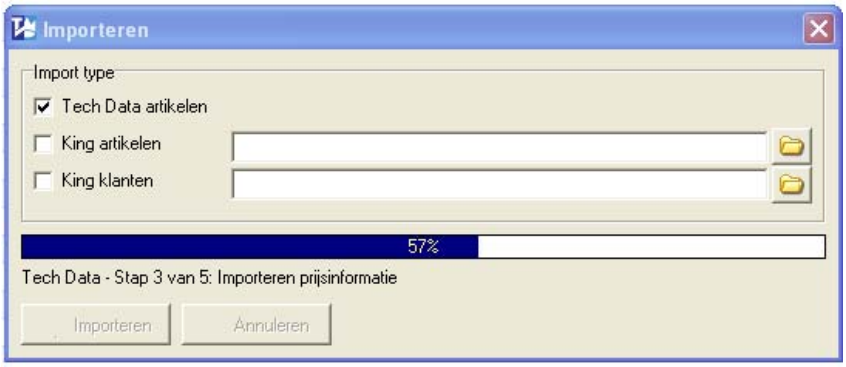
- Proces import tekstueel. In dit onderdeel wordt het importproces tekstueel beschreven. Dit importproces is in overleg met de opdrachtgever opgesteld.

- Structuur importbestand. In dit onderdeel wordt de structuur van het te importeren King – of Tech Data importbestand beschreven.

- Modelleren importproces. In dit onderdeel wordt een model opgesteld om het importproces te specificeren.

Beschrijven importproces tekstueel

De pilot import is een lastige pilot om met de opdrachtgever te bespreken. De importfunctionaliteit wordt vooral onder de motorkap van de applicatie uitgevoerd. Het enige wat de gebruiker zal zien is het volgende scherm:



Figuur: Importscherm

Om toch de opdrachtgever een beeld te geven wat de import doet heb ik besloten de importfunctionaliteit beknopt uit te schrijven. Ik heb dit niet via een model gedaan om te voorkomen dat de opdrachtgever deze niet lezen kan. Deze tekstuele beschrijving heeft ook mij geholpen een beter beeld te vormen wat de import gaat doen.

Het tekstuele importproces is met de opdrachtgever besproken. Doordat het importproces toch een technisch verhaal blijft heeft de opdrachtgever geen wijzigingen aangedragen en mij de vrijheid gegeven het importproces zelf in vullen.

Beschrijven structuur importbestanden

Het beschrijven van de structuur van het importbestand is een essentiële activiteit om het importproces te kunnen realiseren. Met deze activiteit kan namelijk worden bepaald hoe het importbestand is opgebouwd (bevat het bestand de gewenste informatie), maar er kan ook worden bepaald aan welke velden een uniek artikel herkend kan worden. De herkenning is belangrijk, omdat het niet gewenst is dat er dubbele artikelen in de tool worden geïmporteerd.

De structuur van de importbestanden uit King heb ik bepaald uit de King helpfile. De bestanden uit King bleken zeer veel velden te bevatten. Om hier overzicht in te

krijgen heb ik een tabel opgesteld, waarin de benodigde velden uit de tool tegen de velden in het importbestand werden uitgezet. Hieronder wordt een fragment uit deze tabel getoond.

Benodigd in tool	King veld	Veld in Delphi
Leverancier_artikel		
Leverancier_artikelnummer	Artikelnummer bij leverancier	ART_LEVERANCIER_ARTIKEL-NUMMER
Artikel_Id	-	Via artikeltabel
Leverancier_Id	Leveranciernummer	ART_LEVERANCIER_NUMMER

Figuur: Benodigde velden uit King

Met deze tabel kreeg ik overzicht in welke velden benodigd waren en kon ik achterhalen welk veld of veldcombinatie gebruikt kon worden om een uniek artikel terug te kunnen vinden. Hierbij heb ik drie mogelijke veldcombinaties uit de bovenstaande tabel onderzocht:

- *Oplossing 1*: Fabrikant_id in combinatie met fabrikant_artikelnummer.
  Deze combinatie van deze velden geeft een unieke waarde, waaraan elk artikel herkent kan worden. Het fabrikant_artikelnummer is in King boekhoudprogramma zelf opgenomen. In het exportbestand van King is deze echter niet opgenomen, waardoor ook deze combinatie niet bruikbaar was.

- *Oplossing 2:* EAN code
  De EAN code is een Europese codestandaard en wordt gebruikt voor snellere herkenning van producten. Hiermee wordt ook meteen de zwakte van deze code voor gebruik in de herkenningsmodule duidelijk. De meeste producten buiten Europa hebben geen EAN code. Deze code is daarom voor mij niet bruikbaar.

- *Oplossing 3:* Leverancier_artikelnummer in combinatie met leverancier_id
  Deze combinatie is zwakker dan de fabrikant combinatie. Dit omdat de leverancier kan besluiten haar artikelnummers te herzien. Hiermee zou de herkenning van artikelen helemaal mislopen. Ik heb dit probleem aan de opdrachtgever voorgelegd. De opdrachtgever schatte de kans dat een leverancier haar artikelnummers zou herzien zodanig klein dat de keuze toch voor deze combinatie is gevallen.

Ook het klanten importbestand heb ik bestudeerd. Bij het bestuderen van dit bestand ben ik geen problemen tegengekomen. Alle in de tool benodigde velden zijn in het bestand opgenomen.

Naast de King importbestanden zijn ook de Tech Data importbestanden onderzocht. Hiervoor heb ik via de opdrachtgever documentatie ontvangen. In deze documentatie staat precies beschreven hoe de Tech Data bestanden zijn opgebouwd. Daarnaast heb ik een test importbestand via Tech Data ontvangen. Bij de artikelimport van Tech Data wordt slechts van één XML document gebruik gemaakt. De informatie uit van Tech Data moet uit 5 verschillende tab-gescheiden importbestanden worden gehaald, te weten: products, categories, prices, stock en obsolete. Om hier overzicht in te krijgen is daarom ook voor de Tech Data import een tabel opgesteld om de gewenste velden te kunnen herkennen. Voor de Tech Data import is wel gebruik gemaakt van de eerder beschreven fabrikant combinatie.

Mochten er toch nog artikelen zijn die niet herkend worden, heb ik besloten een extra vangnet in het programma op te nemen. Ook voor artikelen zal een integratiemodule

worden opgenomen. Deze techniek van samenvoegen heb ik al eerder voor categorieën in dit verslag beschreven.

Modelleren importproces

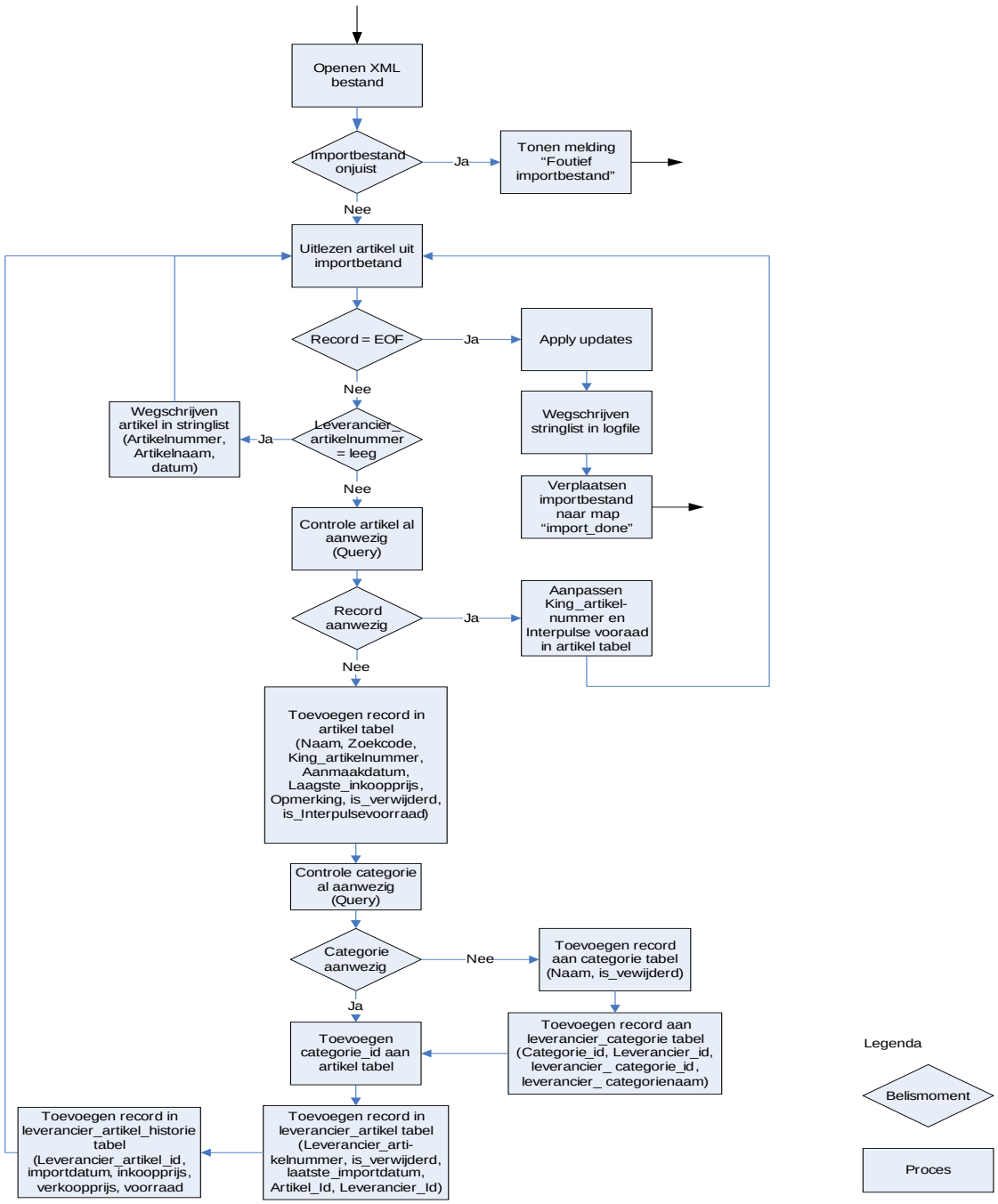
Voordat ik aan de bouw van de importfuncties begon, wilde ik eerst de importprocedures in kaart brengen. Dit om te voorkomen dat belangrijke onderdelen achteraf in de functie vergeten worden. Om de inhoud van de importprocedure weer te geven heb ik onderzoek gedaan naar welk model ik hiervoor gebruiken kon. Hierbij heb ik twee modellen bekeken. Dit zijn het sequencediagram uit UML en de proces flowchart die binnen Interpulse wordt gebruikt.

Sequencediagrammen worden gebruikt om interactie tussen objecten weer te geven. Hierbij wordt de nadruk gelegd op het tijdsaspect. Het diagram geeft overzicht op objectniveau. Voor de importfunctionaliteit wilde ik echter alleen de importprocedure in kaart brengen, waarvoor een sequencediagram eigenlijk te diep gaat. Een belangrijk argument om niet voor een sequencediagram te kiezen is het ontbreken van beslismomenten. Een dergelijk moment kan eventueel worden afgevangen met constraints in een sequencediagram. Dit komt het overzicht echter niet ten goede. Daarnaast is naar mijn mening het sequencediagram geen duidelijk diagram. Bij grote processen lopen er veel pijlen tussen de objecten. Een toekomstige lezer van het diagram zou hier mogelijk problemen mee ondervinden. Ik heb daarom besloten dit model niet te gebruiken.

Binnen Interpulse wordt een procesmodel gebruikt dat uitstekend op mijn eisen aansluit. Dit model wordt een proces flowchart genoemd. Een proces flowchart is een zeer eenvoudig model. Hierdoor zou een mogelijke toekomstige ontwikkelaar het diagram goed kunnen lezen en mogelijk zelfs uitbreiden. Daarnaast kunnen in dit diagram beslismomenten worden opgenomen. Het diagram bestaat uit de volgende onderdelen:

- Processen. In dit onderdeel worden de handelingen van het systeem weergegeven. De processen zijn in het diagram aangegeven met een rechthoekige afbeelding.
- Beslissingen. In dit onderdeel wordt een beslissing aangegeven. Vanuit dit onderdeel gaan twee pijlen (ja/nee). Een beslissing kan in Delphi gezien worden als een *if then else* aanroep.
- Pijlen. De pijlen geven de interactie tussen processen en beslismomenten weer.

Ook het tijdsaspect is in het diagram opgenomen. Door de pijlen te volgen kan gezien worden welke processen na elkaar worden aangeroepen. Hieronder word de flowchart van de King import beschreven.



Figuur: Proces flowchart King import

Zoals in het diagram is af te lezen wordt als een record al aanwezig is alleen het King artikelnummer en de voorraad bij Interpulse bijgewerkt. Als een artikel niet aanwezig is zullen alle tabellen worden gevuld.

De proces flowcharts zijn voor elk pilotdeel opgesteld. In de volgende paragraaf wordt beschreven hoe deze flowcharts geholpen hebben bij de bouw.

4.3.2 Bouw

De belangrijkste activiteit bij de bouw van deze pilot is het omzetten van de flowcharts in Delphi code. Naast deze activiteit zullen hieronder de activiteiten bij de bouw van de realtime check worden beschreven.

Omzetten flowcharts in Delphi code

Aan de hand van de proces flowcharts heb ik de bouw uitgezet. Als eerste heb ik alle processen en beslismomenten die in de flowchart voor komen in een grote procedure als commentaar geplaatst. Hiermee verkreeg ik meteen overzicht welke functies gebouwd diende te worden en welk proces op welk moment aangeroepen wordt. Vervolgens heb ik per proces de bijbehorende code geschreven. In onderstaand voorbeeld geef ik een codefragment dat is opgesteld aan de hand van de proces flowchart ‘King import‘ zoals beschreven in de vorige paragraaf.

```
//Check if the product already has been imported.
Leverancier_artikel_id :=
CheckKingArtikel(ArtikelLijst.ChildNodes['ART_LEVERANCIERS'].
ChildNodes['ART_LEVERANCIER'].ChildNodes['ART_LEVERANCIER_
ARTIKELNUMMER'].Text, 1);

//Product has been imported
if Leverancier_artikel_id <> 0 then
begin
   //Fill the Interpulse stock and the King productnumber
   datHM.OpenZoek_Artikel_Laatste_Import(Leverancier_artikel_Id);
   datHM.cdsZoek_Artikel_Laatste_Import.Edit;
   datHM.cdsZoek_Artikel_Laatste_Import.FieldByName
     ('is_Interpulsevoorraad').AsBoolean :=
     StrToBool(ArtikelLijst.ChildNodes['ART_VOORRAADARTIKEL'].Text);
     …
     datHM.cdsZoek_Artikel_Laatste_Import.Post;
end
//Product hasn’t been imported yet
else
begin
   //Add the new product
   datHM.OpenArtikel(Null);
   datHM.cdsArtikel.Append;
   datHM.cdsArtikel.FieldByName('Naam').AsString :=
              ArtikelLijst.ChildNodes['ART_OMSCHRIJVING'].Text;
   …
```

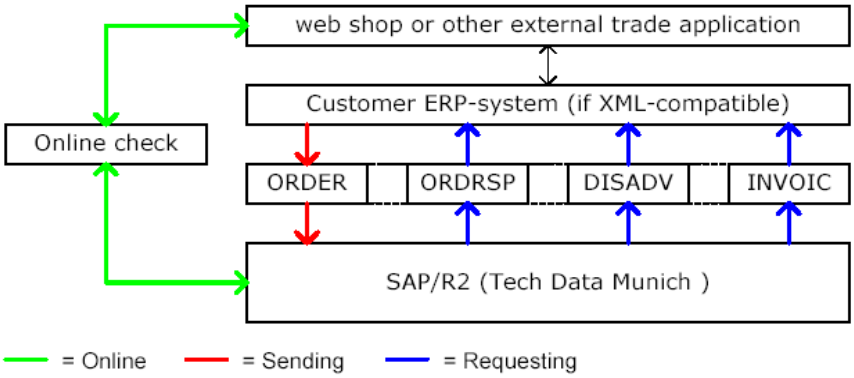
Figuur: Delphi code importeren King artikelen

In bovenstaand codefragment wordt middels een functie ‘CheckKingArtikel’ gecontroleerd of een artikel (gelezen uit een XML bestand) al in de database bestaat. Als dit het geval is wordt de Interpulse voorraad bijgewerkt. Als dit niet het geval is wordt het artikel toegevoegd.

Volgens mijn werkwijze heb ik eerst met commentaar (tekst beginnend met ‘//’) aangegeven wat geprogrammeerd diende te worden. Vervolgens is de code bijgeschreven. Deze manier geeft meteen inzicht in de structuur van de procedure.

Realtime check

De realtime check zorgt ervoor dat de gebruiker op elk moment van de dag de huidige prijs en voorraad bij Tech Data kan ophalen. Hiervoor maakt de Ipulse Hardware Manager met een Tech Data systeem, genaamd IntegrateX. IntegrateX is opgesteld om klanten van Tech Data de mogelijkheid te geven applicaties van klanten aan Tech Data te koppelen. Hieronder wordt een overzicht gegeven van de functionaliteiten van IntegrateX.



Figuur: Structuur IntegrateX

Voor de realtime check functionaliteit wordt alleen gebruik gemaakt van de Online check. Naast de online check kunnen via het sap systeem o.a. orders worden aangemaakt.

In eerste instantie dacht ik dat IntegrateX aangesproken kon worden met SOAP. Bij SOAP wordt er een aanvraag van een klant naar het SAP systeem verstuurd. De aanvraag is een vooraf vastgesteld XML bestand. Er wordt vervolgens een XML bestand met de gewenste data teruggestuurd. Na het bestuderen van documentatie over IntegrateX bleek het systeem echter geen gebruik te maken van SOAP, maar van iets dat er toch wel sterk op lijkt. De Online check module uit IntegrateX is aan te spreken via het volgende http bericht.

```
http://intouch.computer2000.com/xml/xmlonl.asp?custid=CustID&prodid=ProdID
```

Figuur: http bericht

In de link zal het klantnummer van Interpulse (CustID) moeten worden meegegeven. Daarnaast zal het artikelnummer(ProdID) van het artikel waarover meer informatie gewenst is worden meegegeven. Het volgende XML bestand wordt teruggestuurd:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<RealData><MSG ID="0">ok</MSG><ITEM>
              <ID>152</ID>
              <DESC>SCSI Cable 1m LDBL50 M/M</DESC>
              <PRICE CURRENCY="EUR">41,44</PRICE>
              <QTY>6</QTY>
              <DELDATE>26042004</DELDATE></ITEM></RealData>
```

Figuur: XML bestand uit IntegrateX

Het bovenstaande bestand wordt vervolgens uitgelezen en de gebruiker krijgt een melding met de huidige prijs en de huidige voorraad.

4.3.3 Beoordeling en test

Binnen deze pilot zijn naast de tussentijdse test verschillende andere tests uitgevoerd. Deze tests worden hieronder beschreven.

White Box test

Een fout in de import kan rampzalige gevolgen hebben voor bijvoorbeeld de orderverwerking. In een white box test worden alle mogelijke paden en invoer getest. Om geen fouten over het hoofd te zien heb ik besloten een white box test voor deze pilot uit te voeren. De test is per pilotdeel afzonderlijk opgesteld en uitgevoerd. Bij het opstellen van de test heb ik wederom gebruik gemaakt van de flowcharts. Uit de flowchart kunnen snel alle mogelijke paden worden herkend. De testcases heb ik verwerkt in een tabel. In deze tabel hoeft de tester (ikzelf) alleen aan te geven of de uitvoer is wat verwacht werd. Hieronder wordt hier een fragment van getoond.

TestCase	Verwachte uitvoer	Ok?
Testen van het importbestand	Artikelen en klanten	
Importbestand bevat lege regel • Artikelen • Klanten	Lege regel wordt overgeslagen, Import succesvol	 [x] [x]
Importbestand is beschadigd • Artikelen • Klanten	Melding ‘Het importbestand is niet juist’	 [-] [-]
Testen enkelvoudige gegevens	Artikelen en klanten	
Voer een string in met meer dan het toegestane aantal karakters • Artikelen • Klanten	Import succesvol. String wordt afgebroken bij het maximale aantal.	 [x] [x]
Voer een string in, waarin een cijfer voorkomt • Artikelen • Klanten	Import succesvol. Het cijfer wordt gewoon in de string opgenomen.	 [x] [x]

Figuur: White box test format

Black box test

Naast de white box test heb ik de import laten testen door een persoon die niet betrokken bij mijn project was. Deze persoon had toegang tot de in gebruik zijnde King database en kon dus data die echt gebruikt wordt importeren. Ik had om beveiligingsredenen slechts toegang tot een testdatabase. De importfunctie heeft de black box test zonder problemen doorstaan.

Performancetest

Een performance eis die aan het begin van het project was opgesteld was dat de import niet langer dan 5 minuten mocht duren. Ik heb de import daarom op performance getest. De import van 16.000 records duurde ruim 10 minuten. Hiermee werd de performance eis niet gehaald. Ik heb onderzocht hoe ik de import kon versnellen. Uit het bestuderen van Delphi literatuur zijn de volgende performanceversnellers gekomen:

- Plaatsen van indexen
- Apply updates op de juiste plaatsen

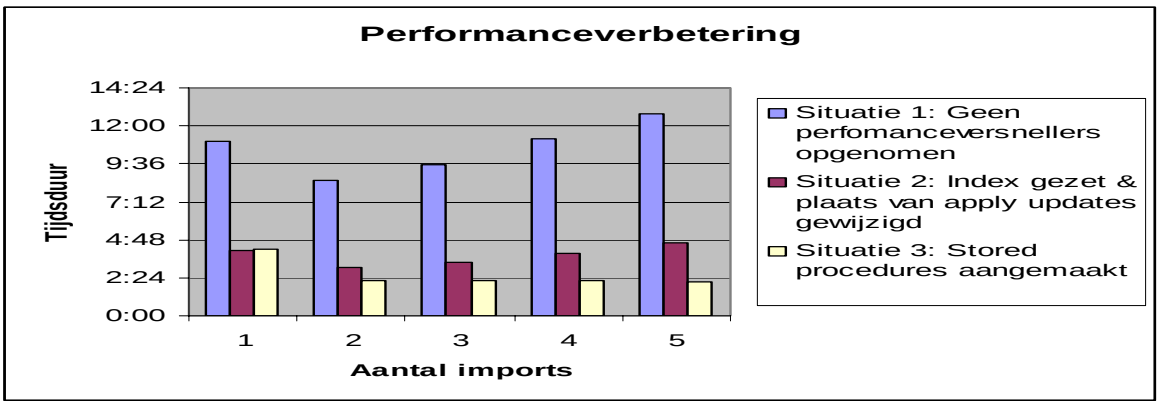
- Stored procedures

Een performanceversneller in het importproces is het plaatsen van indexen. Een index is een sortering op een veld uit een tabel. Er kan bijvoorbeeld een index fabrikantnaam worden gezet. Hiermee wordt de tabel op fabrikantnaam gesorteerd en kan een record aan de hand van de fabrikantnaam snel worden gevonden. Door een index op een tabel te plaatsen wordt het gebruik van specifieke en vaak langzame queries overbodig.

Een belangrijke reden voor de lage performance van de eerste versie van de Tech Data import was het verkeerd plaatsen van de ‘apply updates’ aanroep. Een apply updates zorgt ervoor dat de wijzigingen in de tabellen worden via het eerder beschreven DbExpress component weggeschreven naar de database op de server. Een eigenschap van DbExpress is de lage performance. Ik zal dit punt het probleem verder uitleggen aan de hand van de relatie tussen artikel en leverancier. Bij de King import wordt na elk geïmporteerd artikel een apply updates uitgevoerd. Bij de update word door SQL Server een id aan het artikel toegekend die benodigd is voor de tussentabel leverancier_artikel.
Deze aanpak werkte redelijk tot goed bij de King import. Bij de Tech Data import was deze aanpak echter niet goed bruikbaar. Alleen voor de artikelen werden al 16.000 apply updates uitgevoerd. Dit maakte de import traag.

Ik heb hiervoor de volgende oplossing gevonden. Na het starten van de import wordt eerst alleen de artikeltabel gevuld. Alle records uit het importbestand worden uitgelezen en in de artikeltabel opgeslagen. Na het uitlezen van het bestand wordt de apply updates aangeroepen. Het verschil met de vorige aanpak is dat de apply updates i.p.v. 16.000 keer slechts 1 keer wordt aangeroepen. De import is hiermee echter nog niet afgerond. Het importbestand wordt een tweede keer geopend. Vervolgens wordt de tussentabel van artikel en leverancier gevuld. Het juiste artikel_id wordt gevonden aan de hand van het fabrikantnummer in combinatie met de fabrikantnaam. Ook bij het vullen van de tussentabel wordt de apply updates als laatste uitgevoerd, waarbij er een totaal van 2 apply updates aanroepen worden uitgevoerd. Dit heeft de performance van de import aanzienlijk verbeterd.

Stored procedures zijn vooraf beschreven queries. Een stored procedure kan versnellen, omdat queries niet opnieuw opgebouwd hoeven worden. Ook hoeft de tabel niet vanuit Delphi geopend te worden, omdat de stored procedure in SQL Server wordt gedefinieerd. De historietabel wordt tijdens de import alleen gevuld met nieuwe prijzen en voorraadgegevens. Het nut van de historietabel is dat het prijsverloop later kan worden uitgelezen en worden getoond in een grafiek. Aangezien er geen andere bewerkingen dan het toevoegen op de tabel worden uitgevoerd heb ik een ‘insert’ stored procedure voor deze tabel opgesteld.



Figuur: Performanceverbetering Tech Data import

In de bovenstaande grafiek is per import de tijd af te lezen dat de import duurde. Hierbij is de performanceverbetering duidelijk zichtbaar. Een belangrijk punt hierbij is dat de import in situatie 3 op het gebied van tijd constant rond de 2 minuten blijft (ook bij latere tests). De andere imports werden juist door het voller worden van de database steeds trager.

4.3.4 Beschrijving iteratieslagen

Binnen deze pilot is een drietal iteratieslagen uitgevoerd. Bij de eerste iteratieslag werd de globaal functionele structuur en de globaal technische structuur in het pilotontwerp beschreven. Ik heb ervoor gekozen elk pilotdeel in een aparte iteratieslag te ontwikkelen. Dit betekent dat de iteratieslag per pilotdeel bestaat uit het opstellen van detailspecificaties, de bouw en het testen.
De volgende iteratieslagen zijn gemaakt:

1e iteratie:
- Globaal pilotontwerp
- King import artikelen

2e iteratie:
- King import klanten

3e iteratie:
- Realtime check

4e iteratie:
- Tech Data import

Opvallend aan bovenstaande iteraties is dat de basiseis ‘Tech Data import’ in de 4e iteratie is uitgevoerd. De comforteis van de realtime check is zelfs eerder uitgevoerd. Ik heb dit gedaan om de volgende reden. Voor het importeren van Tech Data gegevens moeten deze gegevens worden gedownload van de Tech Data ftp. Deze ftp is alleen bereikbaar als een account bij Tech Data is aangemaakt. Interpulse had de account nog niet aangemaakt. Het aanmaken van de account duurde zo lang (3 weken) dat ik eerst de realtime check en onderdelen uit de derde pilot heb ontwikkeld en gebouwd. Ik heb dit gedaan om geen tijdverlies op te lopen.

De in de white box test gevonden bugs zijn tevens in een iteratieslag opgelost. In de laatste iteratieslag is de performance verbeterd.

4.4 Pilot 3: Overzicht en beheer

De beheer- en overzichtpilot heeft als doel overzicht te geven in de geïmporteerde en handmatig toegevoegde data. Deze data moeten ook handmatig beheerd kunnen worden. Onder beheer valt het toevoegen, wijzigen en verwijderen van stamgegevens. Daarnaast wordt in deze pilot het probleem betreffende het opzoeken van artikelen opgelost. De volgende pilotdelen zijn in deze pilot uitgevoerd:

- Totaaloverzicht per stamgegeven (*basis*)
    - Overleg met de opdrachtgever (vaststellen gegevens in overzichten)

    - Programmeren code (opstellen en koppelen queries, opstellen benodigde code)
    - Aanpassen lay-out (King look-and-feel)

- Detailoverzichten (*basis*)
    - Overleg met de opdrachtgever (vaststellen gegevens in overzichten)
    - Opstellen gespecificeerd navigatieschema
    - Bouw van detailschermen (inclusief prijs- en voorraadgrafiek)
    - Programmeren code (opstellen en koppelen queries, opstellen benodigde code)
    - Opstellen rapport voor overzicht orders
    - Aanpassen lay-out (King look-and-feel)

- Handmatig toevoegen/wijzigen verwijderen stamgegevens (*basis*)
    - Programmeren code (opstellen en koppelen queries, opstellen benodigde code)

- Filter- en zoekfunctionaliteit (*basis*)
    - Overleg met de opdrachtgever
    - Programmeren code

- Afstellen grids (*comfort*)
    - Aanpassen grids (kleurverloop, sorteringfunctie, kolommen)

- Integreerfunctionaliteit voor artikelen en categorieën (*luxe*)
    - Opstellen proces flowchart
    - Programmeren code

Een oplettende lezer merkt op dat de pilotdelen anders zijn ingedeeld dan in de definitiestudie. Een aantal beheer functies zijn in de definitiestudie als comfort- en luxe-eis opgenomen. Aangezien alle beheerfuncties veel op elkaar lijken koste het weinig moeite deze tegelijk te realiseren. Hierdoor zijn alle beheereisen naar de basiseisen verhuisd. Daarnaast is een extra functionaliteit opgenomen. Dit is de integreerfunctionaliteit. Zoals eerder afgesproken worden nieuwe functionaliteiten als luxe eis opgenomen.

4.4.1 Pilotontwerp

Aan het begin van deze pilot zijn in overleg met de opdrachtgever de systeemeisen voor deze pilot nogmaals bekeken en uitgeschreven. Deze activiteit is uitgevoerd om verkeerde interpretatie van een eis te voorkomen. De systeemeisenlijst is bij deze activiteit met een extra eis uitgebreid.

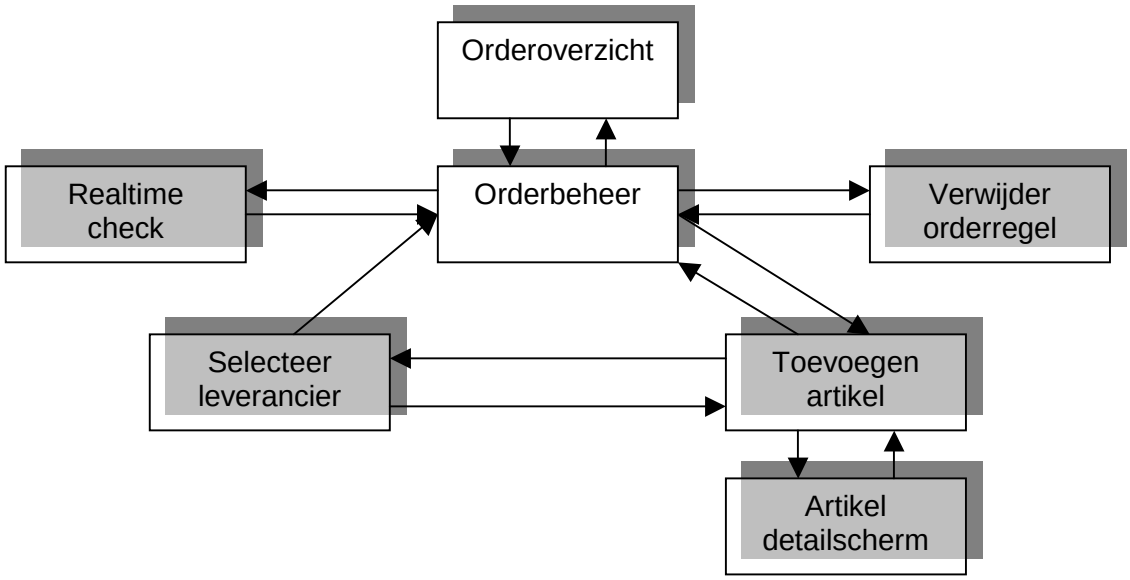
Het pilotontwerp is in vergelijking tot het ontwerp van de importpilot een dun rapport. Dit heeft tot reden dat er minder detailspecificaties zijn opgesteld. Deze pilot is ontwikkeld d.m.v. prototyping. De functionaliteiten waren zo duidelijk dat aan de hand van de systeemeisen lijst en het eerder opgestelde prototype de pilotdelen gerealiseerd konden worden. Het voordeel van deze manier van werken is het snelle resultaat.

Door minder detailontwerp wordt de overdraagbaarheid wel minder. Ik heb dit afgevangen door in de programmeercode veel commentaar op te nemen, zodat een toekomstige ontwikkelaar kan achterhalen hoe dingen zijn opgelost en (belangrijker nog) waarom dingen zo zijn opgelost.

Toch heb ik in deze fase nog een tweetal modellen opgesteld. Dit betreft het opstellen van een model voor het navigatieschema van de orderverwerking en een proces flowchart voor de integratiefunctie. Ik heb deze modellen opgesteld om meer inzicht te krijgen in hoe de functionaliteit te implementeren. Flowcharts zijn al eerder in dit verslag beschreven. Ik zal deze daarom hier niet verder beschrijven. Meer informatie over het pilotontwerp kan gevonden worden in de externe bijlage ‘Pilotontwerp beheer en overzicht’.

Gespecificeerd navigatieschema orderverwerking

In het overleg met de opdrachtgever aan het begin van deze pilot kwam naar voren dat hij met zo min mogelijk handelingen een order wilde samenstellen. De order moest een simpele order zijn, waarin alleen een klant en een aantal orderregels zijn opgenomen. Voor meer detailgegevens wordt de order naar King geëxporteerd en daar verder verwerkt. Om de schermen niet onoverzichtelijk te maken heb ik gekozen een soort van wizard te realiseren, waarin de gebruiker met simpele stappen een order samenstelt. Het resultaat is het volgende schermverloop.



Figuur: Gespecificeerd navigatieschema

Zoals hierboven te zien is wordt begonnen in het orderoverzicht scherm. Dit is het beheerscherm van de applicatie, waarbij het tabblad voor orders is geselecteerd. Als een order wordt toegevoegd wordt het orderregel overzicht getoond. Op dit scherm kan de klant worden geselecteerd. Via het ‘selecteer artikel’ en ‘selecteer leverancier’ scherm kan een artikel aan het orderregel overzicht worden toegekend. De realisatie van deze schermen wordt later in dit hoofdstuk beschreven.

4.4.2 Bouw

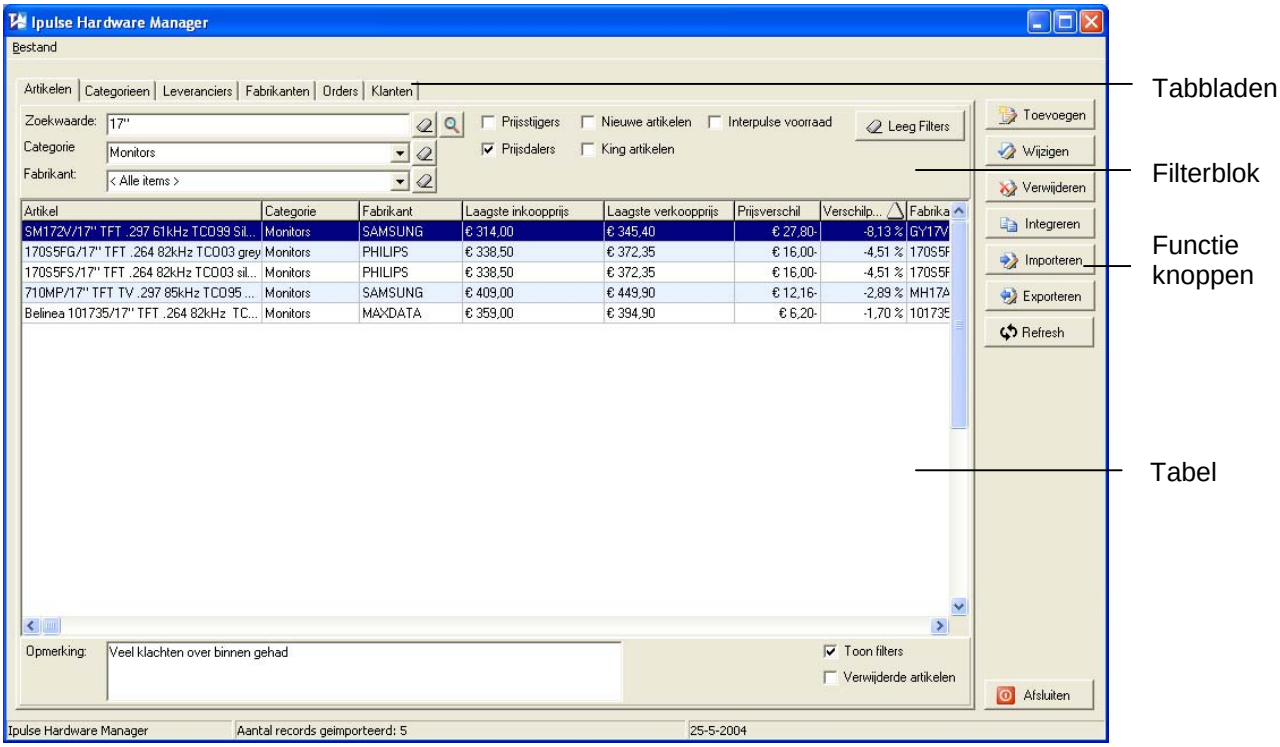
Bij de bouw was de opdrachtgever nauw betrokken. Ten minste één keer per week kwam de opdrachtgever kort langs om de voortgang te bekijken en bijsturing te geven. Zoals eerder beschreven heb ik binnen deze pilot veel gebruik gemaakt van prototyping. De functionaliteit evalueerde hiermee naar het uiteindelijke resultaat. Hieronder worden de belangrijkste overzichten beschreven.

Totaaloverzichten met zoek- en filterfunctie

De pilotdelen ‘totaaloverzichten’, ‘afstellen grids’ en ‘filter- en zoekfunctionaliteit’ zijn in dezelfde iteratie ontwikkeld. Ik heb dit gedaan, omdat de totaaloverzichten overzichtelijker worden als de grids zijn afgesteld en de zoekfuncties zijn geïmplementeerd.

Voor het afstellen van de grids heb ik gebruik gemaakt van functies die al ik al eerder bij het oefenproject Grafi Magazijn had geschreven. Deze code was zodanig herbruikbaar dat grote delen konden worden hergebruikt. Zo is bijvoorbeeld de code voor de kleurwisseling en de sortering van de tabel (grid) dezelfde code die ik bij Grafi Magazijn had gebruikt.

In deze iteratie is het probleem van het niet snel kunnen opzoeken van een artikel opgelost. Als een klant bijvoorbeeld vraagt om een 17” monitor, kan via de zoek- en filterfunctie snel een gewenst product worden gevonden. Hierbij kan meteen worden gekeken welke monitor goedkoop is, of welke printer juist in prijs is gedaald of gestegen.

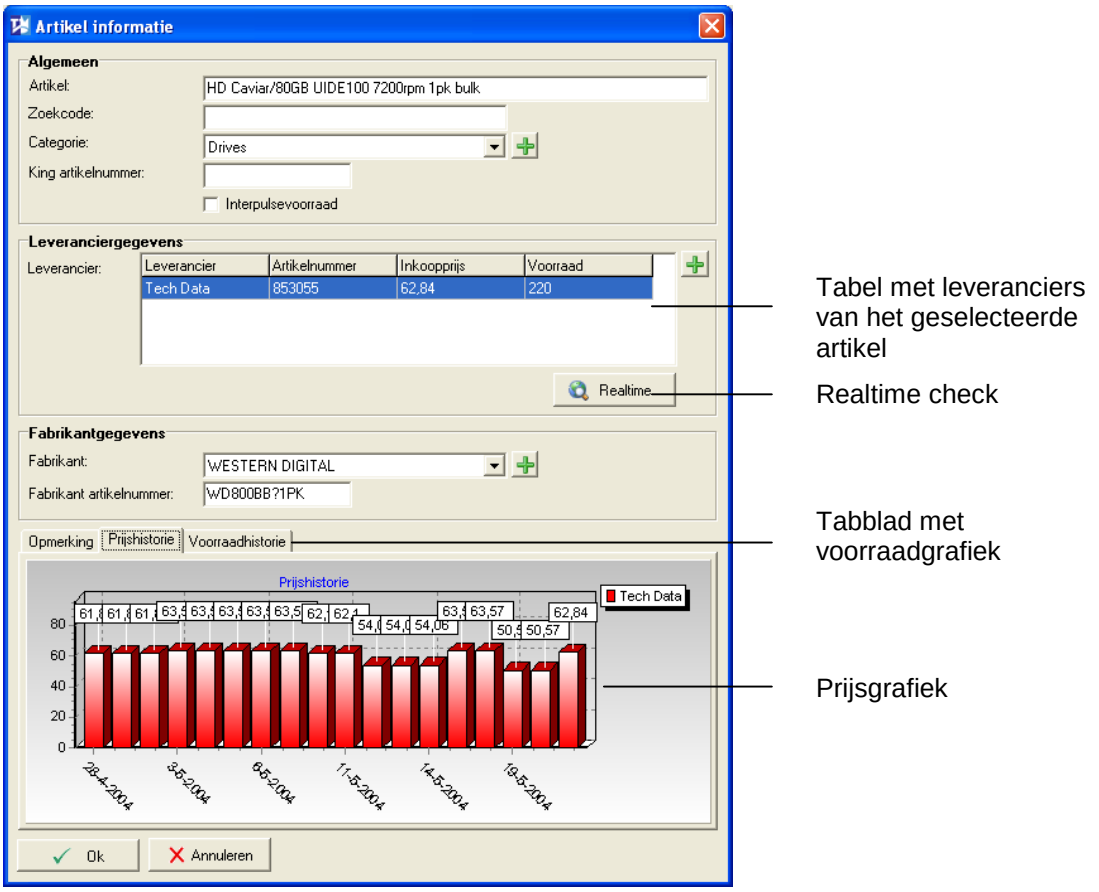


Figuur: Totaal overzicht artikelen

Detailoverzichten en handmatig toevoegen/wijzigen en verwijderen gegevens

In dit pilotdeel zijn de detailoverzichten per stamgegeven opgesteld. Detailschermen worden verkregen door op een stamgegeven uit het totaaloverzicht te dubbelklikken. Met deze detailoverzichten kunnen gegevens worden beheerd. Door de overlap van de pilotdelen heb ik besloten het beheren van gegevens en de detailoverzichten samen te bouwen. De belangrijkste detailschermen zijn die van de artikelen en orders.

In het artikelscherm wordt de prijs- en voorraadhistorie getoond. In overleg met de opdrachtgever is besloten deze historie in grafiekvorm weer te geven. Het grafiekcomponent bleek een lastig component. Om een mooie en correcte grafiek te tonen heb ik een deel van de vastgestelde timebox besteed aan onderzoek. Het onderzoek is middels Delphi literatuur en een testproject uitgevoerd. Met de grafiek wordt het probleem in prijsinzichten opgelost.



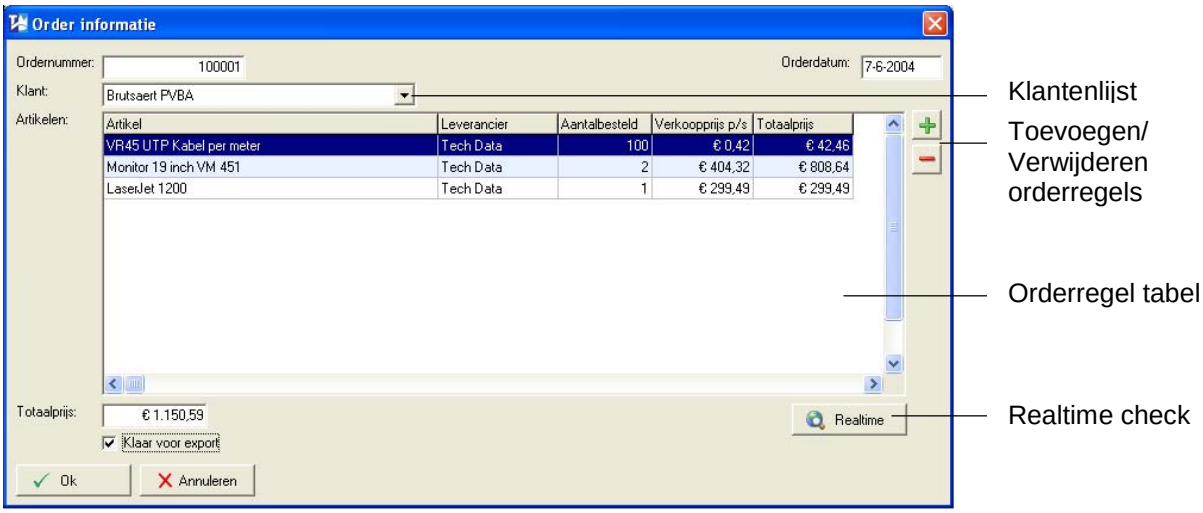
Figuur: Artikel beheerscherm

Via dit scherm kunnen ook nieuwe artikelen worden toegevoegd en gewijzigd. Per stamgegeven heb ik een soortgelijk scherm gebouwd.

Orderverwerking

Zoals eerder omschreven wordt voor de orderverwerking gebruik gemaakt van wizard, waarmee in een aantal stappen een order kan worden samengesteld. Ik heb dit gedaan om te voorkomen dat de eindgebruiker door bijvoorbeeld drukke schermen (met veel gegevens) het overzicht kwijtraakt.

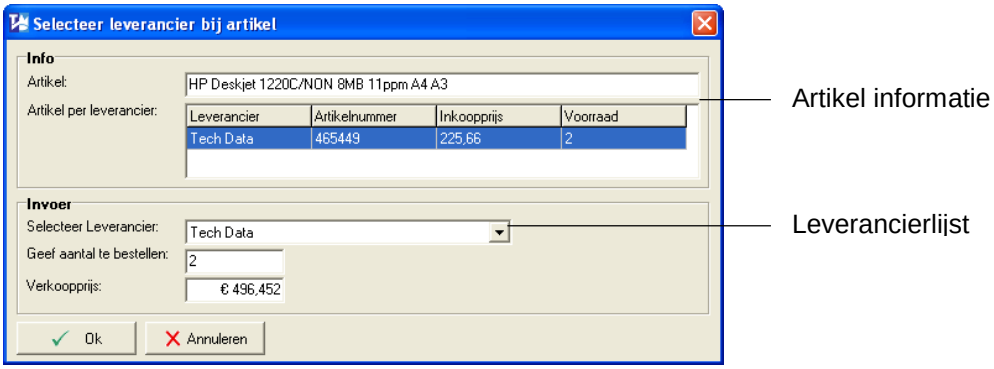
De eerste stap is het toevoegen van orders. Hieronder is het orderscherm te zien. In dit scherm kan een order aangemaakt worden door een klant en orderregels toe te voegen.



Figuur: Orderbeheer scherm

Vervolgens dient een artikel geselecteerd te worden. Voor dit scherm heb ik hetzelfde scherm, inclusief filter- en zoekfunctie, gebruikt als bij het artikel totaaloverzicht. Het doel van beide schermen is identiek, namelijk het opzoeken van artikelen. Het voordeel van het gebruiken van hetzelfde scherm is dat achterliggende code hergebruikt kan worden.

Als laatste stap wordt een leverancier aan de order toegekend. Daarnaast kan het aantal te bestellen artikelen worden aangegeven. Dit gaat via het volgende scherm.



Figuur: Leverancier selecteren

4.4.3 Beoordeling en test

Aangezien na de bouw de tijd in de timebox van deze pilot bijna op was heb ik geen white box test uitgevoerd. Daarnaast bevat deze pilot zoveel (kleine) functionaliteiten dat het erg veel werk is om alle mogelijke input via een testplan te testen. Ik heb daarom elke functionaliteit zelf kort getest en ben vervolgens aan een volgende functionaliteit uit de pilot begonnen. Als laatste test van deze pilot heb ik een persoon, die niet betrokken bij het project was, alle functionaliteiten (ook in combinatie met de functies uit andere pilots) middels een black box test laten testen. Aangezien deze persoon vaak door Interpulse gebouwde applicaties test heb ik de inhoud van de test aan deze persoon zelf overgelaten. Uit deze test zijn een aantal (kleine) bugs naar voren gekomen. Deze zijn echter niet noemenswaardig.

De opdrachtgever heeft de pilot na de tests beoordeeld. Uit deze beoordeling kwamen een aantal wensen naar voren. Zo vond de opdrachtgever de zoekfunctie nog niet uitgebreid genoeg. Ik heb de wensen van de opdrachtgever genoteerd en in een later stadium (na het uitvoeren van de vierde pilot) gerealiseerd.

4.4.4 Beschrijving iteratieslagen

Binnen deze pilot waren veel pilotdelen opgenomen met een kleine doorlooptijd. Een aantal van deze pilotdelen waren nauw gerelateerd aan andere pilotdelen. Ik heb om deze reden deze pilotdelen in dezelfde iteratie ontwikkeld. Via de volgende iteraties is de pilot gerealiseerd:

1e iteratie:
- Bouw totaaloverzichten
- Afstellen grids
- Bouw filter- en zoekfunctie

2e iteratie:
- Bouw detailschermen
- Toevoegen, wijzigen en verwijderen stamgegevens

3e iteratie:
- Bouw orderverwerking

4e iteratie:
- Verbeteren filter- en zoekfunctie
- Verbeteren orderverwerking
- Bouw integratieschermen

De in de black box test gevonden bugs zijn tevens opgelost in een eigen iteratie. In totaal zijn er dus vijf iteraties uitgevoerd. De vierde en vijfde iteratie zijn uitgevoerd nadat de laatste pilot ‘export’ was doorlopen. Ik heb deze iteraties doorlopen, omdat ik na het verlopen van de timebox nog niet tevreden met de functionaliteiten uit deze pilot was. Zo was bijvoorbeeld de zoekfunctie hoofdlettergevoelig. Ik heb dit aangepast, zodat artikelen ook zonder hoofdletters in de zoekwaarde gevonden kunnen worden.

4.5 Pilot 4: Export

De vierde en laatste pilot export heeft als doel orders en artikelen (die op die orders verwerkt zijn) te exporteren, zodat ze in King kunnen worden geïmporteerd. Hiermee is het handmatig invoeren van artikelen en orders niet meer nodig.

Ik heb de export opgedeeld in twee pilotdelen, die elk in een eigen iteratie worden ontwikkeld. Dit zijn de volgende pilotdelen:

- Exportfunctionaliteit artikelen *(basis)*
- Exportfunctionaliteit orders *(basis)*

Het derde pilotdeel voor de exportfunctionaliteit van klanten is niet uitgevoerd. Er was genoeg tijd voor dit pilotdeel binnen de timebox over. Ik heb echter besloten meer urgente functies uit andere pilotdelen eerst te implementeren. De activiteiten die

uitgevoerd moeten worden om bovenstaande pilotdelen te realiseren zijn gelijk. Ik heb per pilotdeel de volgende activiteiten uitgevoerd:

- Overleg met de opdrachtgever
- Bestuderen King importformaten / structuur van importbestanden
- Opstellen proces flowchart voor export
- Bouwen exportfunctionaliteit aan de hand van flowchart
- Testen

4.5.1 Pilotontwerp

In de vorige pilot was er veelvuldig interactie met de opdrachtgever. De export pilot lijkt op de import pilot, waarin veel onder de motorkap van het systeem gebeurd. Om deze reden is het zinloos de opdrachtgever nauw te betrekken bij de bouw deze pilot.

Om de opdrachtgever toch enigszins bij de pilot te betrekken heb ik aan het begin van de pilot een kort overleg met de opdrachtgever gevoerd. Aan dit overleg ging een korte studie naar King importformaten (aan de hand van de King helpfile) vooraf. In dit onderzoek heb ik een aantal ‘problemen’ ontdekt. In het overleg heb ik deze problemen aan de opdrachtgever voorgelegd en in overleg zijn hier oplossingen voor gevonden. Hieronder een samenvatting van dit overleg:

Probleem 1:

Een artikel, welke is opgenomen op een order, dient voorzien te zijn van een King artikelnummer. Als deze geen nummer heeft, wordt de order niet door King geaccepteerd. Tevens moet dit nummer ook in King bekend zijn.

Oplossing:

Het toekennen van een King nummer zal automatisch door de Ipulse Hardware Manager worden uitgevoerd, bij het toekennen van een artikel aan een order. Hierbij houdt de Ipulse Hardware Manager een eigen range van nummers bij, zodat geen dubbele artikelnummers voor komen. Aangezien het nummer ook in King bekend moet zijn moet altijd eerst het artikelbestand worden geïmporteerd in King. Daarna kan het orderbestand worden geïmporteerd.

Probleem 2:

Een categorie van een artikel dient in King bekend te zijn. Als een artikel een categorie bevat die niet in King bekend is kan deze niet worden geïmporteerd, waardoor een order ook niet kan worden geïmporteerd.

Oplossing:

In de Ipulse Hardware Manager worden eenmalig handmatig de categorienummers uit King bij de bestaande categorieën ingevoerd. Een andere manier is de King categorieën te importeren en via de integreerfunctie samen te voegen met de bestaande categorieën die wel al zijn voorzien van een King nummer.

De exportpilot lijkt inhoudelijk sterk op de importpilot. Ik heb daarom gebruik gemaakt van dezelfde techniek om het proces te modelleren als die ik met succes voor de importpilot heb gebruikt, namelijk via een proces flowchart.
Aangezien proces flowcharts al eerder in dit verslag zijn beschreven zal dit model niet nogmaals worden behandeld.

4.5.2 Bouw

Tijdens de bouw is de proces flowchart van de export omgezet in Delphi code. Dit zoals eerder is beschreven bij de importpilot.

In Delphicode wordt het XML bestand opgemaakt. Het XML bestand moet voldoen aan de vastgestelde voorwaarden uit King. Zo moeten de tags dezelfde namen hebben en op dezelfde diepte zijn geplaatst. Het aangeven van diepte wordt gedaan door ‘childs’ toe te voegen. Hieronder wordt hier een voorbeeld van gegeven:

```
NewArtikel := NewArtikelen.AddChild('ARTIKEL');

ValueNode := NewArtikel.AddChild('ART_NUMMER');
ValueNode.Text := datHM.cdsExportArtikelen.FieldByName('King_artikelnummer').
AsString;

ValueNode := NewArtikel.AddChild('ART_OMSCHRIJVING');
ValueNode.Text := datHM.cdsExportArtikelen.FieldByName('ArtikelNaam').AsString;
```

```
- <ARTIKEL>
    <ART_NUMMER>500001</ART_NUMMER>
    <ART_OMSCHRIJVING>Cable/power IEC>IEC 3m</ART_OMSCHRIJVING>
```

Figuur: Opstellen XML bestand in Delphi – code en uitwerking

Bovenstaande manier van opbouwen is voor beide exportbestanden gehanteerd.

4.5.3 Beoordeling en test

In eerste instantie heb ik zelf een test uitgevoerd. Met deze test wilde ik onderzoeken hoe King reageert op bepaalde XML bestanden. Ik heb hierbij verschillende juiste en foutieve XML bestanden in King geïmporteerd. Tijdens deze test kwam ik een paar kleine foutjes tegen. Deze fouten zijn echter niet noemenswaardig en zijn meteen opgelost.

Naast een eigen test heb ik een onafhankelijk persoon de exportfunctie laten testen. Uit deze test kwamen geen fouten naar voren.

4.5.4 Beschrijving iteratieslagen

Binnen deze pilot heb ik twee iteratieslagen gemaakt. In deze iteratieslagen zijn de basiseisen gerealiseerd. In de timebox van deze pilot was nog voldoende tijd over om de exportfunctie voor klanten (comforteis) te realiseren. Ik was echter nog niet tevreden over de uitwerking van de derde pilot. De derde pilot was niet ruim genoeg gepland, waardoor de werkzaamheden afgebroken zijn. Ik heb besloten eerst de derde pilot goed af te ronden i.p.v. een lager geprioriteerde comforteis van de export pilot te realiseren. Uiteindelijk was er niet genoeg tijd meer over om deze comforteis te realiseren. De gemaakte iteratieslagen zijn de volgende:

1e iteratie:
- Exportfunctionaliteit artikelen

2e iteratie:
- Exportfunctionaliteit orders

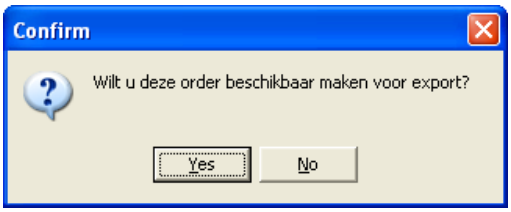
Ook na de testronde werd er een iteratie uitgevoerd. In deze iteratie werden de in de testronde gevonden fouten opgelost.

4.6 Algehele en multi-user test

Aan het begin van het project had ik gedacht aan een grote white box test om het gehele systeem te testen. Met deze test kan worden nagelopen of afzonderlijke pilots ook goed met elkaar kunnen functioneren. Wegens tijdgebrek is deze test helaas niet uitvoerig uitgevoerd. Aan het einde van het project is er zoveel tijd besteed aan het optimaliseren van de applicatie dat de algehele test niet binnen de afstudeerperiode is uitgevoerd. Ook is er geen testplan voor de algehele test opgesteld. Wel heb ik zelf middels een black-box test alle functionaliteiten nagelopen. Ook combinaties van belangrijke functionaliteiten zoals de import, orderverwerking en export heb ik getest.

Ik heb wel een multi-user test uitgevoerd. Het doel van deze test is het controleren of het systeem ook functioneert als meerdere gebruikers tegelijk op de database werken. Met deze test heb ik meerdere clients geopend en deze clients tegelijk bewerkingen laten uitvoeren op dezelfde records. De bewerkingen die ik heb uitgevoerd zijn het tegelijk toevoegen, wijzigen, wijzigen terwijl de andere client het record heeft verwijdert, importeren en exporteren. Zoals eerder beschreven wordt het muteren op hetzelfde record via het reconcile error mechanisme afgehandeld. Het systeem reageerde tijdens deze test zoals verwacht. Hierbij is één noemenswaardige bug gevonden. Dit betreft de volgende:

In een multi-user omgeving kan het voor komen dat één client bezig is met een order, terwijl een andere client de export uitvoert. Hierbij zal de nog niet afgeronde order ook worden geëxporteerd. Door een extra veld, ‘klaarvoorexport’, in de databasetabel orderhoofd op te nemen werd dit probleem afgevangen. De gebruiker zal dan na het opstellen van een order aangeven of de order afgerond is.



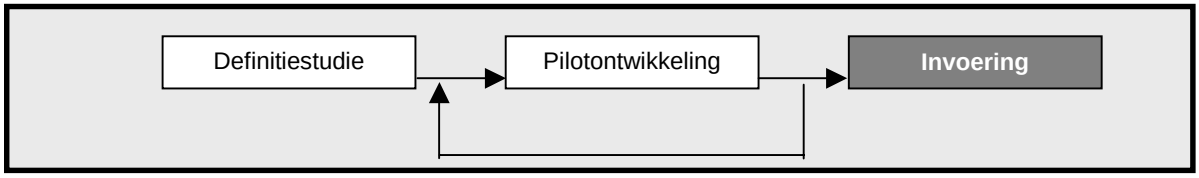
Figuur: Order beschikbaar maken voor export

Na de hierboven genoemde tests en de afzonderlijke tests per pilot werd het systeem door de opdrachtgever voldoende geacht om een eerste versie te implementeren. Ook zelf heb ik het volste vertrouwen in dit systeem. Er zullen ongetwijfeld kleine bugs en nieuwe features worden gemeld. Ik denk echter dat dit onvermijdelijk is bij software.

In het oplossen van bugs en het inbouwen van nieuwe features door een andere programmeur voorzie ik geen problemen. Ik heb bij het programmeren op veel plaatsen rekening gehouden met de leesbaarheid van mijn code. Zo is er op veel plaatsen commentaar opgenomen om de leesbaarheid te vergroten. Een Delphiprogrammeur moet hier zijn weg in kunnen vinden.

5. Invoering

De invoeringsfase heeft als doel het complete systeem operationeel te maken in de gebruikersomgeving. Het is niet mogelijk en gewenst om de Ipulse Hardware Manager in stappen in te voeren. Daarom zal de fase invoering slechts eenmaal doorlopen worden. In de fase invoering staan de acceptatietest en de overdracht en beheer centraal.



Figuur: IAD traject

5.1 Acceptatietest

Aan het einde van het project heeft de opdrachtgever een acceptatietest uitgevoerd. Tijdens deze test heeft de opdrachtgever gecontroleerd of het systeem voldoet aan de vooraf gestelde eisen. Aangezien de opdrachtgever gedurende het gehele project nauw betrokken bij de ontwikkeling van het pakket is geweest, was de opdrachtgever al redelijk thuis in het systeem. Het systeem voldeed aan de verwachtingen van de opdrachtgever en is daarmee geaccepteerd.

5.2 Overdracht en beheer

Na de acceptatietest is de Ipulse Hardware Manager bij de opdrachtgever geïmplementeerd. Hierbij is de database verplaatst van de testserver naar de in gebruik zijnde server. Voor de client is een installatiebestand in InstallShield gemaakt. InstallShield is een programma, waarmee installatiebestanden worden gemaakt. InstallShield wordt o.a. gebruikt in Microsoft producten. Via het installatiebestand is de Ipulse Hardware Manager client vervolgens op de pc’s van de accountmanagers geïnstalleerd.

Naast de installatie van het pakket heb ik de ontwerpdocumentatie aan de opdrachtgever overhandigd. Deze documentatie kan helpen bij het aanpassen en mogelijk uitbreiden van het pakket in de toekomst.

Mochten er in de toekomst bugs worden gevonden verwacht ik dat een gemiddelde Delphi programmeur deze kan oplossen. In de code is namelijk veel commentaar opgenomen die het lezen van de code duidelijker moet maken.

Het programma is zo duidelijk in het gebruik dat een handleiding naar mijn mening niet nodig is. Ik heb dit getoetst door gedurende het project een aantal onafhankelijke personen het programma te laten testen. Deze personen konden moeiteloos met het pakket omgaan en ondervonden eigenlijk geen problemen. Er is daarom in overleg met de opdrachtgever besloten geen gebruikershandleiding op te stellen.

6. Evaluatie

In dit laatste hoofdstuk van mijn afstudeerverslag evalueer ik mijn afstudeerperiode. Hiermee krijgt de lezer inzicht in wat ik zelf van het geleverde werk vind. Als eerste evalueer ik de producten die ik aan de opdrachtgever heb opgeleverd. Vervolgens zal ik het door mij gevolgde proces evalueren.

6.1 Productevaluatie

In deze paragraaf vergelijk ik de door mij opgeleverde producten met de in de opdrachtomschrijving genoemde op te leveren producten. Hiermee moet duidelijk worden in hoeverre de opgeleverde producten aan de gestelde eisen voldoen. Tevens beschrijf ik hoe de eindproducten in de organisatie zijn ontvangen. Deze zaken beschrijf ik per product, zoals deze in de opdrachtomschrijving zijn opgenomen.

6.1.1 Plan van aanpak algemeen

Het plan van aanpak is aan het begin van het project door de opdrachtgever goedgekeurd en is de gehele periode gebruikt bij de evaluatie van de planning en het bewaken van de voortgang. Hierdoor is dit product leidraad geworden van mijn afstudeeropdracht. Ik ben van mening dat de inhoud van dit plan van aanpak voldaan heeft om de opdracht tot een goed einde te brengen.

6.1.2 Ontwerprapport en systeem Grafi Magazijn

Om tijd te besparen ben ik meteen begonnen met het ontwerprapport voor Grafi Magazijn. Het plan van aanpak is niet opgesteld. Dit gaf geen problemen, omdat het project zo klein is dat geen extra documentatie nodig is. Het ontwerprapport voor de voorbereidende opdracht Grafi Magazijn is naar mijn mening inhoudelijk niet volledig voor grotere projecten. Bij het opstellen van het systeem was haast geboden. Om deze reden heb ik alleen de belangrijke onderdelen uitgewerkt. Dit is overigens ook de manier, waarop Interpulse op dit moment werkt. Grafi Magazijn is een klein en relatief simpel systeem, waardoor de uitgewerkte modellen voldoende waren om het systeem te realiseren. Ik ben daarom toch tevreden over dit rapport.

Ook over het systeem Grafi Magazijn ben ik tevreden. Het systeem is opgesteld uit een mondelinge opdrachtomschrijving, waar ik alle kanten mee uit kon. Om hier een systeem uit te vervaardigen was een uitdaging voor mij. Het product is aan het einde van het project bij de klant afgeleverd en is in gebruik genomen. Een leuk feitje hierbij is dat er geen enkele bug is gemeld na de oplevering.

Deze voorbereidende opdracht heeft ervoor gezorgd dat ik snel nuttige informatie opdeed op het gebied van ontwikkelmethode, ontwikkeltechniek en het projectwerk binnen Interpulse. Het alternatief was een literatuurstudie. Ik ben van mening dat in de praktijk sneller en meer nuttige informatie wordt opgedaan dan uit een boek. Een goed voorbeeld hiervan is het use case diagram. Uit het boek werd mij niet goed duidelijk, waarom dit diagram gebruikt werd. Tijdens het echt opstellen van dit model werd het doel me meteen duidelijk. Concluderend kijk ik met tevredenheid terug op deze opdracht.

6.1.3 Definitiestudie

De definitiestudie is gemaakt om de opdracht vast te stellen en een eerste aanzet tot een oplossing te geven. De betrokkenheid van de opdrachtgever was in deze fase van groot belang. Daarom heb ik interviews gehouden, welke nuttige informatie opleverde. Daarnaast heb ik middels een brainstormsessie met de opdrachtgever de systeemeisen vastgesteld. Dit soort momenten zal ik ook zeker in de toekomst terug laten komen. Met de informatie uit de interviews heb ik naar mijn mening een goede definitiestudie kunnen opleveren.
De definitiestudie is in de beginfase vaak aangepast. Zo heb ik bijvoorbeeld het klassendiagram gedurende de databaseontwikkeling een aantal keer gewijzigd. Om geen tijdverlies op te lopen heb ik besloten na de definitiefase geen wijzigingen op de definitiestudie meer uit te voeren. Hierdoor is de definitiestudie niet meer up-to-date. Het rapport bevat een concept waar het project op verder is gebouwd en heeft daarmee z’n doel bereikt. De pilotontwerprapportages vullen de definitiestudie echter genoeg aan. Als de definitiestudie wordt gelezen in combinatie met de pilotontwerprapportages krijgt de lezer wel een volledig en up-to-date beeld. Hiermee verwacht ik dat de overdraagbaarheid niet in gevaar is gekomen.

Een onderdeel binnen de definitiestudie waar ik niet tevreden over ben is het plan van aanpak. Dit onderdeel is een kleinere versie van het plan van aanpak algemeen en heeft eigenlijk geen toegevoegde waarde. Ik heb dit rapport opgeleverd, omdat het rapport in de IAD literatuur beschreven stond. In eerste instantie had ik ook voor elke pilot een plan van aanpak ingepland. Ook deze zouden waarschijnlijk, net als het plan van aanpak voor de definitiestudie, kleinere versies worden van het algemene plan van aanpak. Ik heb dan ook besloten de plannen van aanpak in de pilotontwikkelfase niet op te stellen.

6.1.4 Pilotontwerprapportages

Het opmaken van de pilotontwerprapportages vond ik een relatief lastige klus. Vooral het vinden van geschikte modellen per pilot was in sommige gevallen moeilijk. Hier kom ik later in dit hoofdstuk op terug. Ik denk toch dat ik hier goed in geslaagd ben. De door mij opgestelde modellen hebben mij voldoende geholpen bij de bouw van de applicatie.

Ik heb getracht de rapporten zo duidelijk mogelijk te laten zijn. Zo heb ik bijvoorbeeld aan het begin van elk rapportage een duidelijke uitleg gegeven wat het doel van de pilot is. Dit was voor mijzelf omslachtig. Toch denk ik dat het verstandig is geweest deze uitleg op te stellen. Het doel hiervan is namelijk dat een toekomstige ontwikkelaar ook met het rapport uit de voeten kan. De overdraagbaarheid wordt hiermee vergroot.

6.1.5 Systeem Ipulse Hardware Manager

Waar het uiteindelijk allemaal om draaide was de Ipulse Hardware Manager. Daarbij mag ik concluderen dat de opdracht geslaagd is. Het systeem is bij de opdrachtgever geïmplementeerd en in gebruik genomen.

Wanneer er gekeken wordt naar de systeemeisen kan ik zeggen dat de 100% van de basiseisen gehaald is. De comforteisen zijn grotendeels gehaald. De meeste luxe-eisen zijn wegens tijdgebrek helaas niet gehaald. Wanneer deze in de toekomst

uitgewerkt worden kan een ontwikkelaar de pilotontwerprapportages raadplegen. Met deze rapportages kan de ontwikkelaar achterhalen welke eisen wel en welke niet zijn uitgewerkt.

Een waarschuwing die ik de opdrachtgever gegeven heb is dat het pakket in de toekomst door het voller worden van de database trager kan worden. De database kan namelijk vanuit de tool niet geleegd worden. De opdrachtgever wenste aan het begin van het ontwikkeltraject dat gegevens niet permanent verwijderd kunnen worden, zodat historiegegevens niet verloren gaan. Ik heb met de wens van de opdrachtgever ingestemd, maar heb hem geadviseerd in de toekomst een tweede database te maken, waarin gegevens kunnen worden gearchiveerd.

Een tweede aanbeveling die ik de opdrachtgever heb gedaan is een beveiligingsmodule in te bouwen. Binnen het systeem is bedrijfsgevoelige informatie opgenomen. Door het bouwen van bijvoorbeeld een login scherm zouden de gegevens binnen de tool voor ongewenste gasten al een stuk moeilijker te bereiken zijn.

Ondanks bovenstaande minpuntjes ben ik uitermate tevreden met het opgeleverde systeem. Het is een eerste versie, waarop verder gebouwd kan worden. Ook de opdrachtgever is er tevreden mee en heeft mij zelfs een baan aangeboden voor na het afstuderen.

6.2 Procesevaluatie

Het eindproduct is middels een proces tot stand gekomen. Hieronder beschrijf ik mijn mening over het gevolgde proces.

6.2.1 Communicatie met opdrachtgever en eindgebruikers

Tijdens het project het ik veelvuldig contact gehad met de opdrachtgever. Tijdens de eerste ontmoetingen met de opdrachtgever hebben we flink kunnen brainstormen over het systeem en kwamen we al in een vroeg stadium op een systeemschets uit. Ook tijdens de pilotontwikkelfase heeft de opdrachtgever minstens één keer per week de tijd genomen om de voortgang te bespreken. Dit heb ik als prettig ervaren.

Daarnaast heeft er tijdens de pilotontwikkelfase een ontmoeting plaatsgevonden met de overige eindgebruikers. Tijdens deze ontmoeting werd het voorlopige systeem getoond en uitleg gegeven hoe de toekomstige situatie eruit gaat zien. Ik heb deze ontmoeting als prettig ervaren. De eindgebruikers waren allen enthousiast over het concept.

6.2.2 Ontwikkelmethode en strategie

Tijdens het project heb ik met succes gebruik gemaakt van de ontwikkelmethode IAD. Deze methode was nieuw voor mij en het heeft me daarom veel zoekwerk gekost om te ontdekken welke onderdelen wel en niet belangrijk waren om uit te werken. Ik vond dit vooral aan het begin lastig om in te schatten. Ook de te gebruiken modellen per IAD onderdeel vond ik lastig om in te schatten. In sommige gevallen heb ik daarom voor een model gekozen die misschien voor een buitenstaander niet direct voor de hand ligt. Zo heb ik bijvoorbeeld bij een aantal pilots gebruik gemaakt

van proces flowcharts. De gehanteerde modellen zijn gekozen aan de hand van de behoefte die ik op dat moment had. Deze modellen hebben me zeker geholpen bij de bouw en ik ben daarom zeer tevreden over mijn keuzes.

Ook over de keuze van de ontwikkelstrategie, incrementeel ontwikkelen, ben ik erg tevreden. Door eerste de systeemeisen volledig in kaart te brengen en deze af te beelden in een systeemconcept kreeg de opdrachtgever en de ontwikkelaar al snel een beeld van het systeem.
Het steeds opnieuw ontwerpen en bouwen van een deel van het systeem is mij erg goed bevallen. Het ontwikkelen wordt hiermee een stuk leuker en naar mijn mening van hogere kwaliteit. Dit omdat een ontworpen onderdeel meteen wordt uitgewerkt. Het in één keer invoeren van het systeem was een must voor dit project. De opdrachtgever zou aan de losse bouwstenen niets hebben. Daarom was de keuze voor deze methode van invoeren (‘big bang’) snel gemaakt. Concluderend kan ik tevreden zijn over de keuze van ontwikkelstrategie.

6.2.3 Gehanteerde technieken

Tijdens het project heb ik gebruik gemaakt van de UML modelleertechniek. Hieronder wordt deze geëvalueerd.

Dit project was het eerste project, waarin ik in projectvorm gebruik maakte van UML. Dit betekende dat ik tijd heb moeten investeren om de techniek onder de knie te krijgen. Ik denk dat me dit goed is gelukt. Over het klassendiagram en de use case beschrijvingen ben ik uitermate tevreden. Over het use case diagram iets minder. Gedurende het ontwikkeltraject kwam ik erachter dat er nog een aantal overlappende functies in het systeem zijn opgenomen. Deze waren echter niet in het use case diagram opgenomen. Hierdoor heb ik meer code geschreven dan nodig was.

Gedurende het project kwam ik erachter dat UML niet in alle gevallen de meest geschikte modellen aanbood. Ik heb daarom slechts een paar modellen gebruikt. Ik denk dat ik ondanks mijn minder goede kennis aan het begin van het project de techniek goed en nuttig in mijn ontwikkeltraject heb gebruikt.

6.2.4 Planning

Om kort te zijn is het project niet geheel volgens planning gelopen. Tot de pilotontwikkelfase liep alles nog volgens de planning, maar binnen laatstgenoemde fase zijn niet alle planningen gehaald.

Tijdens de pilotontwikkeling heb ik gebruik gemaakt van time-boxing. Dit werkte tot de derde pilot naar behoren. Voor de eerste en tweede pilot heb ik de basis en comfort onderdelen uitgewerkt die ik binnen de vastgestelde timebox kon doen. Als een timebox was verstreken werd verder gegaan naar de volgende pilot. Tijdens het uitwerken van de derde pilot kwam ik erachter dat ik te weinig tijd had ingepland voor deze pilot. Hiervoor zijn de volgende redenen:

- De derde pilot omvat o.a. het muteren van stamgegevens. Deze functionaliteit had ik reeds in Grafi Magazijn geïmplementeerd. Hiermee dacht ik door het hergebruiken van code minder tijd kwijt te zijn. Grafi Magazijn is echter geen multi-user applicatie, waardoor alsnog veel code, zoals het reconcile error

mechanisme, geprogrammeerd moest worden. Hiermee had ik vooraf bij het inplannen geen rekening gehouden.

- Een andere reden is het vergroten van de opdracht. Zo is een nieuwe functionaliteit voor de integratie van artikelen en categorieën opgenomen. Zoals afgesproken heb ik deze als luxe eis opgenomen. In overleg met de opdrachtgever werd de functionaliteit zodanig belangrijk gevonden dat besloten is deze toch te bouwen. Toen dit besloten werd heb ik direct aangegeven dat de planning voor de derde pilot gevaar zou lopen. Ik heb de opdrachtgever voorgelegd de comforteisen uit de vierde pilot een lagere prioriteit te geven dan de bovenstaande functionaliteit. Dit heeft de opdrachtgever aanvaard. Hiermee is de timebox van de vierde pilot ingekort en die van de derde pilot vergroot. Toen de originele derde timebox verstreken was ben ik wel eerst overgegaan naar de vierde pilot. Dit omdat ik wel zeker wilde zijn dat de basiseisen uit de exportpilot gerealiseerd zouden worden. Toen ik de basis functionaliteiten uit de vierde pilot had uitgevoerd heb ik besloten verder te gaan met de derde pilot.

Er was na het uitlopen van de derde pilot helaas geen tijd meer voor de algehele test. Ook is er om deze reden geen volledig testplan opgesteld. Nadat ik zelf alle functionaliteiten had nagelopen is in overleg met de opdrachtgever besloten de invoeringsfase uit te voeren. De pilots apart zijn uitgebreid getest, waardoor er genoeg vertrouwen is dat de applicatie goed functioneert.

De overall planning is daarom gehaald. Het systeem is op tijd opgeleverd. De tussentijdse planningen, zoals opgenomen in de pilotontwerprapportages, zijn echter niet allemaal gehaald. Toch ben ik tevreden over de manier, waarop het uiteindelijke doel is gehaald.

6.2.5 Doelstellingen

Het doel van de opdracht was het inzicht in het artikelassortiment van de leveranciers verbeteren en het wegnemen van handmatige handelingen bij het invoeren van orders en artikelen in King. Dit doel is voor het grootste gedeelte behaald. Wegens tijdgebrek is de importfunctionaliteit voor de tweede grote leverancier Copaco helaas niet uitgewerkt, waardoor op dit moment nog geen totaaloverzicht van het artikelassortiment van de leveranciers beschikbaar is. Toch verwacht ik dat het systeem, ook in deze vorm, een verademing is bij de orderverwerking.

Verklarende woordenlijst

- Bug : Fout
- DBMS : Database Management System
- EAN code : European Article Number code
- ERD : Entity Relationship Diagram
- Feature : Functionaliteit
- GUI : Graphical User Interface
- HTML : Hypertext Markup Language
- IAD : Iterative Application Design
- RAD : Rapid Application Design
- SO : Systeemontwikkeling
- SOAP : Simple Object Acces Protocol
- SQL : Structured Query Language
- UML : Unified Modelling Language
- Visio : Microsoft Visio 2003
- XML : Extended Markup Language

Literatuurlijst

Beukers, H., *Documentation of IntegrateX – XML order module by Tech Data*, 2001 (Geen ISBN)

Beukers, H., *Datapack - Structural Description*, 2001 (Geen ISBN)

Dewson, R., *SQL Server 2000 Programming*, Wrox Press Ltd., 2001 (ISBN 18 610 0523 7)

Quadrant Software B.V., *Handleiding King*, 2003 (Geen ISBN)

Teixeira, S., Pacheco, X., *Borland Delphi 6 Developer’s Guide*, Sams, 2002 (ISBN 06 723 2115 7)

Tolido, R.J.H., *IAD Het evolutionair ontwikkelen van informatiesystemen*, Academic Service, 1996 (ISBN 90 395 0401 6)

Vandenbulcke, J.A., *Databasesystemen voor de praktijk*, Kluwer bedrijfswetenschappen, 1997 (ISBN 90 267 2317 2)

Wamer, J., Kleppe, A., *Praktisch UML 2^{e} editie,* Addison Wesley, 2001 (ISBN 90 430 0494 4)