Afstudeerverslag
Service Repair Management System

VCS Network Division
4 januari 2004

VCS Network Division
Dalweg 201 3762 AW Soest The Netherlands
Tel: 035-60 30 309 Fax: 035-60 14 325 internet: www.vcs-network.nl

Referaat

D. Velthuizen, Soest, 04 januari 2004.

Afstudeerverslag: *Service Repair Management System*. 44 bladzijden; 14 figuren;

Dit verslag is geschreven in het kader van de afstudeerstage van Denny Velthuizen bij VCS Network Division.

Descriptoren: Extreme Programming, Systeemontwikkeling, Unified Modeling Language, Delphi, Testcomplete, DUnit, Agile, SQL.

Voorwoord

Vrijwel ieder bedrijf maakt gebruik van een soort gegevensopslag. Het uitgangspunt van het gebruiken van gegevensopslag is vaak het verbeteren van het bedrijfsproces. Het structureren en automatiseren van alle belangrijke gegevens die tijdens het bedrijfsproces van VCS Network Division worden gebruikt is het uitgangspunt van dit project.

Er is met veel plezier en inzet aan dit project gewerkt. Een woord van dank gaat uit naar de heer Verhoeff en de heer Beens die tijdens het project hun medewerking en begeleiding hebben gegeven.

Mijn bedrijfsmentor, dhr. Verhoeff, wil ik in het bijzonder bedanken voor zijn inzet en de professionaliteit waarmee hij mij begeleid heeft.

Dhr. van Peski en mevrouw Sewdajal bedank ik voor de ondersteuning en het bewaken van het goede verloop van mijn afstudeerproject.

Denny Velthuizen

Soest, januari 2004

Inhoudsopgave

Figuren

1 Inleiding

Voor u ligt het afstudeerverslag van Denny Velthuizen. In dit afstudeerverslag bespreekt de afstudeerder de procesgang van zijn afstudeerproject. Dit afstudeerproject staat in het kader van het ontwikkelen van een geïntegreerde gegevensbeheeromgeving voor de dagelijkse werkzaamheden bij het bedrijf VCS Network Division.

In dit verslag moet duidelijk worden hoe de afstudeerder te werk gegaan is, welke problemen hij tegengekomen is, welke keuzes hij gemaakt heeft en waarom hij deze keuzes gemaakt heeft.

Het afstudeerproject vindt plaats in het kader van de opleidingsvariant Ontwikkeling van Software en Technische Infrastructuren (OSTI), derhalve moet het project bepaalde raakvlakken hebben met deze opleiding. Voor het lezen van dit verslag en de externe bijlagen kan enige kennis van systeemontwikkeling en Unified Modeling Language (UML) een vereiste zijn.

In het volgende hoofdstuk vindt u een beschrijving van de organisatie waar het afstudeerproject plaats gevonden heeft. In hoofdstuk 3 wordt het projectplan samen met de opdrachtomschrijving besproken. Het projectplan is beter bekend als plan van aanpak. Hoofdstuk 4 beschrijft de uitwerking van de planninggame. Hoofdstuk 5 beschrijft het opstellen van de definitiestudie. Hoofdstuk 6, 7, 8 en 9 beschrijven het verloop van de releases. In hoofdstuk 11 wordt een uitgebreide evaluatie van het gehele afstudeerproject gegeven. Aanvullend bevat dit verslag een literatuurlijst en een lijst met definities en begrippen die voorkomen in dit verslag.

2 Beschrijving Organisatie

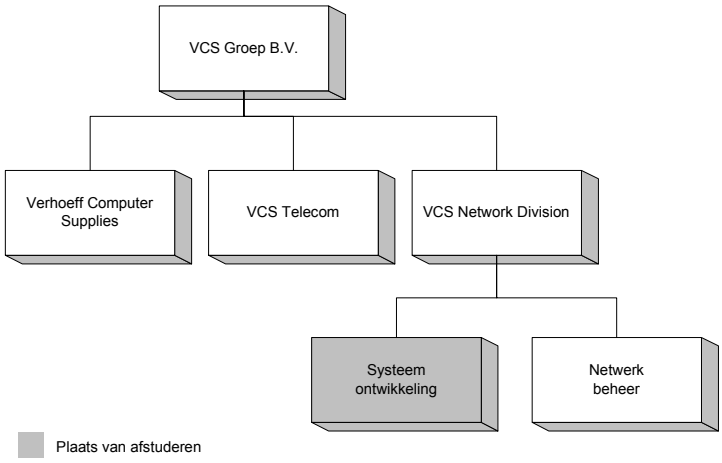
2.1 Organisatie VCS Network Division

Dit hoofdstuk gaat in op het bedrijf VCS Network Division waar de stage heeft plaats gevonden. In dit hoofdstuk wordt gesproken over de historie, werkzaamheden en organisatorische opbouw van het bedrijf.

VCS Network Division is opgericht in 1992 als onderdeel van VCS Groep B.V.. VCS Network Division houdt zich voornamelijk bezig met advies, aanleg en onderhoud van complete bedrijfsnetwerken bij bedrijven door heel Nederland en Duitsland.

Naast VCS Network Division kent VCS Groep B.V. ook de divisies VCS Telecom en Verhoeff Computer Supplies. De kernactiviteiten van VCS Telecom zijn het ontwerpen, verkopen en installeren van telefooncentrales in combinatie met CallCenter oplossingen. De kernactiviteiten van Verhoeff Computer Supplies zijn het verkopen en repareren van computerhardware. De drie divisies zijn gevestigd in één bedrijfspand in Soest.

VCS Network Division bestaat uit drie fulltime en twee parttime medewerkers en is gespecialiseerd in de netwerkomgevingen van Microsoft, Novell, AS400 en Unix. Ondanks dat VCS Groep B.V. een klein bedrijf is, heeft het een aantal grote projecten gekend waarbij het ging om 1000-user netwerken. Naast de genoemde activiteiten van VCS Network Division zijn er ook projecten die gericht zijn op het ontwikkelen van software voor externe klanten. VCS Network Division is een gecertificeerd verkoopadres van Exact software. Met regelmaat betreffen de ontwikkelprojecten aanvullingen op Exact of andere boekhoudprogramma's.



Figuur 1: schematische weergave van VCS Groep B.V.

De basis van VCS Groep B.V is gelegd in 1976. Het is begonnen met het éénmansbedrijf Verhoeff Software Holland. Na het oprichten van Verhoeff Computer Supplies (1989), VCS Network Division (1992) en VCS Telecom (1999) zijn deze divisies ondergebracht in de besloten vennootschap VCS Groep B.V.. De activiteiten van Verhoeff Software Holland zijn overgenomen door VCS Network Division.

2.2 Plaats van de afstudeerstudent

Het project is van het begin tot het eind uitgevoerd in het bedrijfspand van VCS Network Division te Soest. In dit pand is een uitgebreid scala van faciliteiten aanwezig om veel soorten softwareontwikkeling uit te voeren. Te denken valt aan verschillende servers en randapparatuur. Daarnaast kent het bedrijf een rustige en vriendelijke werksfeer, waardoor het bedrijf bekend staat als goed ICT-leerbedrijf voor verschillende opleidingsniveaus.

3 Opdrachtomschrijving en projectplan

3.1 Uitgangssituatie

Het primaire bedrijfsproces omvat het installeren en onderhouden van computernetwerken en haar randapparatuur. Tijdens het installeren en leveren van een netwerk wordt veel informatie opgedaan betreffende de configuratie van de hardware en software binnen het netwerk en de organisatie van het bedrijf. Deze gegevens zijn erg belangrijk als er onderhoud of beheer moet plaatsvinden aan het betreffende netwerk.

De gegevensgebieden binnen VCS Netwerk Division kunnen worden verdeeld in de volgende groepen:

- Relatiegegevens
- Gegevens betreffend installatie/configuratie van de hard- en software
- In- en verkoopgegevens
- Opdrachtgegevens (storingen)
- Interne en externe communicatie

Voor het beheren en opslaan van deze gegevensgebieden worden verschillende methoden gebruikt. Deze methoden omvatten o.a. Exact, Microsoft Word, Microsoft Access en Microsoft Excel.

3.2 Aanleiding

De aanleiding voor dit project is een verzoek van de opdrachtgever, dhr. H. Verhoeff, om het beheer van de genoemde gegevensgebieden te automatiseren.

Het verzamelen en beheren van de gegevens die tijdens het uitvoeren van het gehele bedrijfsproces worden opgedaan is een onderdeel van de werkzaamheden van het primaire bedrijfsproces. Door het ontbreken van een gestructureerde gegevensopslag ontstaan er problemen. Deze problemen zijn o.a.:

- Het niet kunnen vinden van belangrijke gegevens
- Werken met incorrecte gegevens
- De snelheid waarmee gegevens worden gevonden

Deze problemen resulteren in een moeizamer verloop van het gehele bedrijfsproces. Het vinden van de benodigde gegevens is een complex en tijdrovend proces en vaak noodzakelijk voor het leveren van een goede service aan de klant. Het leveren van slechte of incomplete service aan de klant veroorzaakt al gauw een negatief beeld van de kwaliteit en dit moet daarom zoveel mogelijk worden vermeden.

3.3 Doelstelling opdracht

De doelstelling van deze opdracht is het bevorderen van de efficiëntie bij het beheren van belangrijke gegevens van het gehele bedrijfsproces binnen VCS Network Division. Deze efficiëntie moet worden behaald aan de hand van de begrippen *tijd* en *gebruiksgemak*. De betreffende gegevensgebieden moeten kunnen worden beheerd met behulp van één grafische interface. Het is de bedoeling dat de geïntegreerde omgeving de bestaande opslagmethoden gaat vervangen.

3.4 Opdrachtformulering

Er moet een applicatie worden ontwikkeld waarmee de gegevensgebieden: relatiegegevens, installatie- en configuratiegegevens, storinggegevens en interne softwaregegevens, kunnen worden beheerd.

3.5 Gevolgen voor VCS Network Division

Door het invoeren van het Service Repair Management System zullen de volgende gevolgen tot stand komen.

- **Organisatie**
  - Alle werknemers van VCS Network Division zullen gebruik gaan maken van de applicatie. Alle activiteiten en betrokken gegevensgebieden zullen worden beheerd met behulp van de geïntegreerde omgeving.
  - Door het ontwikkelen van een gehele nieuwe omgeving, zullen de eindgebruikers bekend moeten raken met de nieuwe gegevensbeheeromgeving .
  - Het primaire bedrijfsproces zal sneller en beter plaats kunnen vinden.
- **Gebruikers**
  - Door het gebruik van één grafische interface en de daarbij komende standaarden zal het gebruiksgemak toenemen ten opzichte van de huidige gegevensbeheermethoden.
  - De gebruikers kunnen snel en gemakkelijk de gewenste gegevens opvragen en gebruiken.

3.6 Methode en Technieken

Inleiding

Het gehele project wordt doorlopen met behulp van Extreme Programming. Extreme Programming is een vrij nieuwe ontwikkelmethode die door veel ontwikkelaars gezien wordt als een “nieuw” tijdperk. Omdat deze methode een belangrijk onderdeel is van het project en sterk afwijkt van de door school gedoceerde methoden, zal ik hieronder de belangrijkste punten van Extreme Programming kort toelichten.

Extreme Programming (XP)

XP bestaat uit een aantal practices. Een practice is een richtlijn of activiteit die ervoor moet zorgen dat een XP-traject met succes kan worden afgerond. Hieronder worden de belangrijkste practices kort toegelicht.

De eerste practice van Extreme Programming is de planninggame. Dit is een overleg tussen opdrachtgevers en ontwikkelaars. In dit overleg worden o.a. de functionaliteiten, randvoorwaarden en de planning vastgelegd. Meer informatie over de planninggame van dit project is te vinden in de paragraaf *Planninggame*.

In tegenstelling tot de watervalmethoden maakt Extreme Programming geen gebruik van gedocumenteerde ontwerpen. In plaats van gedocumenteerde ontwerpen maakt Extreme Programming gebruik van tests, mondelinge communicatie en documentatie in de sourcecodes. Na het opleveren van een programma of release, moeten de tests bewijzen dat het programma naar wens werkt en dat deze geen fouten bevat. Als na oplevering nog wijzigingen of aanvullingen gemaakt moeten worden, kunnen de bestaande tests aantonen of alle huidige functionaliteiten nog intact zijn. Bij ontwikkelmethoden waarbij wel gebruik gemaakt wordt van gedocumenteerde ontwerpen komt het regelmatig voor dat het werkelijke product afwijkt van het ontwerp, doordat er achteraf nog wijzigingen of aanvullingen zijn gemaakt. Het bijwerken en wijzigen van gedocumenteerde ontwerpen is vaak een tijdrovend en onprettig werk, waardoor de kwaliteit vaak te wensen over laat. Het ontwerpen met behulp van tests heet Test Driven Programming (Agile).

Een andere practice van Extreme Programming is *pair programming*. De opzet van *pair programming* is het werken in tweetallen achter één terminal/pc. Deze pairs bestaan meestal uit twee programmeurs, maar ook de opdrachtgever zit regelmatig bij een programmeur om zo een duidelijk beeld te krijgen van het product en haar voortgang. De samenstelling van deze pairs wisselen dagelijks, zodat elk projectlid bekend raakt met alle onderdelen van het systeem.

Refactoring is een practice die zich richt op de kwaliteit van de sourcecodes. Er zijn heel veel regels die gebruikt kunnen worden bij het maken van de sourcecodes die er voor moeten zorgen dat deze beter leesbaar en onderhoudbaar is. Te denken valt o.a. aan korte routines, naamgeving van variabelen, schoonhouden van de code en uitlijning.

Ondanks dat Extreme Programming het gebruik van documentatie niet voorschrijft, is er toch besloten om een plan van aanpak en definitiestudie te maken. Omdat het project overdraagbaar moet zijn naar o.a. school, is er besloten om deze documenten op te nemen als bijlage van dit verslag. Het plan van aanpak en de definitiestudie kunnen worden gezien als een uitwerking van de planninggame.

Om de lezer van dit eindverslag een beter beeld te geven van de technische aspecten van het project, zijn er op een aantal plaatsen diagrammen geplaatst volgens de standaarden van Unified Modeling Language (UML). ULM is gebruikt, vanwege het feit dat het project objectgeoriënteerd is gerealiseerd. De schema's die zijn gebruikt betreffen voornamelijk usecase beschrijvingen en diagrammen en klasse diagrammen. Deze diagrammen dienen dus alleen ter ondersteuning en verduidelijking en niet als uitgangspunt. Voor een visuele weergave van het databasemodel is gebruik gemaakt van ERD.

Waarom XP?
XP biedt in mijn ogen tal van voordelen ten opzichte van andere ontwikkelmethoden. XP is een praktisch ingestelde methode die na het doorlopen van de eerste release een werkend stuk applicatie oplevert. Omdat de methode geen documentatie in de vorm van verslagen voorschrijft, kan er veel tijd gestoken worden in de ontwikkeling zelf. Ikzelf ben ook erg praktisch ingesteld en zie graag snel resultaat van mijn werk. Hierdoor kan ik me helemaal vinden in de levensloop van het XP-traject.

De opdrachtgever werkt actief mee aan de ontwikkeling, waardoor het bijna onmogelijk is een applicatie te ontwikkelen die niet voldoet aan de gestelde eisen. Daarnaast zal het op te leveren product makkelijk te onderhouden zijn, omdat de sourcecodes volgens de regelgeving van *refactoring* worden opgeleverd. Bovendien zijn er aan het eind van het project tests die alle bestaande functionaliteiten kunnen controleren wanneer de sourcecodes worden aangepast en/of uitgebreid.

3.7 Projectorganisatie

Tijdens het doorlopen van het project kwam naar voren dat er binnenkort een overname van VCS Network Division zal plaatsvinden door enkele huidige medewerkers. Door deze bekendmaking heb ik als projectleider besloten om de toekomstige eigenaren te betrekken bij het project. Deze beslissing is genomen, omdat de medewerkers de hoofdgebruikers gaan worden van de applicatie. Het betreft hier dhr. Beens en dhr. van Vulpen.

Naam	Taak
Denny Velthuizen	Ontwerper/programmeur
dhr. H. Verhoeff	Opdrachtgever/begeleider
dhr. R. Beens	Opdrachtgever/begeleider
dhr. J. van Vulpen	Opdrachtgever
dhr. Peski	Vakdocent
mevr. Sejadwal	Vakdocent

3.8 Gebruikte faciliteiten

Tijdens de realisatie van het project is gebruikt gemaakt van Borland Delphi 7. Dit is een grafische ontwikkelomgeving waarmee het mogelijk is om Windows-applicaties te maken. Deze omgeving is gekozen, omdat ik zelf veel ervaring heb met Delphi en graag met deze ontwikkelomgeving werk. Daarnaast zijn de huidige en toekomstige medewerkers van VCS Network Division in staat de codes gemaakt met Delphi te lezen en te begrijpen.

Voor het beheren en bewaken van de voortgang van het project heb ik gebruik gemaakt van een zelfgemaakte support-template. Dit is een Microsoft Excel-bestand waarmee de planning en voortgang van het project kan worden bijgehouden. Elk uit te voeren activiteit in de support-template kan worden voorzien van een status. In de support-template worden ook alle aanvullende eisen en bugs van dit project toegevoegd. In de paragraaf *Planninggame* is een uitwerking te vinden van de support-template van dit project. Voor het maken van de verslaggeving is gebruik gemaakt van Microsoft Office 2000 en Microsoft Visio 2000 en Dezign for Databases.

Voor het maken van de tests is gebruik gemaakt van twee testomgevingen: Testcomplete en DUnit. Meer informatie daarover is te vinden in de paragraaf *Testomgeving*.

Microsoft SQL Server is gebruikt als database management system. Microsoft SQL Server wordt al gebruikt door VCS Network Division voor andere projecten en mij werd verzocht deze ook te gebruiken in verband met onderhoud. Door het gebruik van Microsoft SQL Server was het noodzakelijk om een third party component voor Delphi aan te schaffen voor de communicatie met Microsoft SQL Server. Dit component heet SDAC en een licentie kost $90 dollar (US). Het voordeel van dit component is dat er geen extra omgeving op de cliënt pc geïnstalleerd hoeft te worden om met Delphi te kunnen communiceren met Microsoft SQL Server, wat met standaard Delphi-componenten vaak wel het geval is.

Resources
Omdat ik zelf al redelijk vaak gebruik gemaakt heb van Delphi, beschik ik over een groot aantal units waarin verschillende bruikbare routines zitten. Deze sourcecodes kunnen dus van pas komen bij de ontwikkeling van dit project.

3.9 Risico's en maatregelen

Nr	Risico	Kans	Impact	Preventieve maatregelen
1	Het niet tijdig realiseren van het project	Middel	Middel	• Elke 2 weken planning aanpassen • Voortgangsoverleg • Aanpassen van systeemeisen • Project kan intern worden overgenomen
2	Het eindproduct voldoet niet aan de wensen van de opdrachtgever	Laag	Hoog	• Middels pair programming en voortgangsoverleg krijgt de opdrachtgever een duidelijk beeld van de gehele applicatie.
3	Het eindproduct moet gewijzigd worden na afloop van het project	Middel	Laag	• Kennis van applicatie aanwezig bij werknemers (pair programming) • Code makkelijk onderhoudbaar (refactoring)

Het project wordt gerealiseerd door een kleine projectgroep, waardoor het risico aanwezig is dat het systeem niet tijdig wordt gerealiseerd. Omdat de planning om de twee weken wordt bijgesteld kan er vroegtijdig gezien worden of de gestelde eisen tijdig kunnen worden gerealiseerd. Mocht dit niet het geval zijn, kan er besloten worden een aantal systeemeisen aan te passen of te laten vervallen. Mochten de vervallen systeemeisen onmisbaar zijn, kan er besloten worden om na het project intern het programma verder te ontwikkelen. Mede omdat het project goed overdraagbaar is, is het probleem als *middel* geclassificeerd.

De kans dat het opgeleverde resultaat niet voldoet aan de eisen van de opdrachtgever is als *laag* geclassificeerd. De classificatie *laag* is gekozen, omdat de opdrachtgever dagelijks op de hoogte wordt gebracht van het product en haar voortgang. De gewenste wijzigingen zullen direct na overleg worden doorgevoerd in de planning en in de applicatie. Samen met de opdrachtgever is bepaald dat de impact *hoog* is, wanneer het product niet voldoet aan de vooraf gestelde eisen.

3.10 Ontwikkelscenario

Bij Extreme Programming is het verplicht eerst de test te maken en vervolgens de betreffende routine of routines. Deze regel is van kracht om er voor te zorgen dat er altijd een test wordt gemaakt voor de betreffende routines of functionaliteit. Het is dus een disciplinaire maatregel. Een programmadeel mag ook alleen door de opdrachtgever goedgekeurd worden als alle tests goed worden doorlopen.

3.11 Testomgeving

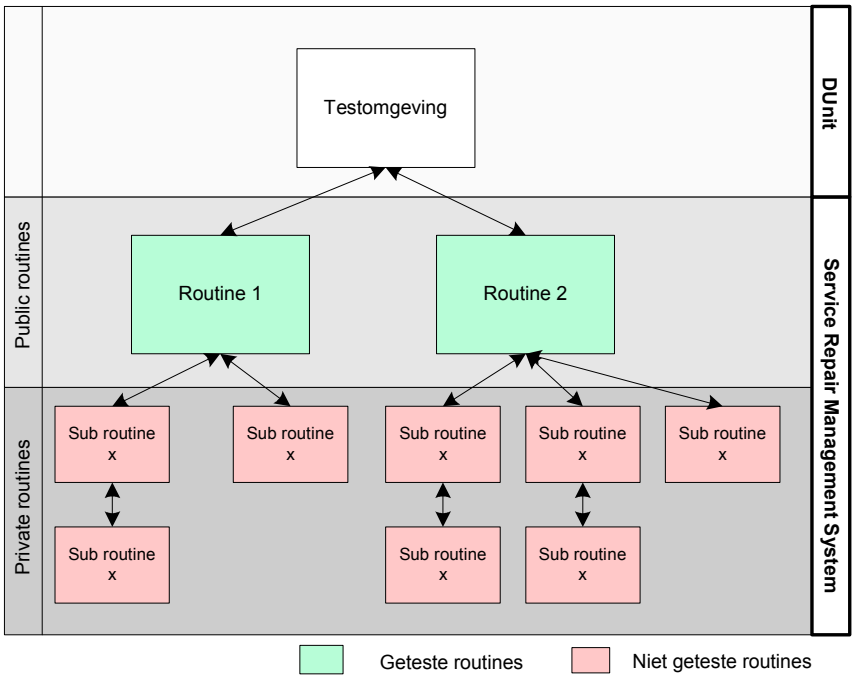
Zoals al eerder aangegeven is XP een testdriven ontwikkelmethode waarbij de tests een belangrijk onderdeel van het project vormen. Bij het realiseren van het project is er gebruik gemaakt van twee testomgevingen, DUnit en Testcomplete (voorheen AQTest), waarmee het mogelijk is om Delphi-applicaties te testen. Testcomplete en DUnit zijn twee testomgevingen die op verschillende niveaus tests kunnen uitvoeren. Met deze twee testomgevingen is het mogelijk om alle aspecten van een applicatie te testen.

Voor het testen van de user interface is gebruik gemaakt van Testcomplete. Met Delphi Script, broncodes die lijken op Object Pascal (Delphi), kunnen er met Testcomplete tests gemaakt worden die de handelingen van de gebruiker emuleren. Deze handelingen zijn alle denkbare acties die op een user interface kunnen plaats vinden, bijvoorbeeld het indrukken van knoppen en het invullen van tekstvelden. Omdat het hier gaat om een geautomatiseerde testomgeving kan een machine een grote hoeveelheid tests achter elkaar uitvoeren zonder tussenkomst van een persoon of operator. Tijdens het testen van de user interface was er sprake van whitebox testing, wat inhoudt dat we alle public gegevens van de applicatie op de achtergrond mochten en konden benaderen tijdens het maken en uitvoeren van de tests. Dus ook de niet zichtbare variabelen konden waar nodig worden getest.

Voor het testen van de user interface onafhankelijke routines van de applicatie is gebruik gemaakt van DUnit. Dit is een testomgeving speciaal voor Delphi routines. Met DUnit is het mogelijk om het gedrag van de routine te testen met verschillende invoer. Een veel voorkomend aspect bij het testen van een routine is het controleren van de uitvoer. Met DUnit kan niet getest worden aan de hand van een user interface, waardoor het noodzakelijk was alle routines systeem(user interface) onafhankelijk te maken. Het systeemonafhankelijk maken van de routines heeft als resultaat gehad dat de meeste routines te hergebruiken zijn (een onderdeel van refactoring), tijdens het project zelf, maar ook in de toekomst bij andere projecten. Het gebruik van unit-tests is een belangrijk uitgangspunt van Extreme Programming. Tijdens het testen van de routines was er sprake van blackbox testing, wat inhoudt dat je tijdens het testen alleen zicht hebt op de invoer en uitvoer van de routines en niet op wat er op de achtergrond gebeurt. Vooraf moet worden vastgelegd op welk niveau er getest gaat worden. Dit om de kwaliteit/kwantiteit verhouding van de tests goed te houden. Als routines

gebruikmaken van subroutines kan er voor gekozen worden om elke subroutine apart te testen. Tijdens dit project zijn er alleen unit-tests gemaakt voor de public (interface) routines. Dit zijn dus alleen de routines die direct van buiten de betreffende unit beschikbaar zijn. In figuur 2 is een schematische weergave gegeven van het gebruikte testniveau bij de unit-tests.

Tijdens het testen van de user interface worden vaak ook user interfaceonafhankelijke routines aangeroepen waardoor er soms overlapping ontstaat. Deze overlapping is vaak onvermijdelijk en vaak ook niet erg. Omdat de verdeling gemaakt wordt tussen de user interfacehandelingen en achtergrondroutines zijn de fouten veel beter te lokaliseren. Ook is het gebruikelijk om de user interface pas te testen als alle unit-tests werken.



Figuur 2: testniveau voor de unit-tests.

3.12 Releaseplan

Tijdens de planninggame wordt er een definitief releaseplan opgesteld. In het releaseplan zijn alle af te bakenen delen van het uiteindelijke programma opgesteld. In het hoofdstuk *Planninggame* worden per release de bijbehorende iteraties aangegeven. De volgorde van de releases die zijn opgesteld geeft tevens de prioriteitvolgorde aan. De releaseplanning die voor aanvang van de planninggame is opgesteld, in verband met het opstellen van de opdrachtomschrijving, ziet er als volgt uit:

- **Relatiebeheer**

Deze release omvat het beheren van de relatiegegevens omvattend de standaard naam, adres, woonplaatsgegevens en de installatie/configuratie van de hard- en softwaregegevens van de betreffende relatie.

- **Storingbeheer**

Deze release omvat het beheren van alle storingen die binnenkomen bij VCS Network Division. Dit omvat tevens het beheren van de interne werknemersgegevens en het koppelen van de storingen aan de interne werknemers.

- **Intern softwarebeheer**

Deze release omvat het beheren van de interne software bank.

- **Telefoonafhandeling**

Deze release omvat het beheren van alle in- en uitgaande telefoongesprekken van VCS Network Division. Naast het beheren van de telefoongesprekken is het beheren van de leveranciers een onderdeel van deze release.

3.13 Op te leveren producten

Aan het eind van het project moeten de volgende producten opgeleverd zijn:

1: Plan van aanpak
2: Definitiestudie
3: Release 1 - Relatiebeheer (sourcecodes)
4: Release 2 - Storingbeheer (sourcecodes)
5: Release 3 - Intern software beheer (sourcecodes)
6: Release 4 - Telefoonafhandeling (sourcecodes)
7: Interface en unit tests die bewijzen dat de releases werken

4 Planninggame

4.1 Inleiding

Een belangrijk onderdeel van Extreme Programming is de planninggame. In deze planninggame worden alle belangrijke aspecten van het project besproken, zoals de gewenste functionaliteiten van het programma en de planning. De planninggame vindt plaats voor aanvang van het project. In het geval van het project Service Repair Management System vond de planninggame plaats op de eerste dag van de stage. Het overleg vindt vaak plaats met behulp van een whiteboard. Hierdoor is het mogelijk dat alle ideeën worden uitgetekend, waardoor een interactief overleg plaatsvindt.

Tijdens de planninggame waren alle betrokken medewerkers en opdrachtgevers aanwezig. Het overleg heeft de volgende resultaten opgeleverd:

- Globaal databasemodel
- Schetsen van de user interfaces
- Beschrijvingen van de gewenste functionaliteiten
- Planning

Omdat VCS Network Division niet eerder gewerkt heeft volgens de richtlijnen van XP heb ik de belangrijke punten van deze methode tijdens de planninggame toegelicht.

4.2 Planning

De planning bestaat uit verschillende releases met ieder een aantal iteraties. Per iteratie kunnen er verschillende stories worden toegekend. De opbouw van de planning binnen XP is te zien in figuur 3.

Tijdens de planninggame is ook de release- en iteratieverdeling gemaakt aan de hand van de gewenste eisen van de opdrachtgever. XP schrijft voor om maximaal 4 iteraties per release te plannen. In iedere iteratie mogen drie punten (tijdseenheden) worden toegekend aan één of meerdere stories. Deze tijdseenheid staat gelijk aan één ideale werkdag. Dit houdt in dat er rekening gehouden wordt met onvoorspelbare tegenslagen zoals ziekte en storing.

Release 1		
	Iteratie 1	
		Story 1
		Story 2
	Iteratie 2	
		Story 1
		Story 2
	Iteratie 3	
		Story 1
		Story 2
Release 2		
	Iteratie 1	
		Story 1
		Story 2
	enz...	

Figuur 3: opbouw planning

Omdat de planning in alle gevallen een schatting is, moet er na elke iteratie bekeken worden of de planning aangepast moet worden. Als het verloop sneller gaat dan verwacht, mag de opdrachtgever meer stories plannen in de volgende iteratie. Als er vertraging is mag de opdrachtgever minder stories toewijzen aan alle iteraties. Ook mag de opdrachtgever activiteiten verplaatsen en toevoegen aan de planning. Het resultaat van de planninggame verwerkt in de support-template is te zien in figuur 4.

Om de planning te maken moet er besloten worden hoeveel dagen er nodig zijn voor één ideale werkdag. Als er geen idee is hoeveel dagen er nodig zijn voor één ideale werkdag, schrijft XP standaard 3 normale werkdagen voor. Uit mijn ervaring met deze tijdseenheden heb ik besloten om 2 normale werkdagen te gebruiken voor één ideale werkdag. Dit is na te rekenen door het totaal aantal beschikbare dagen te delen door het aantal

toegewezen punten. Het totaal aantal dagen voor het gehele project is **75** dagen waarvan **2** dagen zijn gebruikt voor het opstellen van het plan van aanpak en het uitvoeren van de planninggame. Dus, het resterend aantal dagen voor het project is **73**, het aantal toegewezen punten is **36**. Eén punt is dus: **73/36 = 2,03** dagen.

Activiteit	Punten	Status
Release 1 (Relatiebeheer)		**O**
Iteratie 1		**O**
Installeren werkomgeving	**1**	**O**
Opstellen definitiestudie	**2**	**O**
Iteratie 2		**O**
Applicatie interface design + interface design relatiebeheer	**1**	**O**
Implementeren toevoegen/wijzigen (enz.) van relaties	**2**	**O**
Iteratie 3		**O**
Genereren overzichten relatiegegevens	**1**	**O**
Interface design configuratie/systeembeheer	**1**	**O**
Implementeren beheer gebruikers bij relatie	**0.5**	**O**
Implementeren beheer software bij relatie	**0.5**	**O**
Iteratie 4		**O**
Implementeren beheer hardware en gebruikers bij relatie	**1**	**O**
Implementeren plaatjes/schema's van bedrijfsnetwerk/structuur	**1**	**O**
Versturen e-mail naar alle relaties	**1**	**O**
Release 2 (Storingbeheer)		**O**
Iteratie 1		**O**
Interface storingbeheer	**1.5**	**O**
Implementeren toevoegen/bekijken/wijzigen/verwijderen storingen	**1.5**	**O**
Iteratie 2		**O**
Implementeren statusbeheer van de storingen	**1.5**	**O**
Interface design interne medewerkersbeheer	**1.5**	**O**
Iteratie 3		**O**
Implementeren van toevoegen/wijzigen/verwijderen werknemers	**1.5**	**O**
Genereren stroring-views en overzichten	**1.5**	**O**
Release 3 (Internsoftwarebeheer)		**O**
Iteratie 1		**O**
Ontwerpen interface internsoftwarebeheer	**1.5**	**O**
Implementeren van toevoegen en zoeken software	**1.5**	**O**
Iteratie 2		**O**
Implementeren van wijzigen/verwijderen software	**1**	**O**
Conversie van huidige softwaredatabank	**2**	**O**
Release 4 (Telefoonafhandeling)		**O**
Iteratie 1		**O**
Onwerpen interface telefoonafhandeling	**1.5**	**O**
Implementeren van toevoegen/wijzigen/verwijderen afhandelingen	**1.5**	**O**
Iteratie 2		**O**
Implementeren van status beheer afhandelingen	**1.5**	**O**
Interface design leverancierbeheer	**1.5**	**O**
Iteratie 3		**O**
Implementeren van toevoegen/vewijderen/wijzigen leveranciers	**1.5**	**O**
Genereren views en overzichten van gesprekken	**1.5**	**O**
Totaal aantal punten	**36**	

Legenda status

Symbool	Betekenis
O	Open (to do)
A	Active - er mag maar 1 activiteit tegelijk actief zijn
T	Test - het onderdeel wordt gekeurd/getest door de opdrachtgever
D	Done - activiteit is klaar en gekeurd door de opdrachtgever

Figuur 4: uitwerking van planning

4.3 Beslissingen

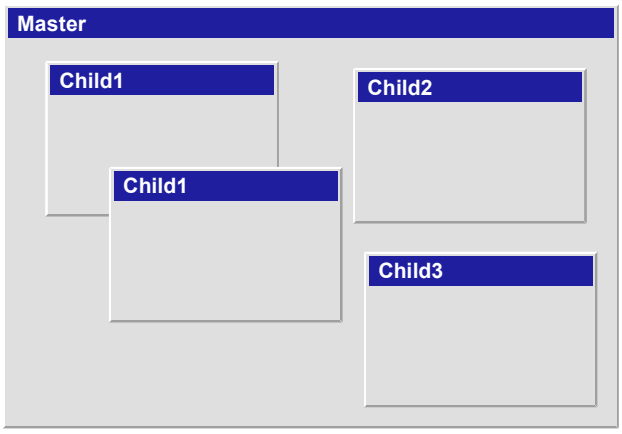
Interface

Tijdens het whiteboard-overleg werd al snel duidelijk dat het project veel verschillende schermen nodig heeft om alle gewenste functionaliteiten van de verschillende releases te kunnen implementeren. Mede vanwege dit gegeven heb ik besloten om een MDI-applicatie (Multiple Document Interface) te maken. Een MDI-applicatie is een user interface-techniek waarmee één master-scherm gemaakt wordt waarin verschillende child-schermen kunnen worden gestart. Met het oog op de komende releases is het dus de bedoeling dat er meerdere schermen tegelijk en door elkaar gebruikt kunnen worden. Een MDI-applicatie biedt tal van voordelen, maar helaas ook een aantal nadelen. De voordelen zijn de volgende:

- Meerdere *verschillende* schermen tegelijk open
- Meerdere *dezelfde* schermen tegelijk open
- Een fraaiere en duidelijkere user interface volgens de MDI-standaard

De nadelen zijn de volgende:

- Ontwikkeling van MDI-applicaties is complexer en moeilijker
- Door het gebruik van een MDI-omgeving zijn er meer testscenario's



Figuur 7: opbouw MDI-applicatie

Vooral het feit dat het aantal testscenario's vele malen groter is met een MDI-omgeving, was een belangrijk punt om rekening mee te houden tijdens het beslissen. XP schrijft namelijk voor om alle soorten scenario's af te vangen met de tests. Want wie garandeert zonder te testen of het gedrag van een scherm hetzelfde is wanneer er meerdere instanties van een scherm geopend zijn?! Door mijn ervaring met het ontwikkelen van MDI-applicaties heb ik toch besloten deze user interface-techniek te gebruiken en de tests per scherm op te stellen.

Gebruikerstoegang

Tijdens het overleg kwam duidelijk naar voren dat het nog niet interessant is voor VCS Network Division om gebruik te maken verschillende gebruikersrechten. Dit omdat het niet noodzakelijk is om enkele gegevensgebieden af te schermen voor de medewerkers van VCS Network Division. De functionaliteit kan uiteraard in een later stadium als *story* toegevoegd worden aan de planning.

5 Definitiestudie

5.1 Inleiding

De definitiestudie kan gezien worden als een uitwerking van de planninggame. Het document bevat dus de wensen en afspraken die zijn overeengekomen tijdens de planninggame.

Planning
Normaal gesproken wordt er in de definitiestudie een gedetailleerde planning weergegeven van het gehele project. Omdat XP gebruik maakt van een dynamische planning en omdat deze planning voor iedereen binnen VCS Network Division toegankelijk is, heb ik besloten deze planning niet toe te voegen aan de definitiestudie. Een gedetailleerde uitwerking van de planning is te vinden in het hoofdstuk *Planninggame*.

5.2 Systeemeisen

De systeemeisen kunnen worden opgedeeld in *functionele* systeemeisen en *niet functionele* systeemeisen. De functionele systeemeisen zijn opgesteld aan de hand van de release-verdeling die is voortgekomen uit de planninggame. De functionele systeemeisen zoals deze vooraf zijn opgesteld zijn:

Relatiebeheer

- Geavanceerd zoeken
- Beheren van relatiegegevens (toevoegen, verwijderen, wijzigen)
- Beheren van configuratie– en installatiegegevens
- Weergeven en afdrukken van zoekopdrachten

Storingbeheer

- Beheren van storingen (toevoegen, verwijderen, wijzigen)
- Geavanceerde weergave van de storingen
- Specificeren en afdrukken van storingoverzicht

Intern softwarebeheer

- Geavanceerd zoeken
- Beheren van interne software (toevoegen, verwijderen, wijzigen)

Telefoonbeheer

- Beheren van gesprekken (toevoegen, verwijderen, wijzigen)
- Geavanceerde weergave van de gesprekken
- Specificeren en afdrukken van gesprekoverzicht

Bij Extreme Programming heeft de opdrachtgever het recht om tijdens de ontwikkeling van het product nog extra functionaliteiten op te stellen en te plaatsen in elke iteratie naar wens. Ook tijdens dit project is het enkele malen voorgekomen dat de opdrachtgever, na het tussentijds zien van het product, extra functionaliteiten heeft toegevoegd aan de eisen. Bij het tussentijds toevoegen van systeemeisen heb ik zelf ook een grote rol gespeeld. Ik mocht altijd voorstellen doen om het product beter te maken. Het systeem moet aan de volgende niet functionele systeemeisen voldoen:

1. Het systeem moet kunnen draaien op elk Microsoft Windows systeem
2. Het systeem dient toegang te hebben tot een Microsoft SQL DBMS
3. Het systeem dient vanaf een netwerk met meerdere computers tegelijk toegankelijk te zijn
4. Het systeem moet worden geïmplementeerd in het bestaande bedrijfsnetwerk van VCS Network Division

5.3 Systeemconcept

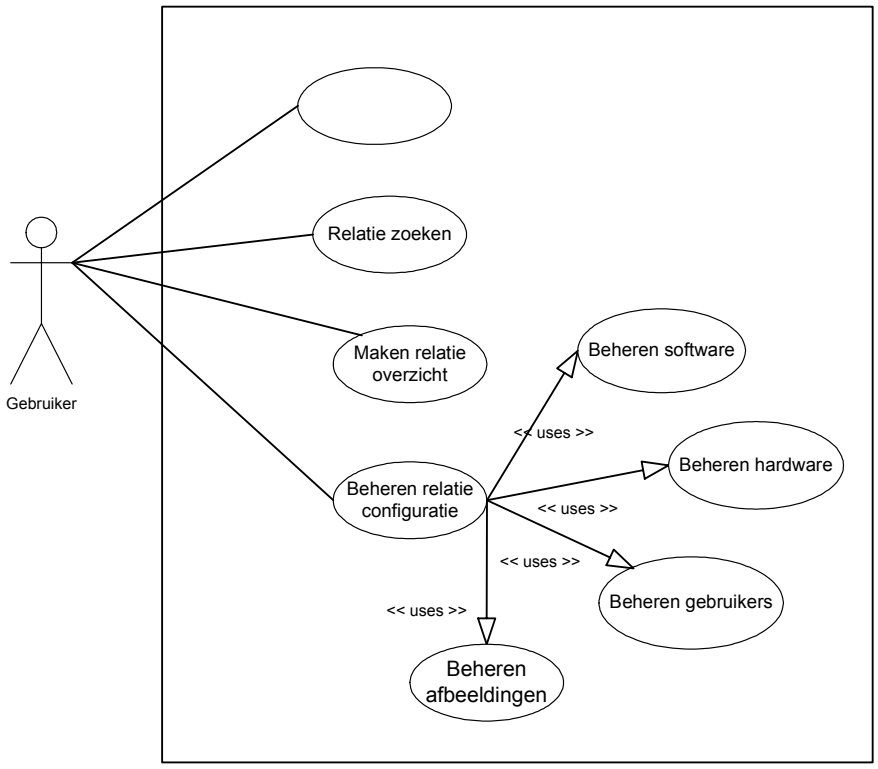
Tijdens de planninggame is samen met alle projectleden op het white-board een opzet gemaakt die duidelijk weergaf hoe het programma er globaal uit moest gaan zien. Deze opzet is gemaakt met behulp van simpele lijntjes en blokjes om het systeemconcept zo simpel en duidelijk mogelijk te maken voor alle deelnemende projectleden. Het systeemconcept van het relatiebeheer is duidelijk te vertalen in use-case-beschrijvingen en een use-case-diagram weergegeven in figuur 5.

Use-case - Relatiebeheer	
Naam	Beheren relatie
Samenvatting	De gebruiker ziet een scherm waarmee hij alle standaardhandelingen kan verrichten zoals toevoegen, verwijderen en wijzigen.
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	De gebruiker start het relatiebeheerscherm. Om een gebruiker te kunnen wijzigen of verwijderen moet eerst de betreffende relatie gezocht worden. Als er een relatie verwijderd wordt zullen ook de bijbehorende installatiegegevens verwijderd worden.
Resultaten	De gebruiker heeft de mogelijkheid om alle gewenste handelingen van de relatie uit te voeren.

Use-case - Relatiebeheer	
Naam	Zoeken relatie
Samenvatting	De gebruiker ziet een scherm waarmee relaties gezocht kunnen worden
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	De gebruiker start het relatiebeheerscherm. Hiermee kan er gezocht worden op verschillende categorieën zoals naam, contactpersoon, adres, telefoonnummer. Alle entries van de database zullen gevonden worden wanneer een deel ervan overeenkomt met de zoekstring. Met navigatieknoppen (vorige/volgende) kan er in het zoekresultaat genavigeerd worden.
Resultaten	De gewenste relatie of relatiegroep is gevonden.

Use-case - Relatiebeheer	
Naam	Maken overzicht
Samenvatting	De gebruiker kan een uitdraai (overzicht) maken van de gevonden relatie of relatiegroep.
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server, er is minimaal 1 entry in het zoekresultaat.
Beschrijving	Als de gebruiker een zoekopdracht heeft gedaan, is het mogelijk deze weer te geven en af te drukken.
Resultaten	Een lijst met daarop de gewenste gegevens van een relatie of relatiegroep.

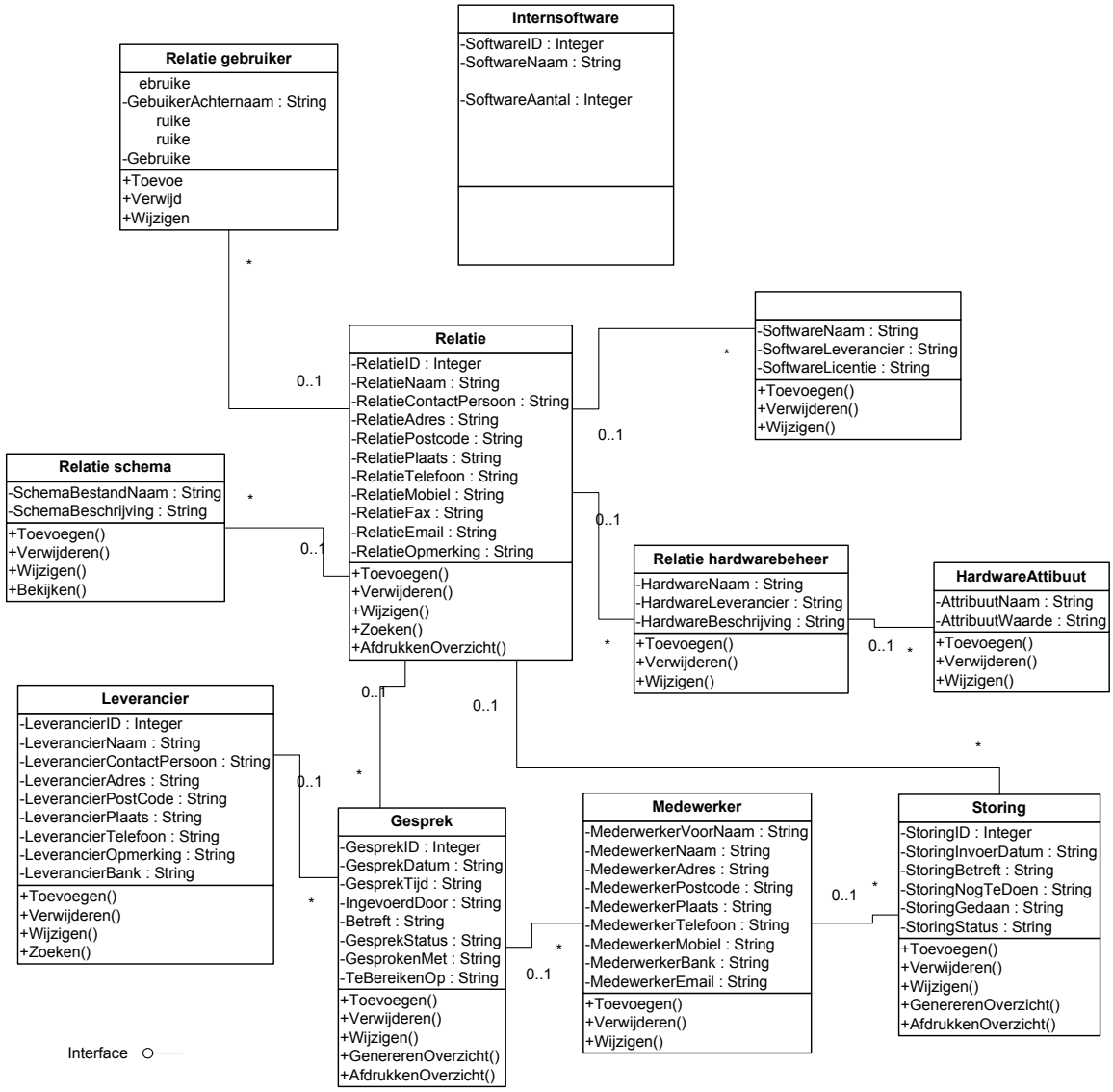
Use-case - Relatiebeheer	
Naam	Beheren installatie en configuratie
Samenvatting	De gebruiker kan per relatie alle gegevens beheren betreffende de installatie en configuratie van systeem of netwerk.
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server, er is minimaal 1 entry in het zoekresultaat.
Beschrijving	Als de gebruiker een relatie gezocht heeft, is het mogelijk om de installatie van deze relatie te beheren. De gegevensgebieden zijn: beheren software, beheren hardware, beheren gebruikers en beheren configuratieschema's. Beheren omvat bij alle gegevensgebieden weergeven, toevoegen, verwijderen en wijzigen.
Resultaten	De configuratiegegevens kunnen naar wens worden beheerd en ingezien.



Figuur 5: usecase-diagram relatiebeheer

In dit schema is ook duidelijk te zien hoe de programmadialoog ongeveer zal zijn voor het gedeelte *relatiebeheer*. Er is in dit diagram maar één *actor* weergegeven. De applicatie maakt gebruik van één niveau van gebruikerstoegang waardoor elke gebruiker toegang heeft tot elk gedeelte van de applicatie.

Van alle informatie die is opgedaan bij het bepalen van het systeemconcept kon een klassendiagram gemaakt worden voor het gehele project. Het klassendiagram zoals in figuur 6 staat beschreven is in de opbouw van het programma duidelijk terug te vinden. Opvallend is het feit dat elke individuele klasse een scherm in de uiteindelijke implementatie van SRMS representeert.



Figuur 6: klassendiagram SRMS

In de klassen zijn alleen attributen opgenomen die de interface van de klasse representeren. De ID-attributen die in enkele klassen voorkomen zijn enkel opgenomen, omdat deze voor de gebruiker interessant zijn, en daarom dus ook in de interface van de betreffende klasse aanwezig zijn. Zo is bijvoorbeeld het relatienummer (RelatieID) handig voor het maken van offertes/facturen, maar het ID van een hardwareattribuut niet.

5.4 Technische structuur en inrichting

Het opstellen van de technische structuur en inrichting was niet veel werk. Omdat het systeem gebruikt gaat worden in het bestaande netwerk van VCS Network Division was het niet nodig om zaken te bepalen zoals waar de hardware te staan zou komen. Het is de bedoeling dat de applicatie vanaf het interne intranet gaat werken. Deze keuze is gemaakt, omdat dit technisch goed mogelijk is en het onderhoud maar op één plek hoeft plaats te vinden. Met het oog op verschillende programmeertechnieken is dit een vrij belangrijke beslissing. De

applicatie moet door deze beslissing met behulp van systeemonafhankelijke technieken ontwikkeld worden. Enkele voorbeelden hiervan zijn: het gebruiken van ini–files in plaats van registerkeys en geen gebruik van locale dll's.

6 Release 1 Relatiebeheer

Inleiding

In deze release moet een interface ontwikkeld worden waarmee het mogelijk is de relaties van VCS Network Division te beheren. Naast de standaard klantgegevens moet het mogelijk zijn om de complete installatie van de betreffende klant te beheren. Te denken valt hier aan hardware- en softwareconfiguratie en aanwezige gebruikers.

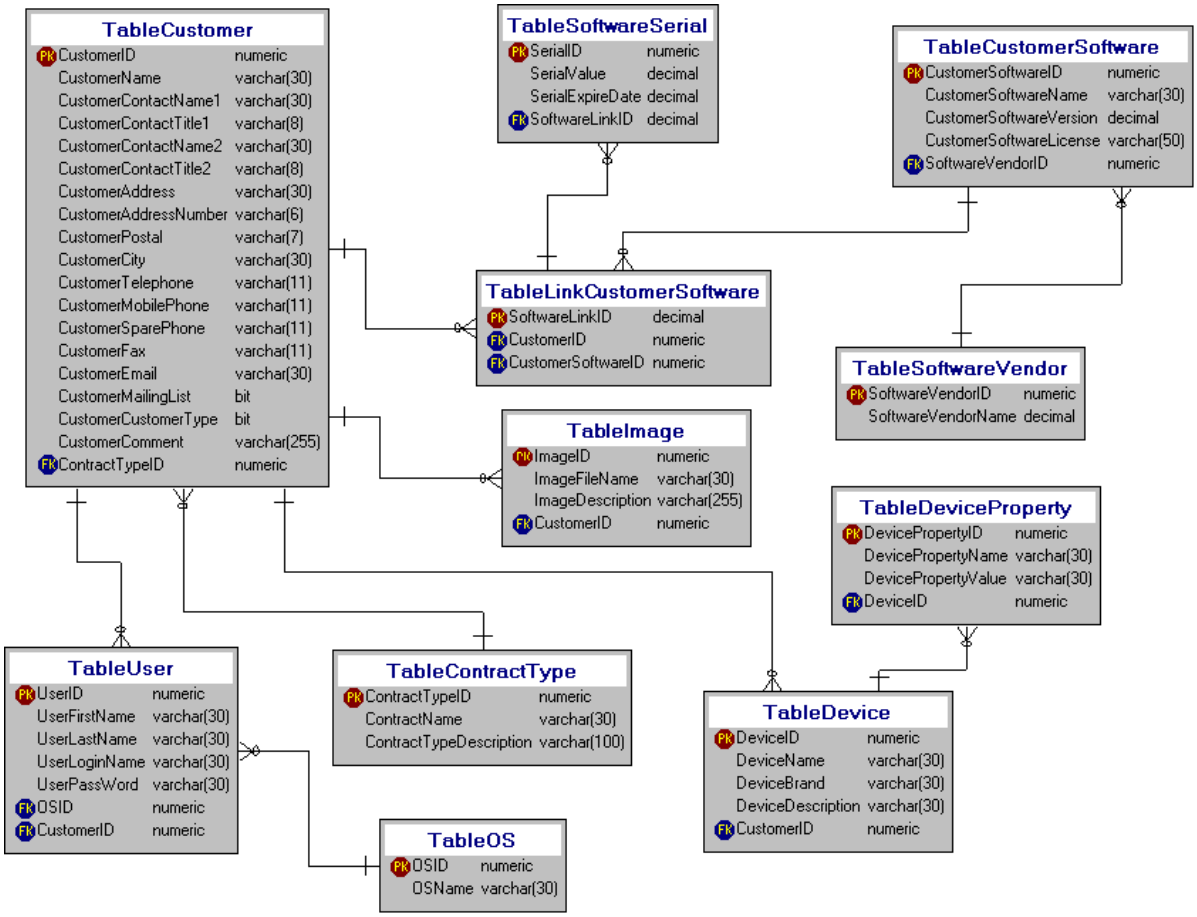
Voor er begonnen kon worden met de ontwikkeling moest de werkplek geconfigureerd worden. Borland Delphi, Microsoft SQL Server en de testomgevingen moesten op elkaar afgestemd worden. Ook is er een opzet gemaakt voor de opslag van alle projectbestanden.

Ontwerp

Databasemodel
Tijdens de planninggame is er een globale opzet gemaakt voor het gehele databasemodel. De globale databaseopzet is opgesteld om de haalbaarheid van het model vooraf te testen, zodat er tijdens de realisatie geen grote problemen plaats zouden vinden. Het globale databasemodel is tijdens de realisatie vrijwel geheel intact gebleven. Ondanks dat er een globaal databasemodel opgezet was, werden tijdens deze release alleen de benodigde tabellen van relatiebeheer uitgewerkt en gecreëerd.

Om het overzicht in het databasemodel te bewaren en de scheiding tussen de releases duidelijk weer te geven zal ik per release het definitieve databasemodel weergeven. Het databasemodel is gemaakt aan de hand van het klassendiagram. Het overall-klassendiagram was de basis voor het databaseontwerp. Iedere klasse met één of meer attributen was een potentiële tabel.

Zoals in figuur 8 is weergegeven heb ik besloten om het beheer van de relatiesoftware vrij omslachtig te implementeren in het databasemodel. Deze keuze is gemaakt om veel fysieke dataopslag in de database te voorkomen. In de praktijk komt het veel voor dat verschillende klanten dezelfde software hebben en het is niet efficiënt om de software meerdere keren in *TableCustomerSoftware* terug te laten komen. Daarnaast is er gekozen om een aparte tabel te maken voor het bijhouden van de serials (installatiesleutels) van de software. Om het model betreffende de software compleet te krijgen moest er bij de *TabelSerial* een verwijzing komen bij welke klant en software deze hoort. Deze relatie was al gelegd in de linktabel *TableLinkCustomerSoftware* waardoor een verwijzing van *TableSerial* naar *TableLinkCustomerSoftware* voldoende was.



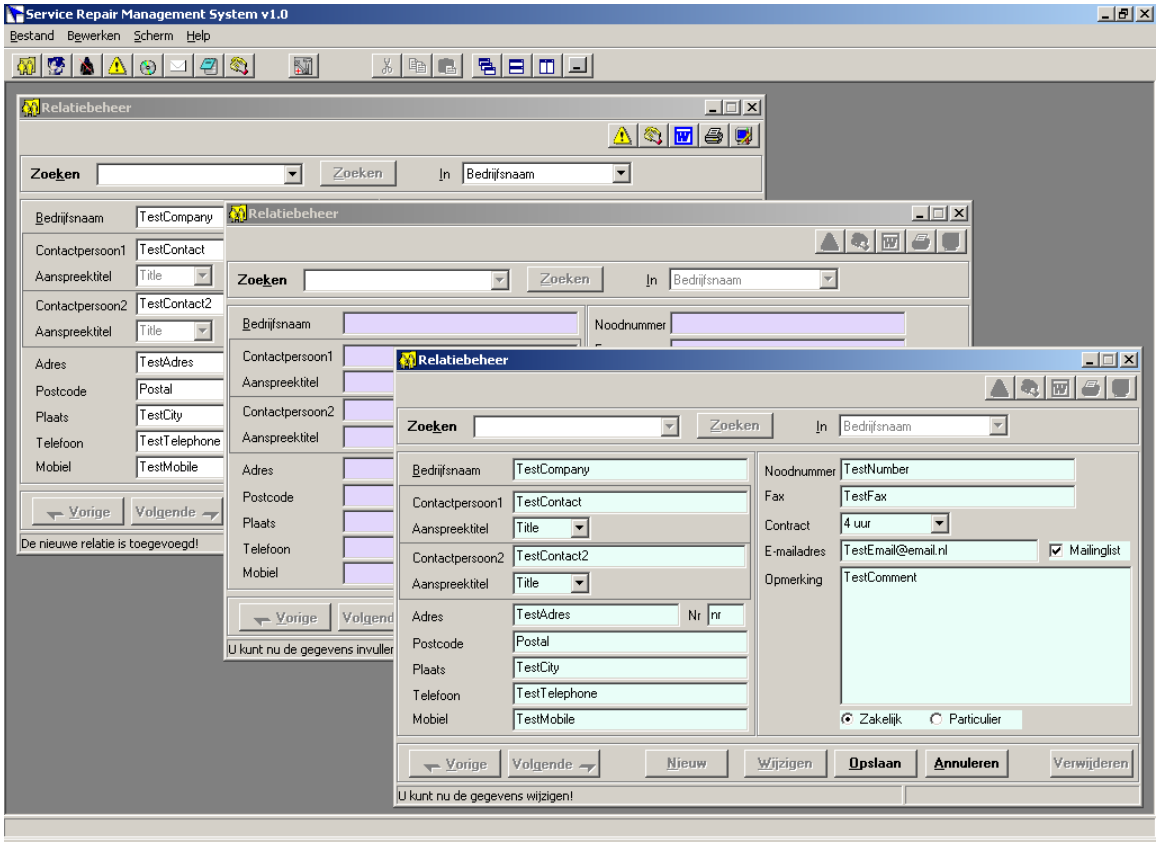
Figuur 8: ERD relatiebeheer

Voor het realiseren van het databasemodel is er van *TableCustomer* naar gerelateerde tabellen gebruik gemaakt van een OnDelete Cascades. Bij het verwijderen van een software-entry moet er eerst gekeken worden of deze niet bij een andere relatie voorkomt voordat deze kan worden verwijderd.

Interface design relatiebeheer

Voor het implementeren van het relatiebeheer is gebruik gemaakt van twee hoofdschermen waarmee alle gewenste informatie te zien en op te vragen is. Tijdens de planninggame zijn er schetsen gemaakt van deze twee hoofdschermen. Deze schetsen zijn direct uitgewerkt met de Visual Component Library (VCL) van Delphi. Vaak wordt het ontwerp van een scherm gemaakt in bijvoorbeeld Visio en na goedkeuring geïmplementeerd in de ontwikkelomgeving. Omdat de VCL-omgeving van Delphi erg uitgebreid en gemakkelijk te gebruiken is heb ik deze onnodige stap achterwege gelaten en de user interface gelijk in Delphi gemaakt. Dit met het voordeel dat er gelijk gebruik gemaakt kon worden van de componenten die Delphi te bieden heeft en niet de componenten die in bijvoorbeeld Visio beschikbaar zijn. Ook de keuring van de user interface is dus gebeurd aan de hand van de interface die gemaakt is in Delphi, wat tevens een beter beeld van de werkelijkheid geeft.

Tijdens het ontwerpen van de interfaces wilde ik zo efficiënt mogelijk met de beschikbare ruimte omspringen, want hoe kleiner de schermen, hoe prettiger de interface werkt met meerdere schermen tegelijk open. De visuele componenten die gebruikt zijn, zitten standaard in Delphi en kunnen op elk standaard Windows systeem worden gebruikt.



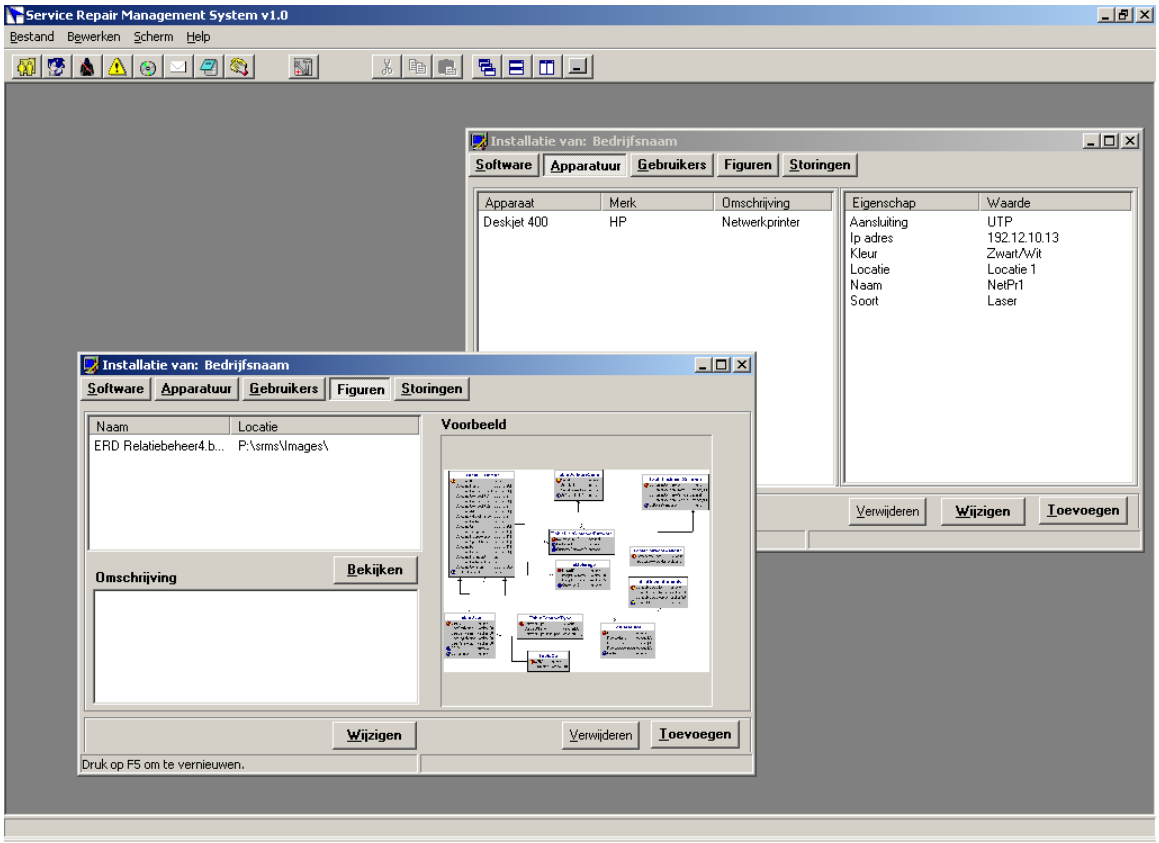
Figuur 9: user interface - 3 instanties van relatiebeheer

In figuur 9 is duidelijk te zien dat er gebruik gemaakt is van kleuren om de gebruiker duidelijk aan te geven dat deze zich bevindt in een andere modus, zoals het wijzigen of toevoegen van een klant. Tijdens het implementeren van de functionaliteiten heb ik gebruik gemaakt van een geavanceerd design time component van Delphi: TActionList. Een design time component houdt in dat deze alleen tijdens het ontwikkelen zelf zichtbaar is, maar tijdens executie alleen onzichtbaar op de achtergrond aanwezig is. TActionList kan verschillende acties bevatten die gekoppeld kunnen worden aan de meeste visuele componenten binnen Delphi. Door de actie te disabelen worden automatisch alle componenten die gelinkt zijn aan deze actie ook disabled. Dit is vooral handig wanneer één routine of actie door meerdere componenten kunnen worden gestart.

Om er voor te zorgen dat de gebruiker geen onbedoelde handelingen kan uitvoeren, heb ik er voor gezorgd dat de gebruiker alleen de handelingen kan starten die op dat moment toegestaan zijn. Zo is het bijvoorbeeld niet toegestaan om een zoekopdracht te geven als er een klant wordt gewijzigd of toegevoegd. In eerste instantie heb ik dit gedaan door een updateactie te maken die alle aanwezige acties aan de hand van de zelf opgestelde voorwaarden aan of uit zet. Het starten van deze update actie heb ik op “on-update” van het scherm gezet, wat inhoudt dat deze updateactie elke keer gestart wordt op het moment dat het scherm Idle is. Tijdens het implementeren ging het goed zodra er één scherm in de MDI-omgeving openstond, maar zodra er meerdere schermen openstonden ging het finaal fout met een vastlopende applicatie als resultaat. Hierdoor was ik genoodzaakt om zelf een “on-update” te maken die elke fractie van een seconde het scherm update. Hiervoor heb ik het component TTimer gebruikt. Met deze timer is het mogelijk om een tijdseenheid in te stellen en na elke tijdseenheid één of meerdere acties of procedures te starten. Door deze oplossing hoefde ik niet veel aan de huidige sourcecodes te veranderen, doordat het aan-/uitzetten van de acties gewoon hergebruikt kon worden. Ik heb initieel de tijdseenheid van het TTimer component gezet op 0,2 seconden. Om de oplossing helemaal netjes te maken heb ik het dusdanig opgezet dat het TTimer component alleen actief is wanneer het betreffende scherm gefocused is. Een update van een scherm is onnodig wanneer er geen veranderingen plaats kunnen vinden tijdens het gebruik van een ander scherm. Na het realiseren van deze oplossing waren

alle problemen verholpen en kon ik terugkijken op een goed resultaat dat ook bij de andere releases kon worden gebruikt.

Interface design configuratie/installatiebeheer
In figuur 10 is duidelijk te zien hoe schermen voor het configuratie/ installatiebeheer zijn uitgewerkt.



Figuur 10: user interface - relatie configuratie /installatiebeheer

Het scherm is opgedeeld in de 5 hoofdgroepen die samen de hele installatie van de relatie representeren. Deze verdeling is gemaakt met behulp van een TTabcontrol. Dit in de praktijk veel gebruikte Windows control maakt het mogelijk om een scherm te verdelen in verschillende delen die allemaal apart kunnen worden weergegeven. Ook hier zijn alle handelingen verwerkt in acties die aan/uitgezet worden door een updateactie op het scherm.

Met de user interface is het mogelijk om per relatie één of meerdere afbeeldingen op te slaan. Deze afbeeldingen kunnen bijvoorbeeld betrekking hebben tot het netwerkmodel of de organisatorische inrichting bij de klant. De afbeeldingen zouden fysiek in de database kunnen worden opgeslagen. Ik heb er voor gekozen om het betreffende plaatje te kopiëren naar een sub-directory van de applicatie. Hierbij heb ik als uitgangspunt genomen dat de applicatie van een server gaat werken die voor iedereen binnen het bedrijfsnetwerk toegankelijk is. Door het opslaan van de plaatjes in een sub-directory is het dus mogelijk dat iedereen gebruik kan maken van deze afbeeldingen.

Implementatie SQL
Omdat er gekozen is voor het gebruik van een DBMS is het noodzakelijk te werken met SQL-statements. Met deze statements moeten de functionaliteiten zoals toevoegen, wijzigen, verwijderen en opvragen van gegevens worden ontwikkeld. Het gebruiken van de SQL-statements kon bij dit project op verschillende manieren worden geïmplementeerd. Een veel gebruikte methode is het aanmaken van de statements op de SQL server zelf en

deze vanuit de applicatie te starten. Ik heb echter gekozen om de SQL-statements binnen Delphi te houden en deze rechtstreeks te maken en te starten vanuit de sourcecodes. Het voordeel hiervan is, dat de gehele ontwikkeling lokaal plaats kan vinden met één ontwikkelomgeving. Het nadeel van deze manier van ontwikkelen is dat de query niet geanalyseerd kan worden op juistheid omdat de statements runtime worden gegenereerd. Dit nadeel wordt echter opgevangen door de geautomatiseerde tests.

```
procedure TFormCustomer.ExecuteCustomerSearch (
const aSearch,
      aCategory : string
);
begin
  with Self.SQLCustomer do begin
    Close;
    SQL.Clear;
    SQL.Add ('select * from TableCustomer where');
    SQL.Add (aCategory + ' LIKE ''%' + aSearch + '%''');
    Open;
  end;
end; // TFormCustomer.ExecuteCustomerSearch
```

Het aanmaken van een SQL-statement met Delphi

Om de geautomatiseerde tests goed te laten verlopen was het noodzakelijk om voor uitvoering van de tests de database geheel leeg te maken. Hiervoor is in Testcomplete een routine gemaakt die uitgevoerd wordt voordat het werkelijke testen plaatsvindt.

7 Release 2 Storingbeheer

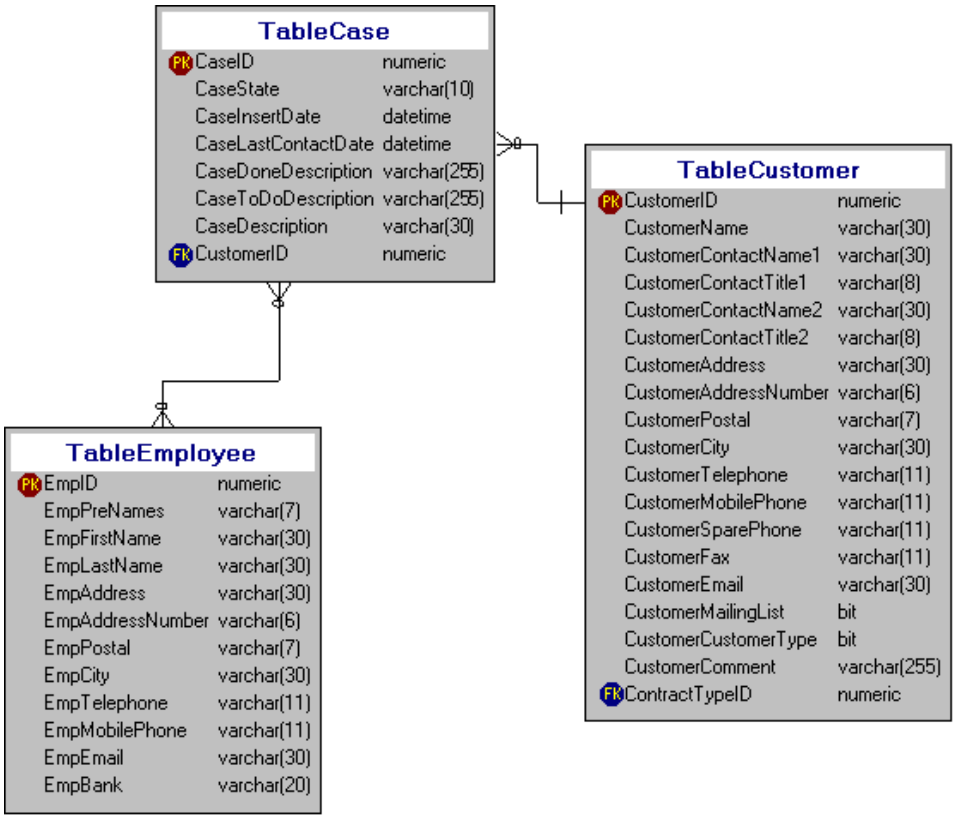
Inleiding

Tijdens de planninggame kwam duidelijk naar voren dat er een goede oplossing moest komen voor het beheer van alle storingen die bij VCS Netwerk Division voorkomen. Deze storingen omvatten dus ook de storingen die zich intern voordoen. Een storing kan gezien worden als een taak. Elke storing moet aan een interne werknemer worden toegewezen die verantwoordelijk is voor de betreffende storing. Hierdoor was de taak om het beheer van de medewerkers toe te voegen aan deze release. Verder werd tijdens de planninggame en overleg duidelijk dat ook deze user interface van alle gemakken voorzien moest worden.

Ontwerp

Databasemodel

Omdat we de storingen voor elke relatie willen bijhouden en beheren was het noodzakelijk om een link te leggen met de tabel *TableCustomer* die staat uitgewerkt in release 1. Na het maken van een schets van het databasemodel kwamen we tot de conclusie dat alle wensen ingedekt waren, behalve de mogelijkheid om storingen van VCS Network Division zelf te beheren. Om het model zo praktisch en efficiënt mogelijk te houden heb ik voorgesteld om VCS Network Division als klant op te nemen in de relatiedatabase, om de mogelijkheid te creëren om storingen voor VCS Network Division te beheren. Het databasemodel voor storingbeheer is als volgt geïmplementeerd:



Figuur 11: ERD storingbeheer

Voor het correct beheren van een storing is het nodig om per storing tweemaal een datum op te slaan in de database. Dit betreft de velden CaseInsertDate en CaseLastContactDate. Een datum kan op verschillende manieren worden opgebouwd, waardoor ik heb besloten een datum als string in de database op te slaan in plaats van een datumvariabele die wordt voorgeschreven door Microsoft SQL Server. Het voordeel hiervan is

dat ik zelf kan bepalen hoe de datum wordt opgeslagen en dat er geen problemen kunnen ontstaan met verschillende systemen die een datum anders opbouwen. Kortom, de applicatie verzorgt zelf het beheer van de datumopbouw. De opbouw van een datum in Delphi is erg gemakkelijk en snel toe te passen. Ook is het mogelijk om de opgeslagen datumstring weer terug te converteren naar een echte datum.

Het formatteren van een datum naar een string

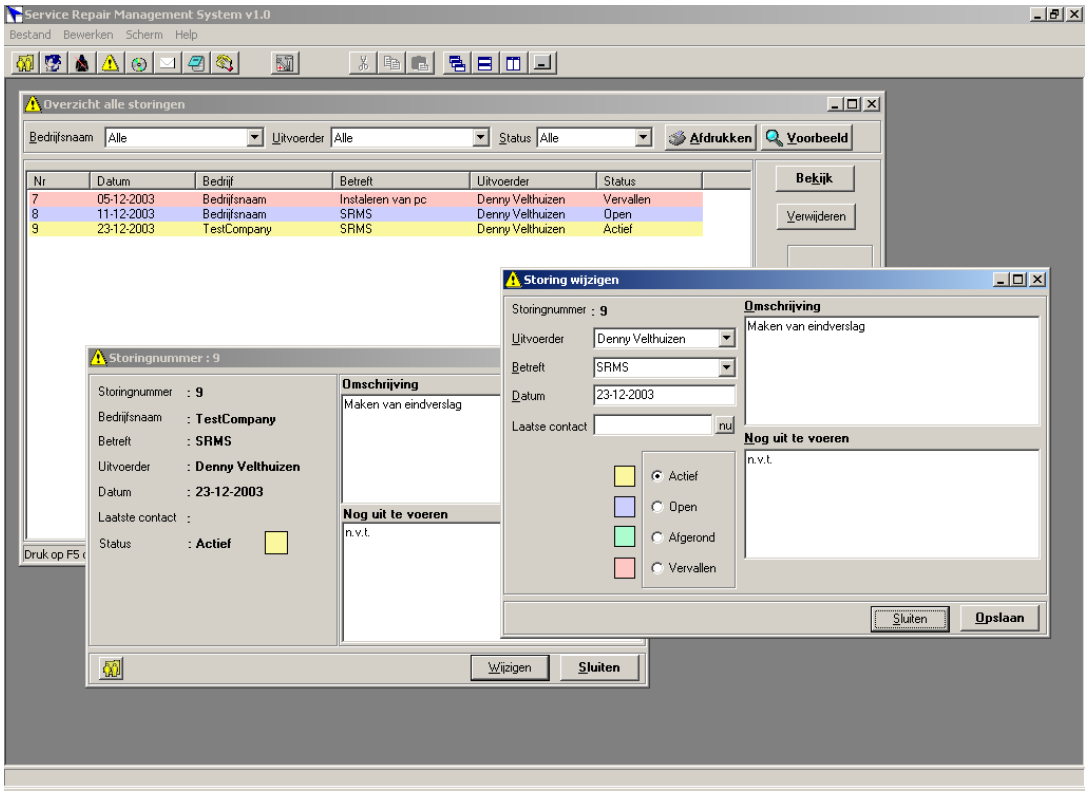
In het databasemodel is te zien dat elke storing een status heeft. Tijdens een overleg is er besloten om een viertal statussen voor een storing te hanteren: *Open*, *Actief*, *Afgerond* en *Vervallen*. Deze statussen heb ik tijdens het ontwikkelen van de user interface uitgebreid gebruikt. Deze statussen moesten vooraf dus definitief vast staan.

Interface design storingbeheer

Alle eisen die aan de user interface gesteld werden zijn ook uitgewerkt. De belangrijkste zijn:

- Het laten zien van een overzicht van de storingen
- Een overzicht van storingen specificeren op een combinatie van medewerker, bedrijf en status
- Het afdrukken van het overzicht
- Het snel en gemakkelijk wijzigen van de status van een storing
- Sorteren van het overzicht

Om deze wensen te verwerken in één interface heb ik gebruik gemaakt van TListView. Dit is een Delphi-component die het mogelijk maakt om een lijst weer te geven met verschillende kolommen, in dit geval storingen. Om optimaal gebruik te maken van TListView heb ik gebruik gemaakt van kleuren die per storing laat zien welke status de betreffende storing heeft.



Figuur 12: user interface - storingbeheer

Toevoegen storing

Bij het scherm van relatiebeheer is een knop  toegevoegd om een storing toe te kunnen voegen. De storing wordt direct gekoppeld aan de betreffende relatie. Een storing *moet* gekoppeld zijn aan een relatie dus is dit de enige manier om een storing toe te voegen.

Genereren overzichten

Een belangrijk onderdeel van deze release is het dynamisch aanmaken van het SQL-statement. Omdat er verschillende views mogelijk zijn moet het SQL-statement opgebouwd worden aan de hand van de selectie betreffende *bedrijfsnaam*, *uitvoerder* en *status*. Het resultaat, de query, moet direct worden weergegeven in de ListView. Daarnaast moet de lijst kunnen worden gesorteerd door middel van het aanklikken van de titelbalk van de betreffende kolom. TListview kent zelf een sorteeroptie waarmee dit mogelijk is, alleen heb ik hier geen gebruik van gemaakt. Dit heb ik gedaan, omdat het de bedoeling is dat de storingen die zichtbaar zijn in de betreffende view ook afgedrukt kunnen worden. Het afdrukken van een formulier gebeurt aan de hand van een SQL-resultaat en niet aan de hand van de volgorde die het TListView component laat zien. Hierdoor heb ik besloten om het sorteren van de view in het SQL-statement op te nemen en telkens uit te voeren wanneer een kolombalk wordt aangeklikt. Dit alles met het resultaat dat de view afgedrukt kan worden aan de hand van de sorteerkeuze (bijvoorbeeld datum of werknemer).

Voor het maken en afdrukken van formulieren zijn er in Delphi een aantal componenten beschikbaar. In eerste instantie wilde ik gebruik maken van Quickreport waarmee het maken en beheren van formulieren mogelijk is, maar tijdens implementatie bleken er fouten (bugs) te zitten in het afdrukvoorbeeld van de formulieren gemaakt met Quickreport. Hierdoor was ik genoodzaakt om een andere component te gebruiken. Na een hoop uitzoekwerk heb ik besloten om het freeware component FastReport te gebruiken. De opzet en gebruik van dit component is nagenoeg hetzelfde als Quickreport en kon daardoor vrij makkelijk worden gebruikt. Na het gebruik van dit component waren alle problemen verholpen. Het oplossen van dit probleem heeft me echter drie werkdagen gekost.

8 Release 3 Intern softwarebeheer

Inleiding

Bij VCS Network Division is een Access database aanwezig waarin alle gegevens van de interne softwarebank staan. Om SRMS zo compleet mogelijk te maken was het de bedoeling om deze database te verwerken in de applicatie. Aangezien het hier gaat om een vrij eenvoudige database en user interfacemodel zal ik niet te diep ingaan op de technische details.

Ontwerp

Het ontwerpen van het databasemodel was geen enkel probleem, omdat deze slechts bestond uit twee simpele tabellen. De tabelopbouw is grotendeels overgenomen van de bestaande tabellen met hier en daar een kleine aanpassing. Deze aanpassingen betroffen alleen velden die niet meer gebruikt werden.

Bij het ontwerpen en implementeren van de user interface is gebruik gemaakt van de opbouw die is gebruikt bij het relatiebeheer.

Omdat alle gegevens betreffende de interne software al in een database staan, was er een verzoek deze gegevens éénmalig te importeren in de nieuwe database. Hiervoor heb ik een losstaande applicatie gemaakt die deze actie voor zijn rekening nam. Omdat het hier een éénmalige actie betrof, is er geen energie gestoken in de user interface en dergelijke. Ook zijn er geen tests gemaakt voor deze kleine applicatie. Na het eenmalig importeren van de gegevens konden de werknemers van VCS Network Division gebruik maken van het nieuwe gedeelte *softwarebeheer* in de applicatie.

9 Release 4 Telefoonafhandeling

Inleiding

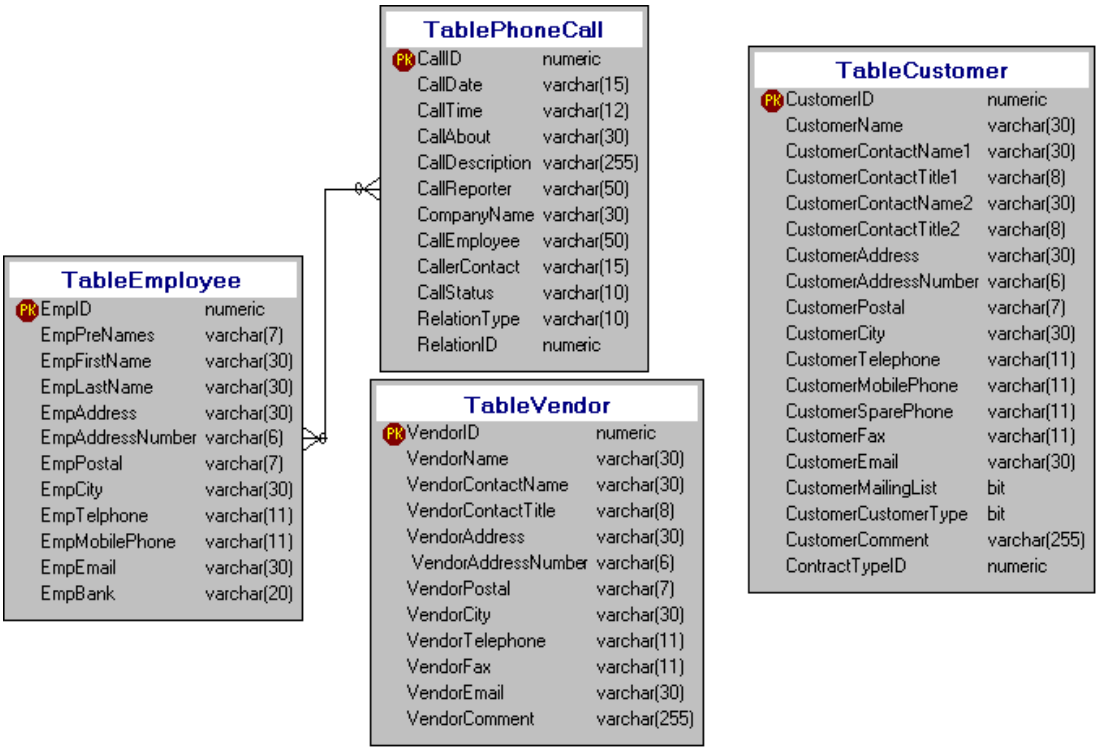
De telefoon is bij een VCS Network Division een belangrijk onderdeel van het primaire bedrijfsproces. Zo worden vaak afspraken met klanten gemaakt en wordt er regelmatig support geleverd via de telefoon. Het efficiënt beheren van deze gesprekken kan erin resulteren dat de werknemers van VCS Network Division de klant beter kunnen voorzien van informatie en support door het teruglezen van eerder gevoerde gesprekken. Bij deze support kan gedacht worden aan instellingen van software en hardware.

Omdat sommige gesprekken werknemergericht zijn én VCS Network Division werkt met buitendienstmedewerkers, komt het voor dat er van gesprekken notities gemaakt moeten worden. Het efficiënt beheren van telefonische en interne communicatie bij VCS Network Division is het uitgangspunt van deze release. Om deze release zo volledig mogelijk te kunnen uitvoeren is er voor gekozen om ook het leverancierbeheer in deze release op te nemen.

Ontwerp

Databasemodel

Bij het ontwerpen van het databasemodel is rekening gehouden met alle soorten scenario's. Zo moet het mogelijk zijn om gesprekken te registreren van klanten, leveranciers maar ook gesprekken van personen die niet direct gerelateerd zijn met VCS Network Division. Om dit alles te realiseren zijn er twee velden opgenomen in de tabel *TablePhoneCall*: RelationID en RelationType. Deze velden moeten duidelijk maken om wat voor relatie het gaat. Tevens is het hiermee mogelijk om met enkele handelingen alle gegevens van de relatie of leverancier op te vragen. Als het veld *RelationID* leeg is (ik gebruik altijd de waarde –1) dan gaat het om een persoon die niet direct gerelateerd is aan VCS Network Division. Het databasemodel ziet er als volgt uit:



Figuur 13: ERD telefoonafhandeling

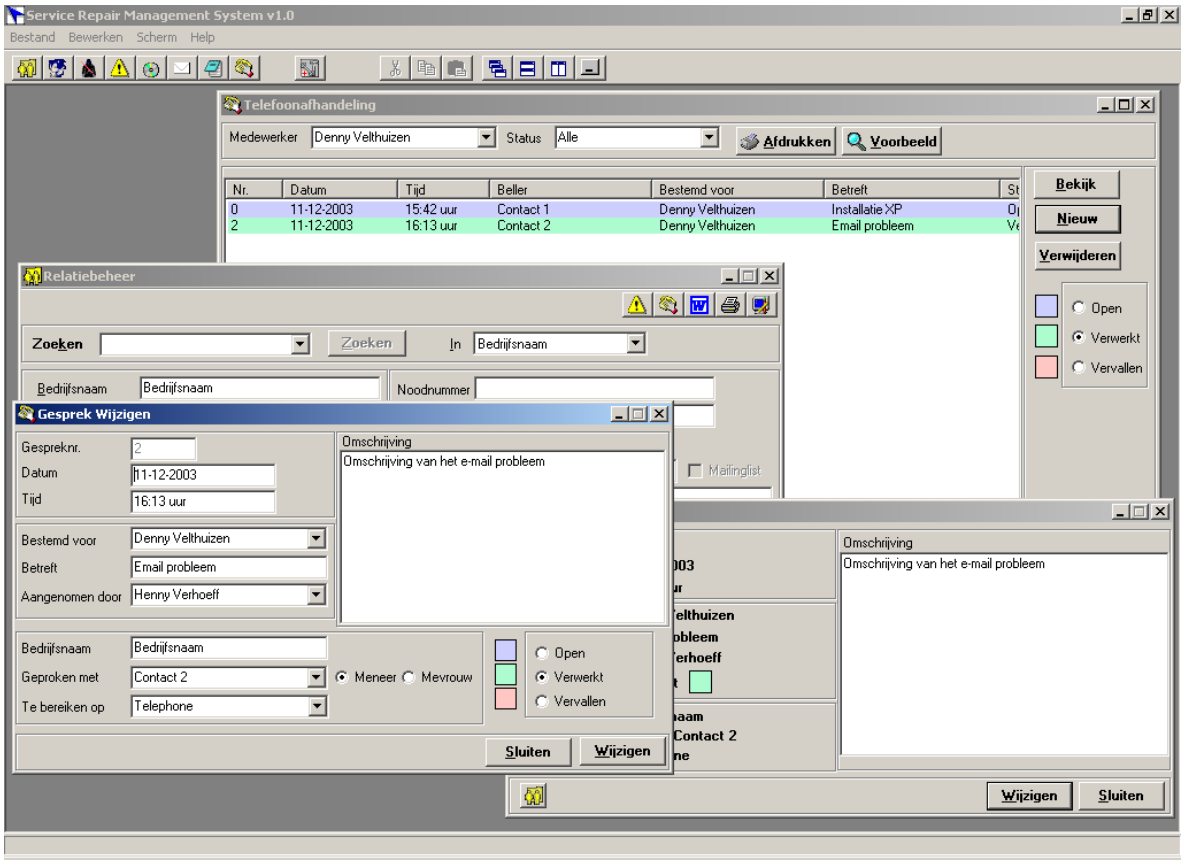
In het ontwerp is duidelijk te zien dat er geen fysieke koppeling gemaakt is van TableVender en TableCustomer naar TablePhoneCall. De koppeling is virtueel wel aanwezig alleen wordt deze door sourcecodes tot stand gebracht.

Interface design telefoonbeheer

Voor het maken van de interface is gebruik gemaakt van de standaarden die zijn opgesteld in de release *storingbeheer*. Dit houdt in dat er wederom gebruik gemaakt is van een TListView. De interface van *telefoonbeheer* kent dezelfde functionaliteiten als de release *storingbeheer*.

- Het laten zien van een overzicht van de gesprekken
- Een overzicht van gesprekken specificeren op een combinatie van medewerker, bedrijf en status
- Het afdrukken van het overzicht
- Het snel en gemakkelijk wijzigen van de status van een gesprek
- Sorteren van het overzicht

Het invoeren van de gesprekken moet vanaf verschillende plaatsen gedaan kunnen worden. Deze plaatsen zijn het scherm van relatiebeheer, het scherm van leverancierbeheer en het scherm waar het overzicht van de gesprekken weergegeven wordt.



Figuur 14: user interface telefoonbeheer

Bij relatiebeheer en leverancierbeheer is een knop  geplaatst waarmee een gesprek kan worden toegevoegd. Het scherm zal automatisch de bedrijf- of leveranciergegevens invullen waardoor het invoeren snel en gemakkelijk plaats kan vinden. Ook wordt het gesprek automatisch gekoppeld aan de betreffende klant of

leverancier. Als een gesprek wordt toegevoegd vanuit het overzicht scherm, zal er geen enkele verwijzing gemaakt worden.

Net als bij het storingbeheer is er gekozen om een aantal statussen te hanteren voor een telefoongesprek. Deze zijn: *Open*, *Verwerkt* en *Vervallen.*

10 Applicatie add ons

Na het afronden van de tweede release werd duidelijk dat het project sneller verliep dan aanvankelijk was verwacht. In de eerste release is veel tijd gaan zitten in het opzetten van de ontwikkelomgeving en het uitzoeken van de werking van de Microsoft SQL Server en het Delphi-component die ermee communiceert. Nadat dit met succes was uitgezocht kon er in snel tempo ontwikkeld worden, waardoor veel tijdwinst is behaald. Dit resulteerde in aanvullende functionaliteiten die in veel gevallen los stonden van de geplande releases. Omdat XP een dynamische planning kent, zijn enkele van deze functionaliteiten tussen de huidige planning door ontwikkeld. Hier volgt een overzicht van deze functionaliteiten:

- Exporteren gehele SRMS database (i.v.m back-up)
- Importeren gehele SRMS database (i.v.m back-up)
- Instellen van omgevingvariabelen zoals kleuren
- Instellen van SQL-verbinding, gebruiker, wachtwoord e.d.

Deze aanvullende functionaliteiten zijn allemaal met succes ontwikkeld.

11 Evaluatie

11.1 Procesevaluatie

Gekeken naar het verloop van het gehele project kan ik stellen dat het project op alle fronten geslaagd is. Waar vaak problemen met de planning optreden had ik tijdens het project veel tijd om aanvullende eisen van de opdrachtgever te verwerken in de applicatie. Tijdens het project zijn een klein aantal kleine onvoorziene problemen opgetreden. Deze problemen betroffen onder andere een fout in Delphi en een fout in een third party component.

De samenwerking met de opdrachtgevers verliep erg goed. Al vanaf het begin was duidelijk wat de opdrachtgevers met de applicatie wilden en zij konden dit tijdens de verschillende whiteboard gesprekken goed duidelijk maken. Ik denk dat mede hierdoor het gebruik van XP als ontwikkelmethode een goede keuze was. Een goede samenwerking tussen opdrachtgever, uitvoerder en begeleider was een goede basis voor deze ontwikkelmethode.

Ondanks dat ik al vrij veel kennis van en ervaring met Delphi had en een redelijke kennis van XP, ben ik toch met veel nieuwe programmeermethoden en technieken in aanraking gekomen. Vrijwel alle practices van XP zijn gebruikt tijdens dit project, behalve pair programming. Het ontwikkelen in tweetallen is niet altijd van toepassing geweest. Dit kwam voornamelijk doordat de andere projectleden ook deelnamen aan andere projecten. Wel werd er dagelijks met de opdrachtgever naar de applicatie en haar voortgang gekeken.

Een sleutelwoord voor het gebruik van XP is *discipline*. Ik ben van mening dat XP een hele volwassen methode is die zorgvuldig moet worden gebruikt om tot een goed resultaat te komen. Als alle practices goed worden uitgevoerd, denk ik dat XP met de meeste vraagstukken van critici raad weet.

Een ander belangrijk aspect van XP is het *weten* hoe goed je bent als programmeur, en het *denken* hoe goed je bent als programmeur. Het is heel verleidelijk om te vertrouwen op je eigen programmeerkunsten, met als resultaat dat je snel de neiging hebt om de test(s) voor een functionaliteit achterwege te laten. In mijn ogen kan dat in sommige gevallen ook, maar dat betreft dan alleen onderdelen die net zo ingewikkeld zijn als de test die er voor gemaakt zou moeten worden. Tijdens het project heb ik gemerkt dat het erg belangrijk is hoe er met problemen omgegaan wordt. Zo moet een goede programmeur altijd de juiste beslissingen kunnen nemen en in staat zijn op de juiste plekken naar oplossingen te zoeken.

11.2 Productevaluatie

Naar mijn mening is er een heel compleet en goedwerkende applicatie opgeleverd. Daarnaast vind ik dat de applicatie professioneel is opgezet en is ontwikkeld met een hele krachtige combinatie van Delphi en Microsoft SQL Server. Er is veel tijd besteed aan de verschillende user interfaces, wat resulteerde in een goedogend en prettig werkend programma. Ook vind ik dat de beslissing voor het gebruik van een MDI-omgeving zeer goed heeft uitgepakt.

De opdrachtgevers zijn meer dan tevreden met het product en is ondertussen al in gebruik genomen. Mochten er na afloop van het project nog problemen ontstaan, kan VCS Network Division altijd terugvallen op de sourcecodes en tests om eventueel aanpassingen te maken.

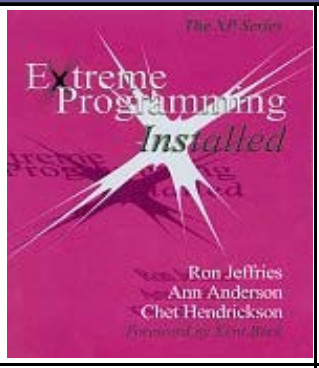
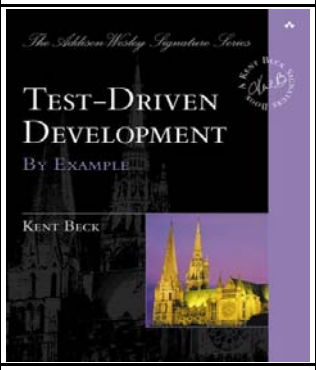
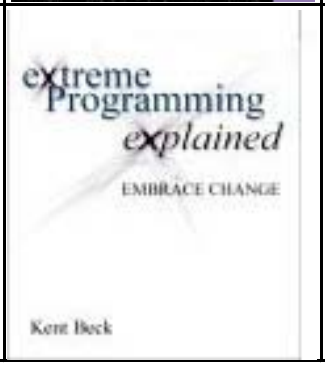
Opmerkelijk feit: de bugs en aanpassingen betreffende het Service Repair Management System worden door de gebruikers ervan ingevoerd bij *storingbeheer* voor medewerker Denny Velthuizen. Dit geeft de kwaliteit aan van een XP product!

Begrippen en Definities

Hieronder zijn de begrippen en definities opgenomen om begripsverwarring te voorkomen.

Begrip	Verklaring
XP	E**x**treme **P**rogramming – Een ontwikkelomgeving gebaseerd op testen.
SRMS	**S**ervice **R**epair **M**anagement **S**ystem – De naam van het project.
DBMS	**D**ata**b**ase **M**anagement **S**ystem - Het systeem dat de database beheerd.
VCL	**V**isual **C**omponent **L**ibrary – Een verzameling van visuele grafische componenten voor het ontwikkelen van software.
Borland Delphi	Een grafische ontwikkelomgeving voor Windows applicaties.
TListView	Een grafisch component uit de VCL waarmee een tabel op het scherm kan worden weergegeven.
MDI	**M**ultiple **D**ocument **I**nterface – Een interfacetechniek voor het weergeven van meerdere schermen in één applicatie.
Runtime	Is tijdens het uitvoeren van een programma.
Designtime	Is tijdens het ontwikkelen van een programma.
TTimer	Een Design Time component van Delphi dat werkt met de systeemtijd.
Planninggame	Een onderdeel van XP waarin alle aspecten van het project worden besproken. Deze planninggame vindt plaats aan het begin van het project.
Pair programming	Met tweetallen achter één ontwikkelomgeving werken.
TestComplete	Een applicatie waarmee Windows applicaties getest kunnen worden.
Dunit	Een stuk code waarmee Delphi-routines getest kunnen worden.
SQL	**S**tandard **Q**uery **L**anguage – Een taal waarmee vragen gesteld kunnen worden aan een database.
SQL-statement	Een vraag aan een database.
TActionList/ ActionList	Een Designtime component van Delphi waarmee het mogelijk is om acties op een formulier of programma te beheren.
TAction/ Action	Een omhulsel voor het starten van verschillende routines. Deze kunnen beheerd worden in de TActionList.
Third party / 3rd party	Een externe leverancier van software en/of componenten.
UML	**U**nified **M**odeling **L**anguage – Een hulpmiddel bij het ontwikkelen van OO-systemen. Omvat o.a. Klassendiagrammen, usecase diagrammen en beschrijvingen en sequence diagrammen.
ERD	**E**ntity **R**elation **D**iagram
Idle	Het realtime moment dat er geen interactie met het systeem is.

Literatuurlijst

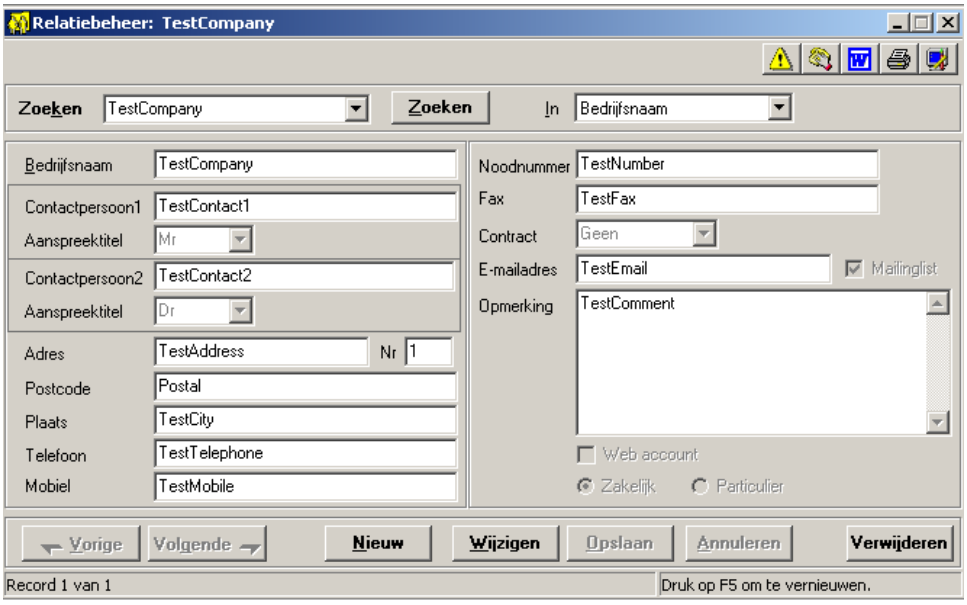
ISBN-nummer	Boek & Titel		Auteur
0-201-70842-6		**Extreme programming installed (The XP-series)**	Ron Jeffries Ann Anderson Chet Hendrickson
0-321-14653-0		**Test-Development By Example**	Kent Beck
0-201-61641-6		**Extreme Pogramming Explained (The XP-series)**	Kent Beck

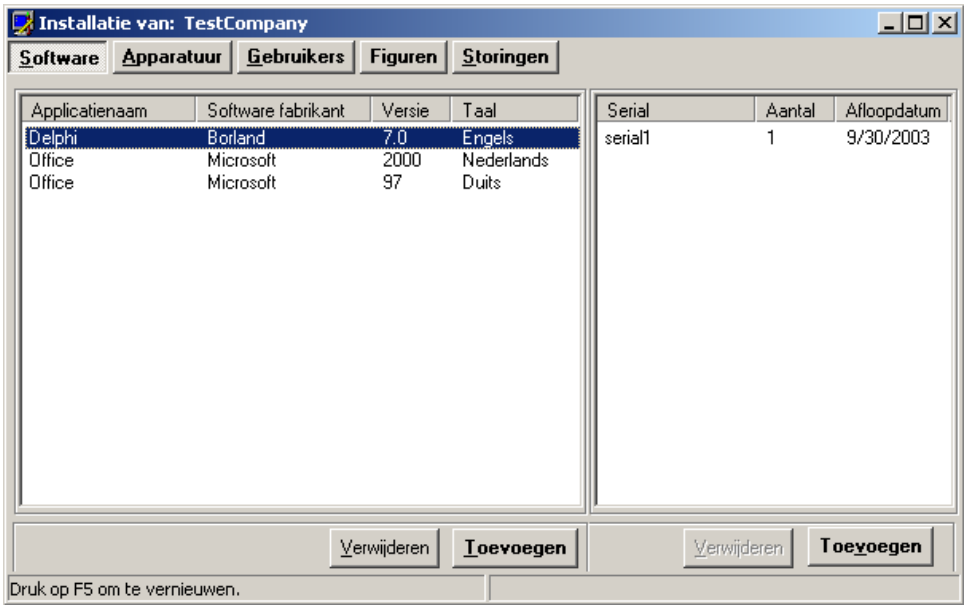
Naast de bovenstaande boeken heb ik ook de onderstaande internetsites gebruikt. Deze internetsites hebben mij goed geholpen bij het ontdekken van Extreme Programming en het vinden van resources betreffende Delphi en Testcomplete. De aanwezigheid en werking van deze sites kunnen niet worden gegarandeerd.

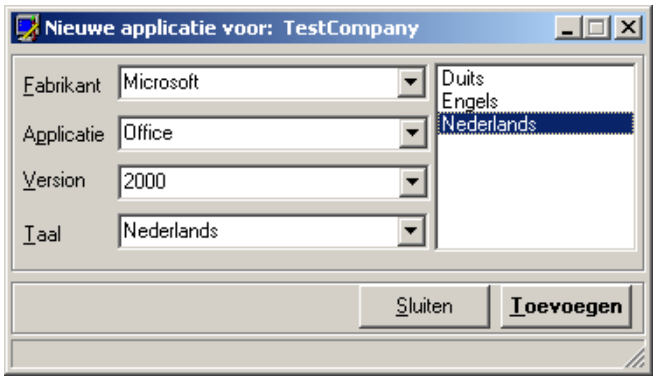
URL	Betreft
http://www.objectmentor.com/resources/articles/why.html	Hoofdlijnen van Extreme Programming
http://www.xprogramming.com/xpmag/expDocumentationInXP.htm	Documentatie tijdens het XP traject
http://www.developer.com/java/other/article.php/761841	Globale uitleg Extreme Programming
http://www.automatedqa.com/products/tc.asp	Testomgeving Testcomplete
http://dunit.sourceforge.net/	Testomgeving D-unit
http://www.torry.net	Delphi resources (tip)

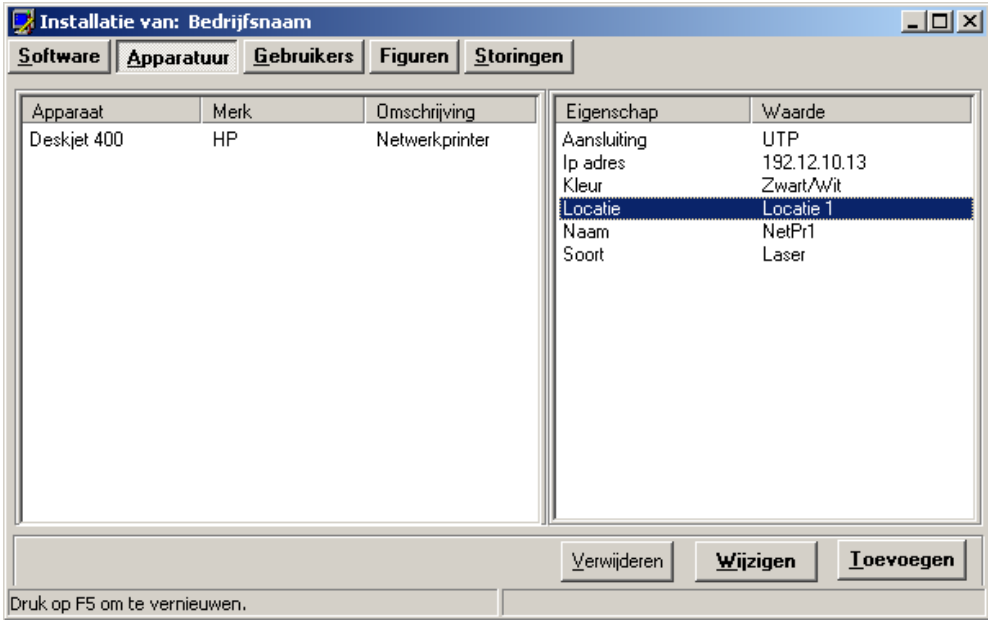
Bijlage I Schermen

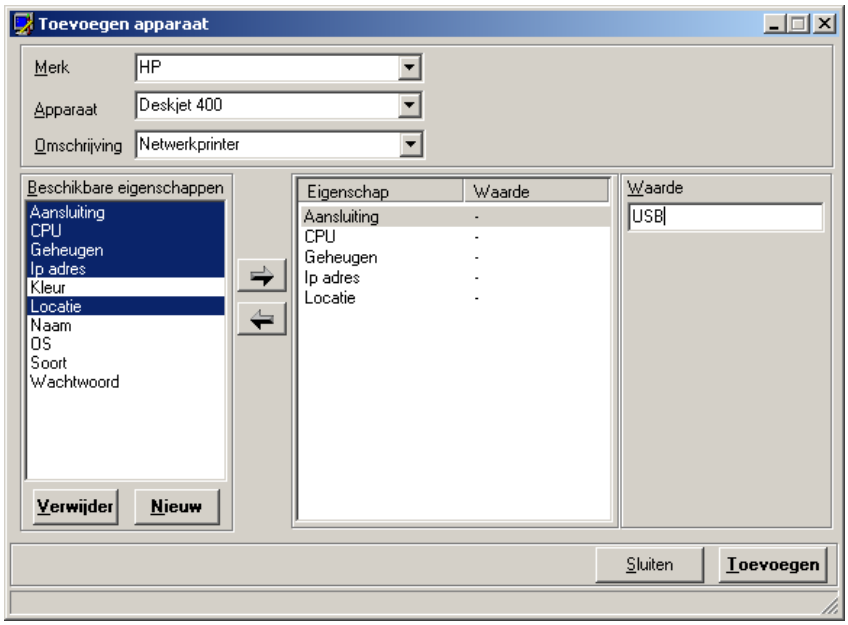
Deze bijlage bevat screenshots van de schermen die horen bij het Service Repair Management System.

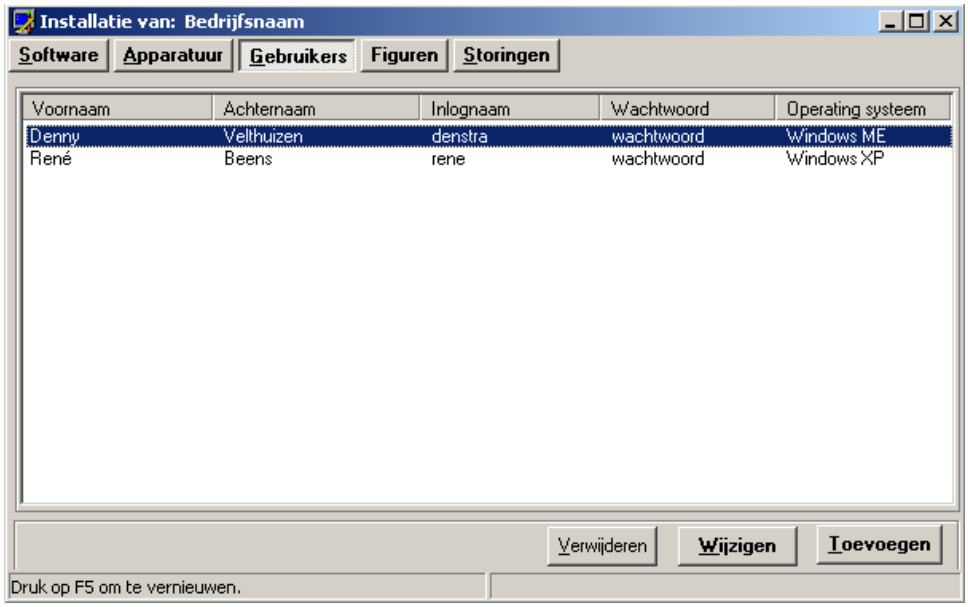




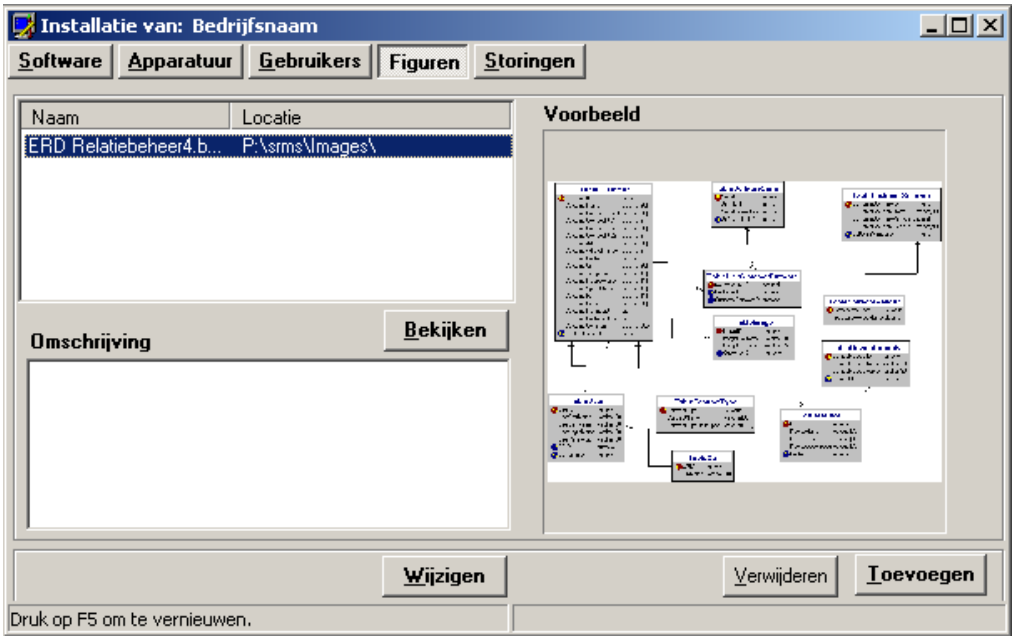


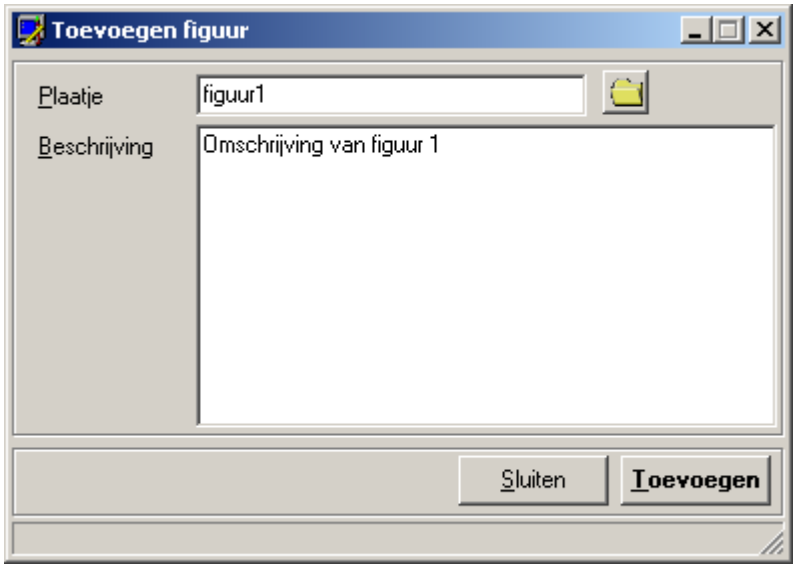

Installatie van: Bedrijfsnaam
Software
Apparatuur
Gebruikers
Figuren
Storingen
Apparaat
Merk
Omschrijving
Deskjet 400
HP
Netwerkprinter
Eigenschap
Waarde
Aansluiting
UTP
Ip adres
192.12.10.13
Kleur
Zwart/Wit
Locatie
Locatie 1
Naam
NetPr1
Soort
Laser
Verwijderen
Wijzigen
Toevoegen
Druk op F5 om te vernieuwen.


Toevoegen apparaat
Merk
HP
Apparaat
Deskjet 400
Omschrijving
Netwerkprinter
Beschikbare eigenschappen
Aansluiting
CPU
Geheugen
Ip adres
Kleur
Locatie
Naam
OS
Soort
Wachtwoord
Verwijder
Nieuw
Eigenschap
Waarde
Aansluiting -
CPU -
Geheugen -
Ip adres -
Locatie -
Waarde
USB
Sluiten
Toevoegen

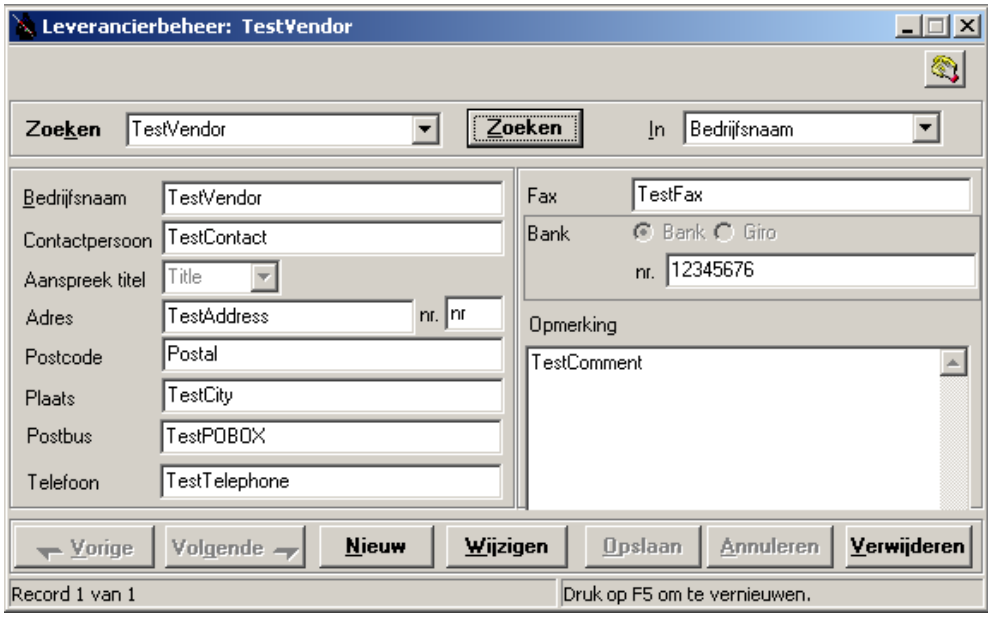

Installatie van: Bedrijfsnaam
Software
Apparatuur
Gebruikers
Figuren
Storingen
Voornaam
Achternaam
Inlognaam
Wachtwoord
Operating systeem
Denny
Velthuizen
denstra
wachtwoord
Windows ME
René
Beens
rene
wachtwoord
Windows XP
Verwijderen
Wijzigen
Toevoegen
Druk op F5 om te vernieuwen.

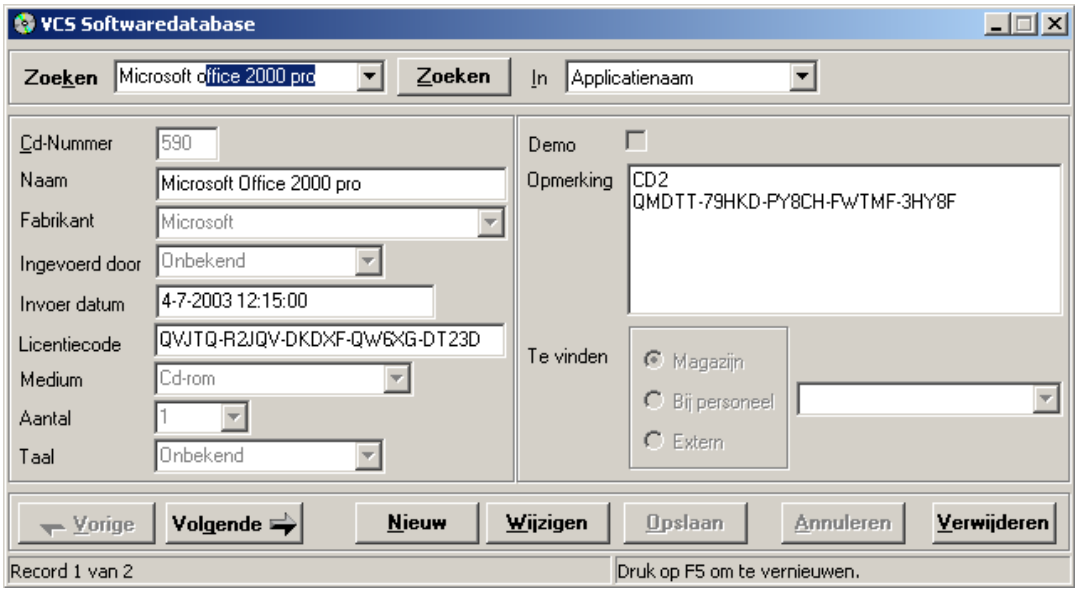

Toevoegen gebruiker
Voornaam
Henny
Achternaam
Verhoeff
Loginnaam
henny
Wachtwoord
wachtwoord
Operatingsysteem
Windows 98
Sluiten
Toevoegen

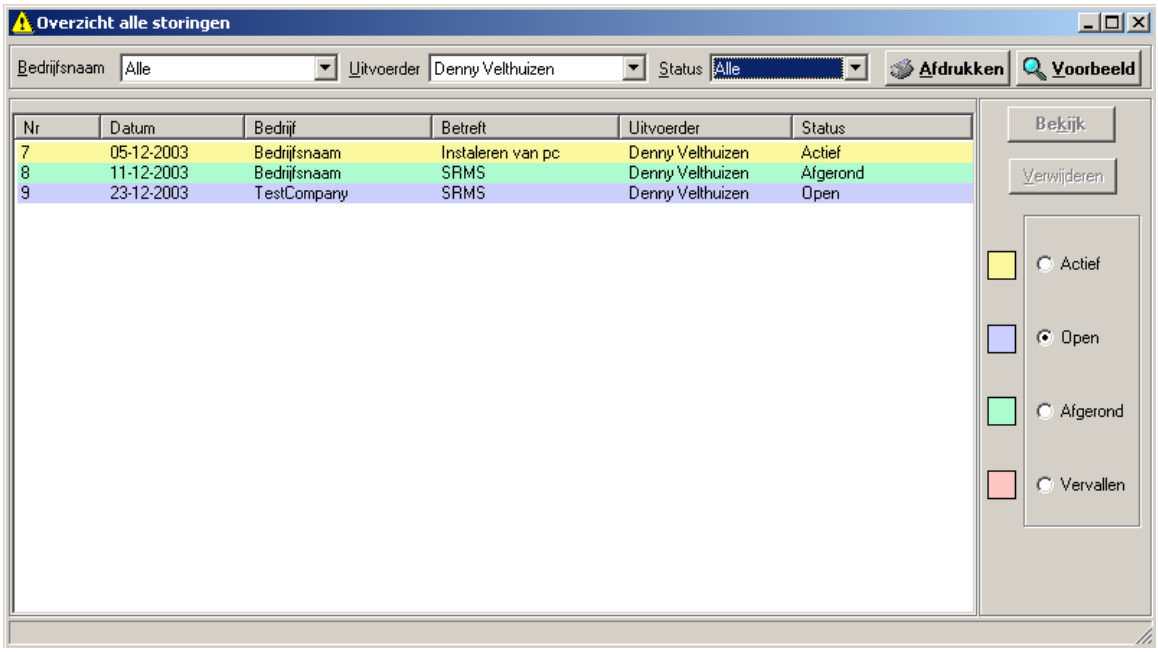

Installatie van: Bedrijfsnaam
Software
Apparatuur
Gebruikers
Figuren
Storingen
Naam
Locatie
ERD Relatiebeheer4.b...
P:\srms\Images\
Voorbeeld
Omschrijving
Bekijken
Wijzigen
Verwijderen
Toevoegen
Druk op F5 om te vernieuwen.


Toevoegen figuur
Plaatje
figuur1
Beschrijving
Omschrijving van figuur 1
Sluiten
Toevoegen


VCS werknemers
Voornaam
Achternaam
Denny
Velthuizen
René
Beens
Henny
Verhoeff
Voorletters
D.
Voornaam
Denny
Achternaam
Velthuizen
Adres
W.van Essenlaan
Nr
18
Postcode
3734 BZ
Plaats
Den Dolder
Telefoon
030-2285014
Mobiel
06-29271100
Email
DennyVelthuizen@wanadoo.nl
Bankrekening
01564842
Sluiten
Verwijderen
Annuleren
Opslaan
Wijzigen
Toevoegen
Druk op F5 om te vernieuwen.

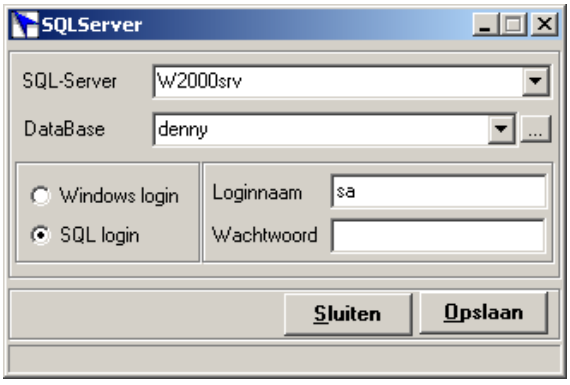

Leverancierbeheer: TestVendor
Zoeken
TestVendor
Zoeken
In
Bedrijfsnaam
Bedrijfsnaam
TestVendor
Contactpersoon
TestContact
Aanspreek titel
Title
Adres
TestAddress
nr.
nr
Postcode
Postal
Plaats
TestCity
Postbus
TestPOBOX
Telefoon
TestTelephone
Fax
TestFax
Bank
Bank
Giro
nr.
12345676
Opmerking
TestComment
Vorige
Volgende
Nieuw
Wijzigen
Opslaan
Annuleren
Verwijderen
Record 1 van 1
Druk op F5 om te vernieuwen.

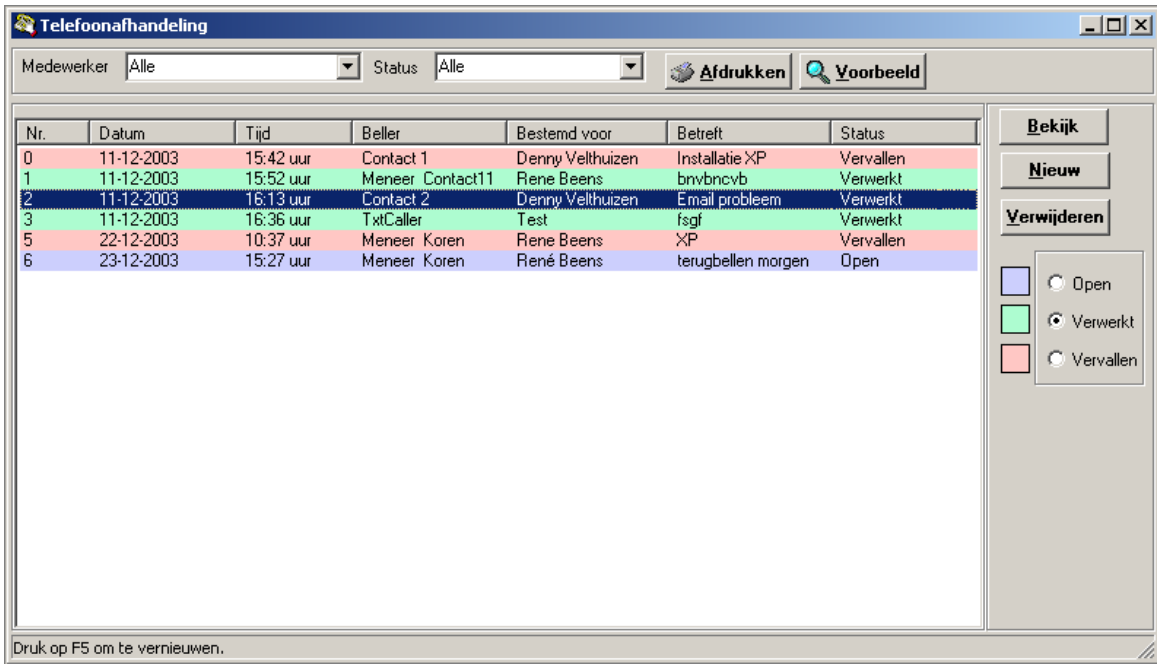

VCS Softwaredatabase
Zoeken
Microsoft office 2000 pro
Zoeken
In
Applicatienaam
Cd-Nummer
590
Naam
Microsoft Office 2000 pro
Fabrikant
Microsoft
Ingevoerd door
Onbekend
Invoer datum
4-7-2003 12:15:00
Licentiecode
QVJTQ-R2JQV-DKDXF-QW6XG-DT23D
Medium
Cd-rom
Aantal
1
Taal
Onbekend
Demo
Opmerking
CD2
QMDTT-79HKD-PY8CH-FWTMF-3HY8F
Te vinden
Magazijn
Bij personeel
Extern
Vorige
Volgende
Nieuw
Wijzigen
Opslaan
Annuleren
Verwijderen
Record 1 van 2
Druk op F5 om te vernieuwen.

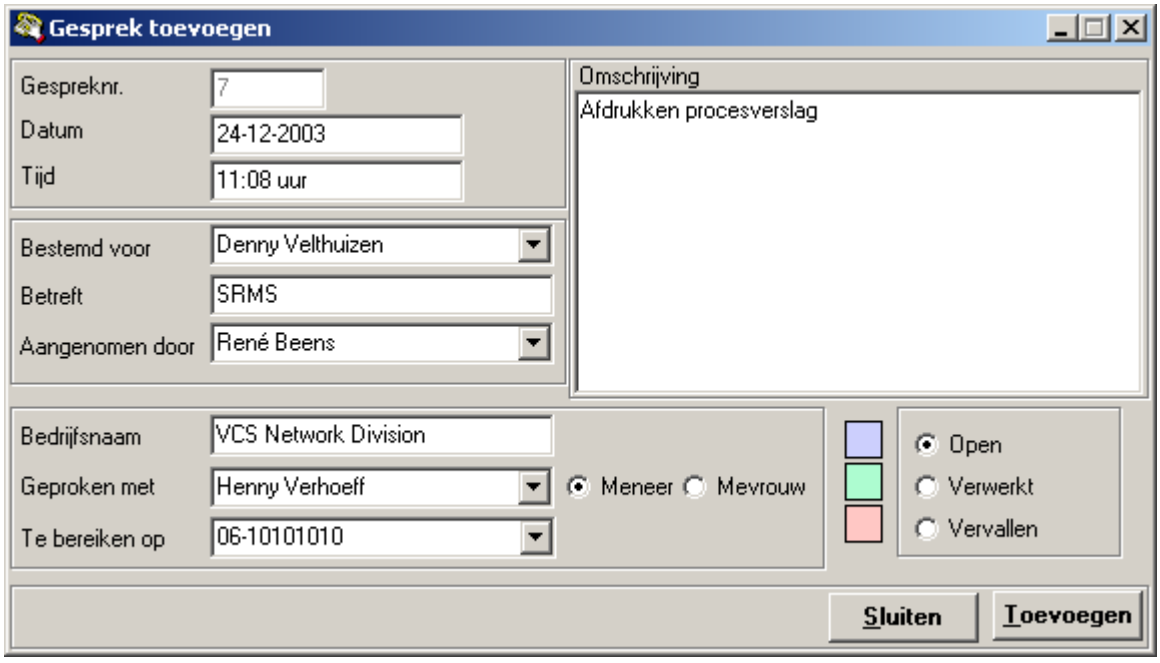

Overzicht alle storingen
Bedrijfsnaam
Alle
Uitvoerder
Denny Velthuizen
Status
Alle
Afdrukken
Voorbeeld
Nr
Datum
Bedrijf
Betreft
Uitvoerder
Status
7
05-12-2003
Bedrijfsnaam
Instaleren van pc
Denny Velthuizen
Actief
8
11-12-2003
Bedrijfsnaam
SRMS
Denny Velthuizen
Afgerond
9
23-12-2003
TestCompany
SRMS
Denny Velthuizen
Open
Bekijk
Verwijderen
Actief
Open
Afgerond
Vervallen


Nieuwe storing toevoegen
Storingnummer : 11
Uitvoerder
Denny Velthuizen
Betreft
SRMS
Datum
24-12-2003
Laatse contact
nu
Omschrijving
Maken Testverslag
Nog uit te voeren
Actief
Open
Afgerond
Vervallen
Sluiten
Toevoegen


Database Importeren
Bestandsnaam
SRMS-2002.dat
Database
denny
Sluiten
Importeren


SQLServer
SQL-Server
W2000srv
DataBase
denny
Windows login
SQL login
Loginnaam
sa
Wachtwoord
Sluiten
Opslaan


Telefoonafhandeling
Medewerker
Alle
Status
Alle
Afdrukken
Voorbeeld
Nr.
Datum
Tijd
Beller
Bestemd voor
Betreft
Status
0 11-12-2003 15:42 uur Contact 1 Denny Velthuizen Installatie XP Vervallen
1 11-12-2003 15:52 uur Meneer Contact11 Rene Beens bnvbncvb Verwerkt
2 11-12-2003 16:13 uur Contact 2 Denny Velthuizen Email probleem Verwerkt
3 11-12-2003 16:36 uur TxtCaller Test fsgf Verwerkt
5 22-12-2003 10:37 uur Meneer Koren Rene Beens XP Vervallen
6 23-12-2003 15:27 uur Meneer Koren René Beens terugbellen morgen Open
Bekijk
Nieuw
Verwijderen
Open
Verwerkt
Vervallen
Druk op F5 om te vernieuwen.


Gesprek toevoegen
Gesprekнr.
7
Datum
24-12-2003
Tijd
11:08 uur
Omschrijving
Afdrukken procesverslag
Bestemd voor
Denny Velthuizen
Betreft
SRMS
Aangenomen door
René Beens
Bedrijfsnaam
VCS Network Division
Geproken met
Henny Verhoeff
Meneer
Mevrouw
Te bereiken op
06-10101010
Open
Verwerkt
Vervallen
Sluiten
Toevoegen


Database Exporteren
Database
denny
Exporteren naar
Desktop
My Documents
My Computer
My Network Places
Recycle Bin
appzip
ecos
Trash
www2
Bestandnaam
SRMSBackup_24-12-2003.dat
Sluiten
Exporteren
C:\Documents and Settings\denny\Desktop\SRMSBackup_2


SRMS Instellingen
Algemeen
E-mail
Kleuren
Status actief
Status open
Status afgerond
Status vervallen
Sluiten
Opslaan

Plan van Aanpak

Service Repair Management System

VCS Network Division
4 september 2003

VCS Network Division
Dalweg 201 3762 AW Soest The Netherlands
Tel: 035-60 30 309 Fax: 035-60 14 325 internet: www.vcs-network.nl

Inhoud

1. Inleiding

Dit document bevat een plan van aanpak dat gebruikt zal worden bij het uitvoeren van het project Service Repair Management System (SRMS). Voor het lezen van dit document kan enige kennis van systeemontwikkeling en Unified Modeling Language (UML) een vereiste zijn.

2. Opdrachtomschrijving

Opdrachtgever:
VCS Network Division is een onderdeel van VCS Groep B.V.. VCS Groep B.V. kan worden onderverdeeld in drie divisies, namelijk VCS Telecom, VCS Network Division en Verhoeff Computers Supplies. De kernactiviteiten van VCS Telecom bestaan uit het ontwerpen, verkopen en installeren van telefooncentrales in combinatie met Call Center oplossingen. VCS Network Division houdt zich voornamelijk bezig met advies, aanleg en beheer van complete bedrijfsnetwerken. Naast het beheer van hard -en software zijn er ook projecten voor het ontwikkelen van software. In veel gevallen betreffen deze projecten aanvullingen op de software van Exact.

Probleemstelling
Bij VCS Network Division wordt dagelijks gebruik gemaakt van verschillende soorten gegevens. Deze gegevens dienen als ondersteuning van de werkzaamheden die bij VCS worden uitgevoerd. Deze gegevens kunnen worden verdeeld in een aantal delen, namelijk relatiegegevens, storinggegevens en installatiegegevens van klanten en (telefonische) gesprekgegevens. Voor het beheren van deze administratieve richtingen worden verschillende methoden en oplossingen gehanteerd. Deze oplossingen verschillen van een Access-applicatie tot Excel-sheets.

De voornaamste werkzaamheid bij VCS Network Division is het leveren van een technische service voor klanten. Om deze service goed te kunnen verlenen zijn alle gegevens van de betreffende klant erg belangrijk en vaak noodzakelijk. Omdat er bij de administratieve werkzaamheden geen gebruik gemaakt wordt van een georganiseerde gegevensopslag is het veel werk en soms onmogelijk om de benodigde en juiste gegevens van een klant op te vragen.

Doelstelling
Het doel van de afstudeeropdracht is het ontwikkelen van een geïntegreerde gegevensbeheeromgeving in Borland Delphi, die de werknemers van VCS op een efficiënte manier ondersteunt bij hun administratieve werkzaamheden. Het doel van de geïntegreerde omgeving is dat de werknemers van VCS Network Division

snel en gemakkelijk alle relevante gegevens kunnen opvragen voor een optimale service aan de klant en de dagelijkse werkzaamheden te structureren.

3. Projectorganisatie

Met de volgende organisatie zal het project worden doorlopen:

Denny Velthuizen	-	Ontwerper/programmeur
Dhr. Verhoeff	-	Opdrachtgever/begeleider
Dhr. Beens	-	Begeleider

De functies binnen de projectorganisatie zijn hieronder beschreven.

- De *ontwerper/programmeur* zorgt er voor dat de eisen van opdrachtgever worden onderzocht en indien mogelijk worden ontworpen en gerealiseerd
- De *opdrachtgever* legt de gewenste eisen voor aan de ontwerper en moet samen met deze ontwerper tot een overeenstemming komen. Tijdens het project zullen alle beslissingen van de ontwerper gekeurd moeten worden door de opdrachtgever
- De *begeleider* moet de ontwerper waar nodig ondersteuning bieden om het project tot een goed einde te laten verlopen

4. Afbakening opdracht

Tijdens het project zullen de volgende gegevens gebieden geautomatiseerd worden:

- Relatiebeheer
- Storingbeheer
- Intern softwarebeheer
- Telefoonafhandeling

Andere gegevensgebieden zullen er in eerste instantie niet uitgewerkt worden. Mocht het zijn dat het project sneller voorloopt dan gepland, kunnen er nog gegevensgebieden aan toegevoegd worden.

5. Methode en wijze van rapporteren

Het project zal doorlopen worden met de ontwikkelmethode Extreme Programming (XP). XP maakt gebruik van een serie practices (activiteiten en richtlijnen) die tijdens dit project zo compleet mogelijk worden uitgevoerd.

Plan van aanpak en definitiestudie
Om de overdracht naar externe partijen te bevorderen is er besloten om dit plan van aanpak en een definitiestudie te maken van het gehele project. Deze documenten vallen dus niet onder de voorschriften van XP en dienen dus alleen als ondersteuning.

Technische documentatie
Technische documentatie in de vorm van een verslag is eveneens geen practice van XP. Wederom is er besloten om in de definitiestudie en het procesverslag een aantal technische schema's op te nemen om het technische gedeelte van het project te ondersteunen. Omdat er gebruik gemaakt wordt van een OO-ontwikkelomgeving is er in overleg met de opdrachtgever besloten om UML (Unified Modeling Language) te gebruiken en ERD voor het databasemodel.

6. Uitgangspunten en randvoorwaarden

- Er moet regelmatig overleg zijn tussen de ontwerper en opdrachtgever.
- Er moet regelmatig overleg zijn tussen de ontwerper en begeleider.

- Er moet toegang zijn tot alle nodige gegevens binnen VCS Network Division om de gegevensgebieden zo goed mogelijk te kunnen automatiseren.
- De betrokken personen bij dit project moeten zich zo goed als mogelijk houden aan de voorschriften die zijn opgesteld voor het gebruik van XP.

7. Risicofactoren

Nr	Risico	Kans	Impact	Preventieve maatregelen
1	Het niet tijdig realiseren van het project	Middel	Middel	• Elke 2 weken planning aanpassen • Voortgangsoverleg • Aanpassen van systeemeisen • Project kan intern worden overgenomen
2	Het eindproduct kan niet voldoen aan de wensen van de opdrachtgever	Laag	Hoog	• Middels pair programming en voortgangsoverleg krijgt de opdrachtgever een duidelijk beeld van de gehele applicatie
3	Het eindproduct moet gewijzigd worden na afloop van het project	Middel	Laag	• Kennis van applicatie aanwezig bij werknemers (pair programming) • Code makkelijk onderhoudbaar (refactoring)

Omdat het project wordt uitgevoerd door één persoon is het grootste risico dat het systeem niet tijdig is gerealiseerd. Omdat de planning om de twee weken wordt bijgesteld kan er vroegtijdig gezien worden of de gestelde eisen tijdig kunnen worden gerealiseerd. Mocht dit niet het geval zijn kan er besloten worden een aantal systeemeisen te laten vervallen. Mochten de vervallen systeemeisen onmisbaar zijn kan er besloten worden om na het project intern het programma verder te ontwikkelen. Mede doordat het project overdraagbaar is, is het probleem als “middel” geclassificeerd.

8. Benodigde resources

Extreme Programming

Om het ontwikkeltraject met behulp van Extreme Programming goed te kunnen doorlopen is het belangrijk om over goede documentatie van deze methode te beschikken. Onder andere zullen de boeken gebruikt worden van de “XP Series”. Dit is een boekenreeks waarin alle aspecten van Extreme Programming in het engels worden toegelicht. In deze reeks o.a.: “Extreme Programming Explained” van Kent Beck 2001 en “Extreme Programming Installed” van Ron Jeffies, Ann Anderson en Chet Hendrikson 2000.

Support template

Voor het beheren van een XP project wordt gebruik gemaakt van een dynamische planning. Omdat het beheren van een dynamische planning een belangrijke taak is heb ik een support template gemaakt waarmee snel wijzigingen gemaakt kunnen worden. Deze template is gemaakt in Microsoft Excel.

Broncodes

Omdat ik zelf al redelijk vaak gebruik gemaakt heb van Delphi, beschik ik zelf over een groot aantal units waarin verschillende herbruikbare routines zitten. Deze sourcecodes kunnen dus van pas komen bij het ontwikkelen van dit project.

9. Standaards, richtlijnen en procedures

Omdat XP een praktisch ingestelde ontwikkelmethode is, gaat er veel aandacht uit naar de sourcecodes. Een belangrijke practice is dan ook het refactoren van de sourcecodes, wat inhoud dat de codes geoptimaliseerd worden aan hand van internationale standaarden. Bij deze internationale standaarden kan gedacht worden aan naamgeving van variabele en routines, routineopbouw, wordbreaks, maar ook zaken als uitlijning. Deze practice word gebruikt bij het ontwikkelen van de applicatie en het ontwikkelen van de tests. Door het hanteren van deze regels moeten de sourcecodes beter leesbaar en beter begrijpbaar worden.

10. Planning

De eerste practice van XP is het uitvoeren van de planninggame. In dit overleg wordt in detail een planning gemaakt van het gehele project. De uitwerking van deze detailplanning is verwerkt in de support template.

Definitiestudie

Service Repair Management System

VCS Network Division

8 september 2003

VCS Network Division
Dalweg 201 3762 AW Soest The Netherlands
Tel: 035-60 30 309 Fax: 035-60 14 325 internet: www.vcs-network.nl

Algemene informatie

Opdrachtgever	Henny Verhoeff
Auteur(s)	Denny Velthuizen
Datum	08-09-2003
Document	Definitiestudie Service Repair Management System

Inhoudsopgave

1 Inleiding

Deze definitiestudie is geschreven ten behoeven van de overdraagbaarheid van het project aan derden. In deze definitiestudie worden de doelen en eisen van het systeem uitgewerkt. De definitiestudie kan worden gezien als een uitwerking van de planninggame die in het begin van dit project is uitgevoerd.

De doelstellingen van de definitiestudie zijn als volgt:

- Opstellen systeemeisen
- Opstellen systeemconcept
- Definiëren van ontwikkelscenario
- Beschouwen van technische structuur
- Releaseplanning

2 Systeemeisen

2.1 Functionele systeemeisen

Omdat het project valt op te delen in vier gegevensgebieden, relatiebeheer, storingbeheer, internsoftwarebeheer en telefoonbeheer, zal ik de systeemeisen per gegevensgebied uitwerken.

Alle gegevensgebieden

Tijdens het ontwikkelen van het gehele project moet er gebruik gemaakt worden van één gestandaardiseerde interfaceopbouw. Deze interface moet dusdanig worden opgebouwd dat het bestand is tegen roekeloos gebruik.

Relatiebeheer

Met relatiebeheer is het mogelijk om alle relaties van VCS Network Division te beheren. De gebruiker moet op een duidelijke en gemakkelijke manier alle gewenste handelingen kunnen verrichten. De volgende eisen kunnen worden verwacht van dit systeemdeel:

- Geavanceerd zoeken
- Beheren van relatiegegevens (toevoegen, verwijderen, wijzigen)
- Beheren van configuratie -en installatiegegevens
- Weergeven en afdrukken van zoekopdrachten

Storingbeheer

Met storingbeheer is het mogelijk om alle storingen te beheren die bij VCS Network Division voorkomen. Een koppeling van storing naar relatie is noodzakelijk. Het storingbeheer kent de volgende systeemeisen:

- Beheren van storingen (toevoegen, verwijderen, wijzigen)
- Geavanceerde weergave van de storingen
- Specificeren en afdrukken van storingoverzicht

Intern softwarebeheer

Met softwarebeheer kan de internsoftware-databank worden beheerd. Systeemeisen voor dit systeemdeel:

- Geavanceerd zoeken
- Beheren van interne software (toevoegen, verwijderen, wijzigen)

Telefoonbeheer

Met telefoonbeheer is het mogelijk om alle storingen te beheren die bij VCS Network Division voorkomen. Een koppeling van gesprek naar relatie is niet noodzakelijk. Het telefoonbeheer kent de volgende systeemeisen:

- Beheren van gesprekken (toevoegen, verwijderen, wijzigen)
- Geavanceerde weergave van de gesprekken
- Specificeren en afdrukken van gesprekoverzicht

2.2 Niet functionele systeemeisen

1. Het systeem moet kunnen draaien op elk Microsoft Windows systeem
2. Het systeem dient toegang te hebben tot een Microsoft SQL DBMS
3. Het systeem dient vanaf een netwerk met meerdere computers tegelijk toegankelijk te zijn.
4. Het systeem moet worden geïmplementeerd in het bestaande bedrijfsnetwerk van VCS Network Division

3 Systeemconcept

In dit hoofdstuk wordt het systeemconcept uitgewerkt zoals deze is opgesteld in de planninggame. Het systeemconcept is weergegeven met behulp van use-case beschrijvingen en een use-case-diagrammen. (UML) In het concept wordt gebruik gemaakt van één actor (alle gebruikers). Omdat er geen gebruik gemaakt wordt van geautoriseerde toegang tot de applicatie is het niet nodig om met verschillende actoren te werken.

3.1 Relatiebeheer

Use-case-beschrijvingen - Relatiebeheer

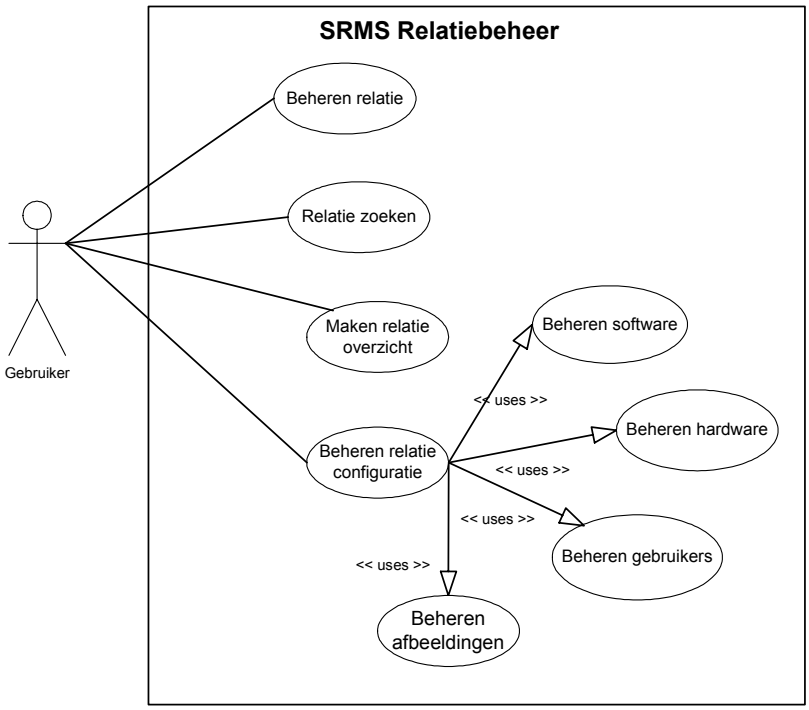
Usecase - Relatiebeheer	
Naam	Beheren relatie
Samenvatting	De gebruiker ziet een scherm waarmee hij alle standaardhandelingen kan verrichten zoals toevoegen, verwijderen, en wijzigen.
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	De gebruiker start het relatiebeheerscherm. Om een gebruiker te kunnen wijzigen of verwijderen moet eerst de betreffende relatie gezocht worden. Als er een relatie verwijderd wordt zullen ook de bijbehorende installatiegegevens verwijderd worden.
Resultaten	De gebruiker heeft de mogelijkheid om alle gewenste handelingen van de relatie uit te voeren.

Usecase - Relatiebeheer	
Naam	Zoeken relatie
Samenvatting	De gebruiker ziet een scherm waarmee relaties gezocht kunnen worden
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	De gebruiker start het relatiebeheerscherm. Hiermee kan er gezocht worden op verschillende categorieën zoals naam, contactpersoon, adres en telefoonnummer. Alle entries van de database zullen gevonden worden wanneer een deel ervan overeenkomt met de zoekstring. Met navigatieknoppen (vorige/volgende) kan er in het zoekresultaat genavigeerd worden. Om alle relaties te vinden kan er gezocht worden op “%” (MS SQL prefix).
Resultaten	De gewenste relatie of relatiegroep is gevonden.

Usecase - Relatiebeheer	
Naam	Maken overzicht
Samenvatting	De gebruiker kan een uitdraai (overzicht) maken van de gevonden relatie of relatiegroep.
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server, er is minimaal 1 entry in het zoekresultaat.
Beschrijving	Als de gebruiker een zoekopdracht heeft gedaan, is het mogelijk deze weer te geven en af te drukken.
Resultaten	Een lijst met daarop de gewenste gegevens van een relatie of relatiegroep.

Usecase - Relatiebeheer	
Naam	Beheren installatie en configuratie
Samenvatting	De gebruiker kan per relatie alle gegevens beheren betreffende de installatie en configuratie van een systeem of netwerk.
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server, er is minimaal 1 entry in het zoekresultaat.
Beschrijving	Als de gebruiker een relatie gezocht heeft, is het mogelijk om de installatie van deze relatie te beheren. De gegevensgebieden zijn: beheren software, beheren hardware, beheren gebruikers en beheren configuratieschema's. Beheren omvat bij alle gegevensgebieden weergeven, toevoegen, verwijderen en wijzigen.
Resultaten	De configuratiegegevens kunnen naar wens worden beheerd en ingezien.

Use-case-diagram - Relatiebeheer



3.2 Storingbeheer

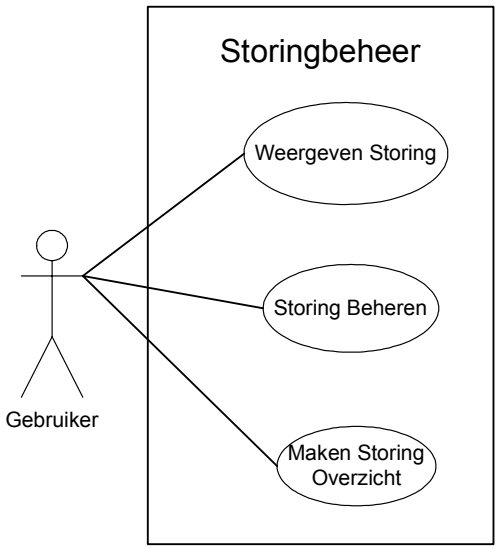
Use-case-beschrijvingen - Storingbeheer

Usecase - Storingbeheer	
Naam	Storingbeheren
Samenvatting	De gebruiker kan storingen beheren voor elke relatie in de database.
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	Onder het beheren van een storing wordt verstaan: toevoegen, verwijderen, wijzigen. Het toevoegen van een storing gaat vanuit het relatiebeheerscherm waar eerst een relatie in het zoekresultaat aanwezig moet zijn.
Resultaten	De gebruiker kan alle storingen voor alle relaties beheren.

Usecase - Storingbeheer	
Naam	Weergeven storing
Samenvatting	De gebruiker kan elke storing bekijken
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server, er is minimaal 1 storing aanwezig om in te kijken
Beschrijving	Het storingsscherm geeft een overzicht van alle aanwezige storingen. De lijst laat alleen de onderscheidende informatie van een storing zien. Elke storing kan worden ingezien zodat alle gegevens van de storing zichtbaar zijn.
Resultaten	Het inzien van elke gewenste storing.

Usecase - Storingbeheer	
Naam	Maken storingoverzicht
Samenvatting	De gebruiker kan de storinglijst specificeren en afdrukken
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	In het storingscherm staan alle aanwezige storingen. Deze lijst kan worden gespecificeerd aan de hand van “uitvoerder”, “bedrijfsnaam” en status. Ook is het mogelijk om deze gespecificeerde lijst af te drukken.
Resultaten	Het maken en specificeren van de storinglijst.

Use-case-diagram - Storingbeheer



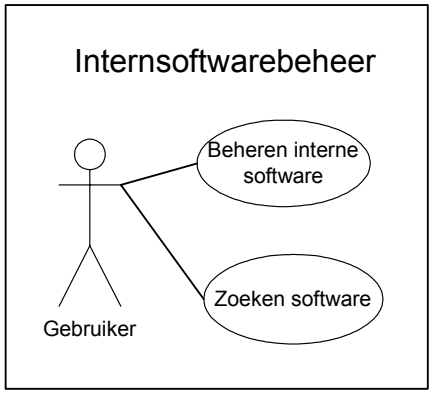
3.3 Internsoftwarebeheer

Use-case-beschrijvingen - Intern softwarebeheer

Usecase - Internsoftwarebeheer	
Naam	Zoeken interne software
Samenvatting	De gebruiker kan software in de database zoeken
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	In het softwarebeheerscherm is het mogelijk om een zoekactie op de database los te laten. Er kan gezocht worden op verschillende categorieën zoals softwarenaam, producent en taal.
Resultaten	Er kan een geavanceerde zoekopdracht worden gegeven op de database.

Usecase - Internsoftwarebeheer	
Naam	Beheren interne software
Samenvatting	De gebruiker kan de interne software databank beheren
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	In het softwarebeheerscherm is het mogelijk om alle gewenste handelingen te verrichten. Zo is het mogelijk software toe te voegen, te verwijderen en te wijzigen. Voor verwijderen en wijzigen dient eerst een entry in het zoek resultaat aanwezig te zijn.
Resultaten	Het beheren van de software databank

Use-case-diagram – Intern softwarebeheer



3.4 Telefoonbeheer

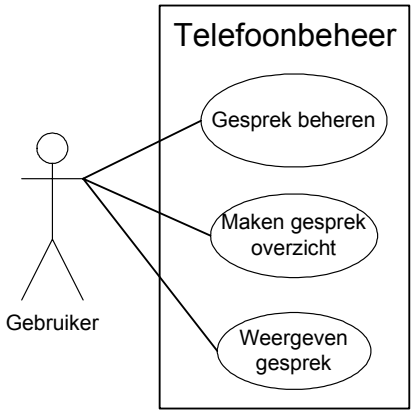
Use-case-beschrijvingen - Telefoonbeheer

Usecase – Telefoonbeheer	
Naam	Gesprek beheren
Samenvatting	De gebruiker kan een gesprek beheren
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	Met het scherm telefoonbeheer kan een gesprek worden toegevoegd, gewijzigd en verwijderd. Ook kan een gesprek gekoppeld worden aan een relatie of leverancier.
Resultaten	Het beheren van alle telefoongesprekken.

Usecase – Telefoonbeheer	
Naam	Weergeven gesprek
Samenvatting	De gebruiker kan elk gesprek bekijken
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server, er is minimaal 1 gesprek aanwezig om in te kijken
Beschrijving	Het storingsscherm geeft een overzicht van alle gevoerde gesprekken. De lijst laat alleen de onderscheidende informatie van een gesprek zien. Elk gesprek kan worden ingekeken zodat alle informatie van het gesprek te zien is.
Resultaten	Het inzien van elk gewenst gesprek.

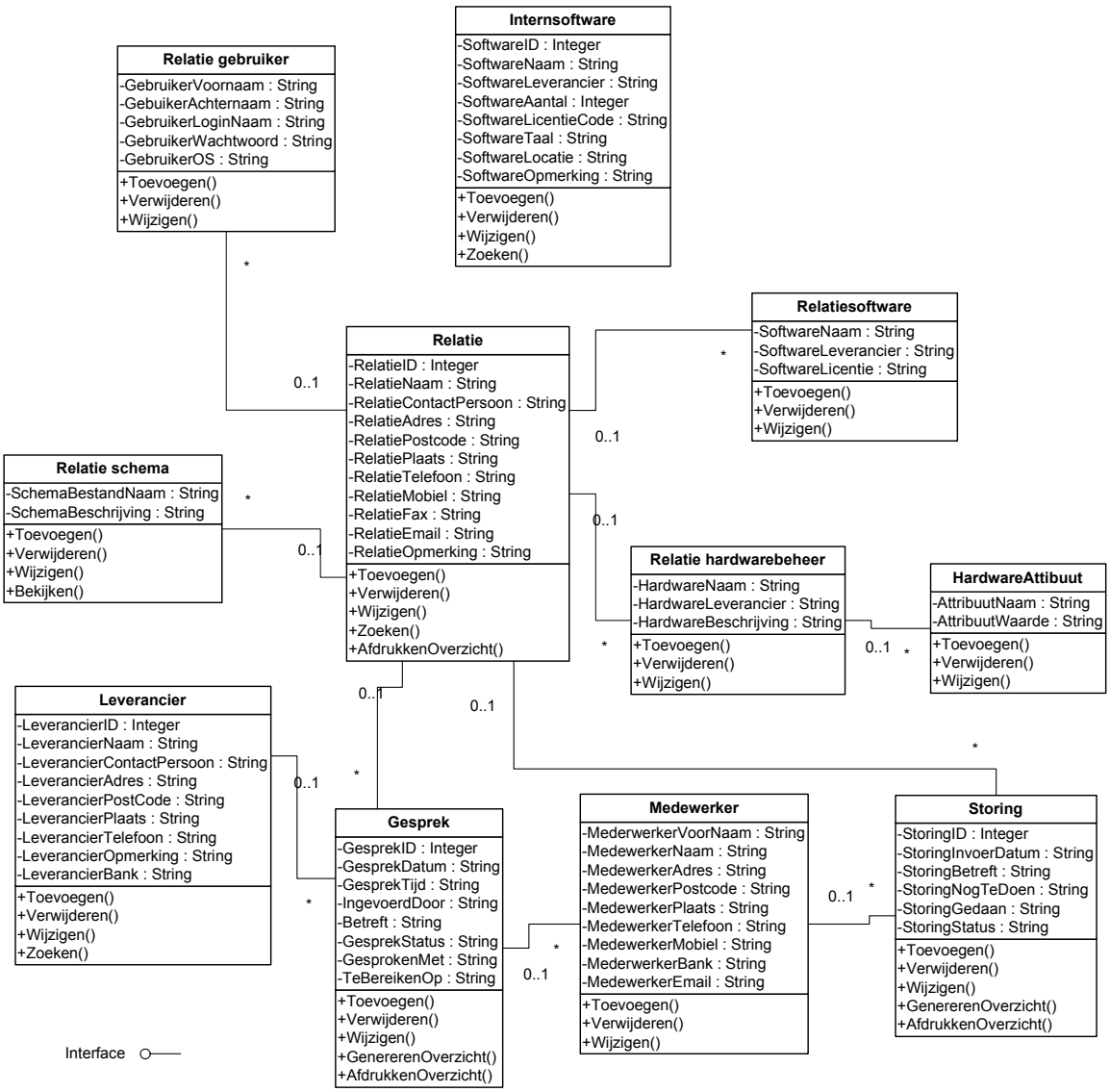
Usecase – Telefoonbeheer	
Naam	Maken gesprekoverzicht
Samenvatting	De gebruiker kan de gespreklijst specificeren en afdrukken
Actoren	Alle gebruikers
Aannamen	Er is verbinding met de SQL Server
Beschrijving	In het gesprekscherm staan alle aanwezige gesprekken. Deze lijst kan worden gespecificeerd aan de hand van medewerker en status. Ook is het mogelijk om deze gespecificeerde lijst af te drukken.
Resultaten	Het maken en specificeren van een gespreklijst.

Use-case-diagram - Telefoonbeheer



3.5 Overall klassendiagram

Klassendiagram gehele project



3.6 Technische structuur

Het programma zal gaan werken vanaf het bedrijfsnetwerk van VCS Network Division. Het onderhoud zal hierdoor maar op één plek plaats vindt. Omdat de applicatie na de eerste release al bruikbaar is, zullen er veel tussentijdse updates plaatsvinden. Het beheren van de applicatie op één plek is erg interessant om de werknemers te laten werken met één executable die vanaf het intranetwerk gestart zal worden. Het vervangen van deze executabel zal resulteren in het feit dat alle werknemers automatisch gebruik maken van de laatste versie van het programma.

De ontwikkeling van het project zal plaats vinden met behulp van Borland Delphi en Microsoft SQL Server.

4 Releaseplan

Tijdens de planninggame is de volgende definitieve release verdeling naar voren gekomen:

- Release 1: Relatiebeheer (incl. installatie- en configuratiebeheer)
- Release 2: Storingbeheren (incl. beheer van interne medewerkers)
- Release 3: Intern softwarebeheer
- Release 4: Telefoonafhandeling (incl. leverancierbeheer)

5 Planning

Omdat XP gebruik maakt van een dynamische planning kunt u voor de huidige stand van zaken van deze planning terecht bij de support-template die op het bedrijfsnetwerk van VCS Network Division staat.