Departement HWBKG
Toegepaste Informatica
Optie: Systeem- en netwerkbeheer

Ontwikkeling van WebParts voor een Microsoft SharePoint omgeving

CAMPUS
Geel

Denis Caeyers

Academiejaar 2007-2008

VOORWOORD

Dankzij mijn stageperiode op het SCK•CEN ben ik hard gegroeid. Mijn stageopdracht heeft mij de kans gegeven om mij te verdiepen in het programmeren. Daarnaast heb ik ook ruime kennis kunnen maken met het bedrijfsleven, wat zeker even belangrijk is.
Daarom wil ik graag de mensen van het SCK•CEN bedanken om mij deze kans te geven. Als eerste mijn stagebegeleider, Dhr Philippe Gouat. Ondanks zijn druk schema stond hij steeds klaar om mij helpen, vragen te beantwoorden en meer duidelijkheid te geven.
Als tweede wil ik graag de medewerkers van Infoplan bedanken. Bij problemen met het coderen kon ik steeds bij hun terecht. Zij hebben mij dan ook met raad en daad geholpen.
Naast de mensen van SCK•CEN wil ik ook de KHK bedanken om mij deze kans te geven. Meer bijzonder, Koen Vangeel, mijn eindwerk begeleider die steeds klaar stond om mijn teksten te verbeteren, mijn periode op te volgen en zelfs presentaties te organiseren ter voorbereiding van de eindwerkverdediging.
Als laatste wil ik ook mijn moeder bedanken. Dankzij haar heb ik kunnen verder studeren en ze heeft mij dan ook steeds gesteund en aangemoedigd.
Naast het opdoen van technische kennis, ben ik ook als mens gegroeid. De laatste stap van de opleiding Toegepast Informatica is het eindwerk. Dankzij dit eindwerk heb ik een kijkje kunnen nemen wat er mij in de toekomst op professioneel gebied te wachten staat.

SAMENVATTING

Tijdens mijn stage periode heb ik gewerkt aan Web Parts voor een Microsoft SharePoint omgeving. Het SCK•CEN gaat in de nabije toekomst z'n intranet vernieuwen en heeft hiervoor gekozen om een Microsoft SharePoint oplossing te implementeren.
Om de medewerkers van de expertise groep "Reactor Technology Design" te ondersteunen, heb ik Web Parts ontwikkeld die Access Databases en data acquisitie bestanden uitlezen en beschikbaar maken op het nieuwe intranet.
Ik ben de eerste die op het SCK•CEN zelf Web Parts schrijft. Mijn stageopdracht kan dus gebruikt worden als voorbeeld te dienen om in de toekomst ook zelf Web Parts te schrijven voor eventueel andere departementen.
De Web Parts worden in VB.NET geschreven met behulp van een Web Custom Control.
Om gegevens op te halen wordt er gelezen uit:

- Microsoft Access Databases,
- Microsoft SQL Server Databases,
- Binaire bestanden.

INHOUDSTAFEL

VOORWOORD **2**

SAMENVATTING **4**

INHOUDSTAFEL **5**

INLEIDING **7**

1 HET STAGEBEDRIJF **8**

1.1 Algemeen **8**
1.2 Organisatieprofiel **9**
1.2.1 De belangrijkste onderzoeksprojecten zijn: 9
1.2.2 De belangrijkste installaties zijn: 10
1.3 Missie **11**
1.4 Organigram **11**
1.5 Rekenbureau **12**
1.5.1 Algemeen 12
1.5.2 Organigram 13
1.5.3 Taken 13
1.6 Infoplan **13**
1.6.1 Algemeen 13
1.6.2 Organigram 14
1.6.3 Taakomschrijving 14

2 DE STAGEOPDRACHT **15**

3 GEBRUIKTE TECHNOLOGIEËN **16**

3.1 Microsoft SharePoint **16**
3.1.1 Algemeen 16
3.1.2 Microsoft Office Sharepoint Server 2007 17
3.1.3 Versies 18
3.2 .NET Framework **19**
3.3 BATCH bestanden **19**
3.4 Web Custom Control **19**
3.5 UML **19**
3.5.1 Use Case diagram 20
3.5.2 Activity diagram 20
3.6 SQL **20**
3.7 Webservices **20**
3.8 XML **21**

4 ANALYSE **22**

4.1 Globale analyses **22**
4.1.1 Componenten analyse 22
4.1.2 C# vs VB.NET 23
4.1.3 Web Parts 25
4.2 CO Calculation Notes – CO Technical Notes **25**
4.2.1 Use Case diagrammen 26
4.2.2 Activity diagram 27
4.2.3 Databases 28

4.2.4 Mappenstructuur 30
4.3 DEO Mechanical Drawings – DEO Instrumentation Drawings – BR2 Drawings 30
4.3.1 Use Case diagrammen 30
4.3.2 Activity Diagrams 32
4.3.3 Databases 34
4.3.4 Samenhang tussen verschillende bestanden 35
4.4 Bidasse 36
4.4.1 Use Case Diagrammen 36
4.4.2 Bestanden 38

5 CODE 40

5.1 Gemeenschappelijke Code 40
5.1.1 Klassen 40
5.1.2 Procedures 43
5.1.3 Snippets 46
5.2 CO Calculation Notes – CO Technical Notes 49
5.2.1 Property's 49
5.2.2 ToolParts 51
5.2.3 CreateChildControls 55
5.3 DEO Mechanical Drawings – DEO Instrumentation Drawings 58
5.3.1 Property's 58
5.3.2 ToolParts 60
5.3.3 CreateChildControls 61
5.4 BR2 Drawings 67
5.4.1 Property's 67
5.4.2 ToolParts 68
5.4.3 CreateChildControls 71
5.5 Bidasse 71
5.5.1 Property's 71
5.5.2 ToolParts 72
5.5.3 CreateChildControls 76

6 DEPLOYMENT 81

6.1 Benodigdheden 81
6.2 Global Assembly Cache 81
6.3 Cabinet File 81
6.4 stsadm.exe 82
6.5 Deployment Scripts 82
6.5.1 Install.bat 82
6.5.2 WPList.bat 83
6.5.3 Uninstall.bat 84
6.6 Stappenplan 84

BESLUIT 88

BIBLIOGRAFIE 89

BIJLAGE1: COMPETENTIES 90

BIJLAGE 2: SCREENSHOTS 98

INLEIDING

Dankzij het nieuwe intranet zal elk team en project z'n eigen Microsoft SharePoint website krijgen. Via mijn Web Parts kan men relevante documenten of informatie beschikbaar stellen aan de bezoekers van deze website.
In dit eindwerk leg ik stap voor stap uit hoe ik ervoor gezorgd heb om zelf Web Parts te schrijven en deze te implementeren op de SharePoint servers.
Het volgende wordt besproken in dit eindwerk:

- Als eerste vind je meer informatie over mijn stagebedrijf, het SCK•CEN.
- Daarna volgt een korte beschrijving van de stageopdracht .
- Hierna beschrijf ik de technologieën die ik gebruikt heb.
- Dan kan je de analyses mee volgen.
- Hierop volgt de uiteindelijke programmacode.
- Als laatste zie je stap voor stap hoe je de Web Parts op de SharePoint server krijgt.

1 HET STAGEBEDRIJF

Vooraleer we het project gaan bekijken wil ik eerst een beeld schetsen van het bedrijf waar ik 13 weken mijn stage heb afgelegd.Het SCK•CEN is op het eerste zicht niet meer dan wat initialen. Als we daarentegen de volledige naam bekijken "StudieCentrum voor Kernenergie, Centre d'Etude de l'énergie Nucléaire" krijgen we al meteen een beeld waarom elke dag 600 mensen zich inzetten voor dit bedrijf.

Allereerst zal ik een algemeen overzicht geven van het SCK (paragraaf 1.1). In het organisatieprofiel (paragraaf 1.2) worden de belangrijkste onderzoeksprojecten en de belangrijkste installaties kort uitgelegd. Daarna zal ik de missie van het bedrijf weergeven (paragraaf 1.3). Een missie is datgene waar een onderneming voor wil staan en naar buiten wil brengen.
Mijn plaats op het SCK•CEN was op het Rekenbureau (Calculation Office - CO). Dit bureau is een onderdeel van de dienst "Ontwerp Reactor Technologie".Wat op zijn beurt onderdeel is van het "Advanced Nuclear Systems Institute". Omdat er pas een volledige reorganisatie heeft plaatsgevonden geef ik het volledig organigram weer in paragraaf x.4. De rol van het "Calculation Office" leg ik uit in paragraaf 1.5.
Omdat de informatica dienst op het SCK•CEN van grote hulp is geweest tijdens mijn stageperiode, situeer en licht ik hen kort toe in paragraaf 1.6.

Figuur 1.1.1: Luchtfoto SCK•CEN

1.1 Algemeen

Het SCK•CEN is een Stichting van Openbaar Nut opgericht in 1952. De site bevindt zich te Mol, telt ongeveer 20 gebouwen en heeft zijn eigen sportclub, voetbalveld, tennisterreinen en appartementen voor werknemers.
De oorspronkelijke oprichters hadden als doel om door middel van het SCK•CEN een weg te vinden naar de nucleaire industrie voor België.

Onderzoek is de kern van dit bedrijf. Onderzoeken kunnen verschillende vormen aannemen, zoals studies naar technische projecten, medische toepassingen en waarnemingen wat de invloed van nucleaire energie op de geografische plaatsen zijn.

Onder de technische projecten staan vooral de kernreactoren centraal. Bestralingen van verschillende materialen, wat te doen met nucleair afval, onderhouden van kernreactoren maken hier een onderdeel van. Niet alleen materialen kunnen bestraald worden, ook levende wezens krijgen te maken met nucleaire bestralingen of komen in contact met radioactieve stoffen. Denk maar aan lichte nucleaire toedieningen om de werking van organen te bestuderen. Onderzoeken naar invloeden op de omgeving worden op het SCK•CEN ook gemaakt. Zo werd er een studie gedaan naar het aantal kankersterftes in de omgeving Mol-Dessel-Geel-Balen.

1.2 Organisatieprofiel

Het Studiecentrum voor Kernenergie SCK•CEN werd opgericht in 1952 teneinde de Belgische academische en industriële wereld toegang te verschaffen tot de wereldwijde ontwikkeling van kernenergie.
Het SCK•CEN is een Stichting van Openbaar Nut, met een privaatrechtelijk statuut, onder voogdij van de Belgische federale minister van Energie.
Het studiecentrum telt ongeveer 600 werknemers waarvan een derde houder is van een universitair diploma.

Sinds 1991 geeft de statutaire opdracht voorrang aan onderzoek over onderwerpen met maatschappelijke relevantie zoals:

- Veiligheid van kerninstallaties;
- Stralingsbescherming;
- Veilige behandeling en berging van radioactief afval;
- Strijd tegen ongecontroleerde proliferatie van splijtbaar materiaal;
- Strijd tegen terrorisme.

Tevens zal het Centrum de nodige kennis ontwikkelen, verzamelen en verspreiden via vorming en communicatie, en alle diensten verlenen die gevraagd worden in het nucleaire domein (door medische sector, nucleaire industrie en overheid). Verder dient het Centrum de nodige pluridisciplinaire wetenschappelijke contacten te leggen betreffende de energieproblematiek.
Deze acties dragen bij tot het globale doel: een excellentiecentrum in stand houden rond vreedzame toepassingen van kernenergie.
Deze beschikbare kennis en infrastructuur worden ook gebruikt voor dienstverlening aan de industrie en voor opleidingen.

1.2.1 De belangrijkste onderzoeksprojecten zijn:

Veiligheid van reactoren en splijtstoffen

- Optimalisering van de configuratie van de reactorkern;
- Het gedrag van MOX-splijtstof en van splijtstof met hoge versplijtingsgraad;
- De studie van de verbrossing van drukvaten en van scheurtjes in inwendige delen van de reactor veroorzaakt door stralingsgeïnduceerde corrosie;
- De invloed van bestraling op instrumentatie en structuurmaterialen voor kernfusiereactoren.

Radioactief afval

- De haalbaarheid en veiligheid van de berging van hoogradioactief afval en van gebruikte splijtstof in geologische kleilagen;
- De studie, ontwikkeling en evaluatie van declasseringstechnieken en -procédés voor kerninstallaties, inclusief decontaminatieprocédés;
- De studie en ontwikkeling van alternatieve afvalverwerking en volumereductietechnieken en -procédés.

Stralingsbescherming

- De biologische gevolgen van straling op levende organismen en het milieu;
- De wetenschappelijke ondersteuning van noodplanning en reacties;
- De Belgische bijdrage aan het non-proliferatieprogramma van het IAEA in Wenen;
- De wetenschappelijke steun bij het bepalen van de impact op het milieu en de sanering van besmette sites;
- Nucleaire metrologie;
- Optimalisering van de blootstellingen in de nucleaire industrie volgens de ALARA-principes;
- Optimalisering van medische blootstelling.

Opleidingen

- Post-graduaat diploma van nucleair ingenieur in samenwerking met zes Belgische universiteiten. Bovendien coördineert het SCK•CEN het European Network for Nuclear Education (ENEN) dat gesponsord wordt door de Europese Commissie.
- Andere specifieke vormingen en opleidingen (bijv. aangaande noodplanning, ontmanteling, stralingsbescherming, enz.)

Nieuwe en sociale thema's

- Het SCK•CEN integreert in zijn nucleair programma de maatschappelijke aspecten (PISA - Programme for Integration of Social Aspects) van kernenergie op vlak van duurzame ontwikkeling, ethiek, beheer van nucleair afval, risicoperceptie, communicatie, veiligheidscultuur en nucleaire wetgeving in diverse nucleaire ontwikkelingen,...
- Er wordt onderzoek gedaan op het gebied van nieuwe radio-isotopen en van de optimalisering van diagnose- en interventietechnieken.
- Voor het Europese Ruimtevaartagentschap (ESA - European Space Agency), voert het SCK•CEN onderzoek uit over de gevolgen van de blootstelling van installaties en levende organismen aan kosmische straling.

1.2.2 De belangrijkste installaties zijn:

BR2

BR2 is één van de krachtigste onderzoeksreactoren ter wereld. Hij wordt gebruikt voor splijtstof- en materiaaltests voor diverse reactortypes en voor het Europese fusieprogramma. Hij is ook het belangrijkste onderdeel in de productie van radio-isotopen voor medische en industriële toepassingen en voor siliciumdopering bestemd voor de elektronica-industrie.

BR1

BR1 is een luchtgekoelde en grafietgemodereerde reactor met een vermogen van 4MWth. Hij wordt veelvuldig gebruikt als neutronenbron voor activeringsanalyses, dosimetrische ijking, neutronenradiografie en referentie reactorexperimenten.

VENUS

De nulenergie-installatie VENUS maakt een gedetailleerde analyse van kernconfiguraties mogelijk, inclusief MOX en splijtstoffen met hoge versplijtingsgraad. Hij wordt intensief gebruikt voor de validatie van de kernconfiguratie van reactoren en kritische codes.

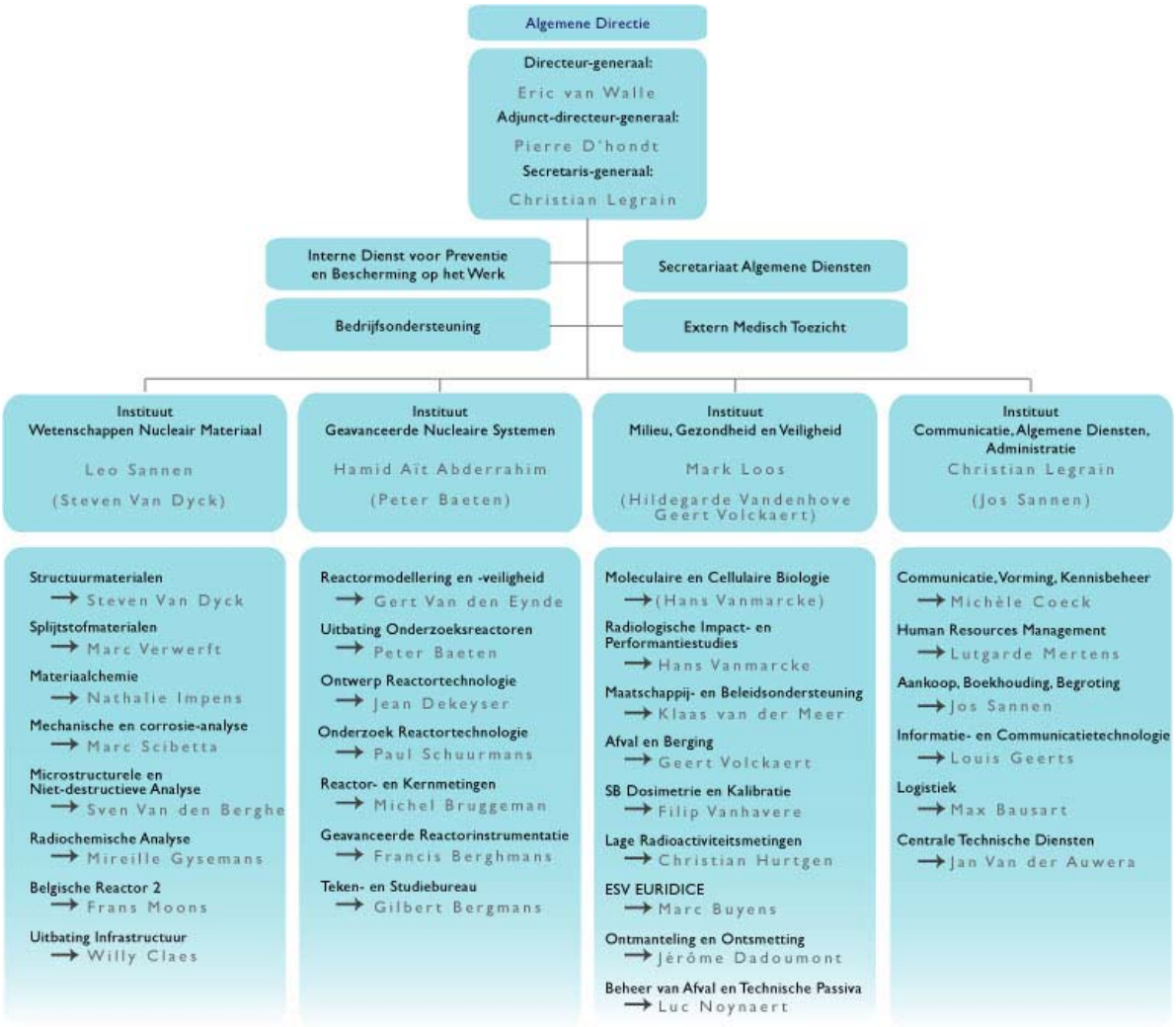
BR3
BR3 was een prototype van drukwaterreactoren (PWR's). Hij werd uitgekozen als Europees pilootproject voor de optimalisering van ontmantelings- en decontaminatietechnieken en -procédés, voor een realistische kostenraming en om nieuwe volumereductietechnieken voor secundair afval en minimalisering van stralingsdosissen voor het personeel te ontwikkelen.
HADES
Het HADES-laboratorium bevindt zich 225 m onder de grond en dient voor de studie van kleilagen als potentiële geologische opslagplaats voor langlevend hoogradioactief afval. Het wordt uitgebaat door EURIDICE, het economisch samenwerkingsverband tussen NIRAS/ONDRAF (de Belgische Instelling voor Radioactief Afval en verrijkte Splijtstoffen) en het SCK•CEN. Dit ondergronds laboratorium werd nog maar pas uitgebreid om op grote schaal tests uit te voeren over de haalbaarheid en de veiligheid van de berging van warmteontwikkelend kernafval.
LHMA
Het Laboratorium voor Hoge en Middelmatige Activiteit evalueert de gevolgen van bestraling op materialen die gebruikt worden in de huidige en toekomstige nucleaire installaties. Een brede waaier van mechanische, fysico-chemische en microstructuur onderzoeksinstrumenten zijn beschikbaar binnen en buiten de afstandsbediende warme cellen. Het laboratorium is betrokken bij toegepast onderzoek en basisonderzoek met ondersteuning van mathematische modellen voor het verifiëren en voorspellen van het gedrag van de nucleaire materialen tijdens hun bedrijfsduur.
Nucleaire analyse en chemische laboratoria
Het SCK•CEN meet en evalueert de interne contaminatie van werknemers en bedieners van nucleaire installaties en de contaminatie van de bodem en de voedselketen.
De laboratoria ondersteunen ook de onderzoeksreactoren en andere labo's voor destructief en niet-destructief onderzoek van hoogradioactief materiaal. Deze laboratoria ondersteunen ook het nucleaire noodplan waartoe het SCK•CEN een belangrijke bijdrage levert voor het Belgische en Europese beleid.
MYRRHA (momenteel: ontwerpstudie)
MYRRHA is een Accelerator Driven Subcritical System (ADS). Het bestaat uit een protonversneller met een protonenbundel van 350 MeV*5 mA naar een vloeibaar Pb-Bi afsplitsingsdoel in een Pb-Bi gekoelde subkritische snelle kern. MYRRHA zal als basis dienen voor het Europese experimentele ADS. Het zal protonen en neutronen leveren voor diverse O&O-toepassingen zoals bijvoorbeeld transmutatiestudies.

1.3 Missie

Het SCK•CEN draagt in een perspectief van duurzame ontwikkeling door onderzoek en ontwikkeling, opleiding, communicatie en diensten bij tot:

- nucleaire veiligheid en stralingsbescherming;
- medische en industriële toepassingen van de stralingen;
- het einde van de splijtstofcyclus.

1.4 Organigram



Figuur 1.4.1: Organigram SCK•CEN

1.5 Rekenbureau

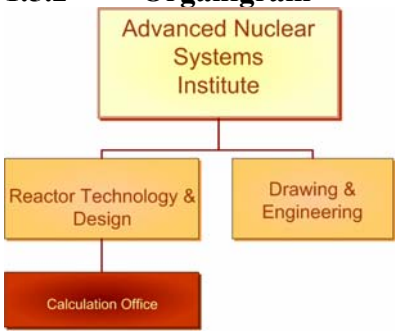
De dienst waar ik mijn 13 weken moest doorbrengen was het rekenbureau. Hier leg ik kort uit wat het rekenbureau is wat de belangrijkste taken zijn.

1.5.1 Algemeen

Voor de reorganisatie in 2006 maakt het Calculation Office (CO) deel uit van de divisie BR2. Het werd toen samengeplaatst met het Tekenbureau. Sinds 2006 is het CO een onderdeel van de "Reactor Technology & Design" expertise groep wat op zijn beurt een deel is van het "Advanced Nuclear System Institute". De projectingenieurs ontwerpen, engineeren en testen bestralingsdispostieven die gepland zijn voor de onderzoeksreactoren. Sinds de reorganisatie, zijn er ook nieuwe werknemers bijgekomen. Zij ontwerpen een volledig nieuwe reactor die binnen een aantal jaar gebouwd zal worden.
De missie van het rekenbureau is het ondersteunen van de projectingenieurs die deel uitmaken van de"Reactor Technology Design Expertise Group".
Om te slagen in deze missie beschikt het rekenbureau over de laatste nieuwe formules, procedures, hard- en software om zowel lineaire als niet-lineaire analyses te maken en veel technische informatie.
Het CO bestaat al sinds het oprichten van het SCK•CEN en maakte toen deel uit van het departement "Technology". In die tijd werkten 5 personen samen om alle berekeningen handmatig uit te voeren. Dankzij verschillende technologieën kwam er de introductie van de

rekenmachines en HP Workstation. Ondertussen zijn ze zover geëvolueerd dat de taken van deze 5 personen worden gedaan door 1 persoon die werkt met 2 DELL Workstations,namelijk, Philippe Gouat, mijn stagebegeleider.

1.5.2 Organigram



Figuur: 1.5.2.1: Organigram Rekenbureau

1.5.3 Taken

Zoals eerder gezegd bestaat het rekenbureau om projectingenieurs te ondersteunen in hun ontwerpactiviteiten. Meer concreet houdt dit in:

- beheren van de technische bibliotheek;
- berekeningsnota's schrijven voor het beoordelen van de apparaten;
- bewaren van de berekeningsnota's;
- helpen bij het ontwerpen van de projecten.

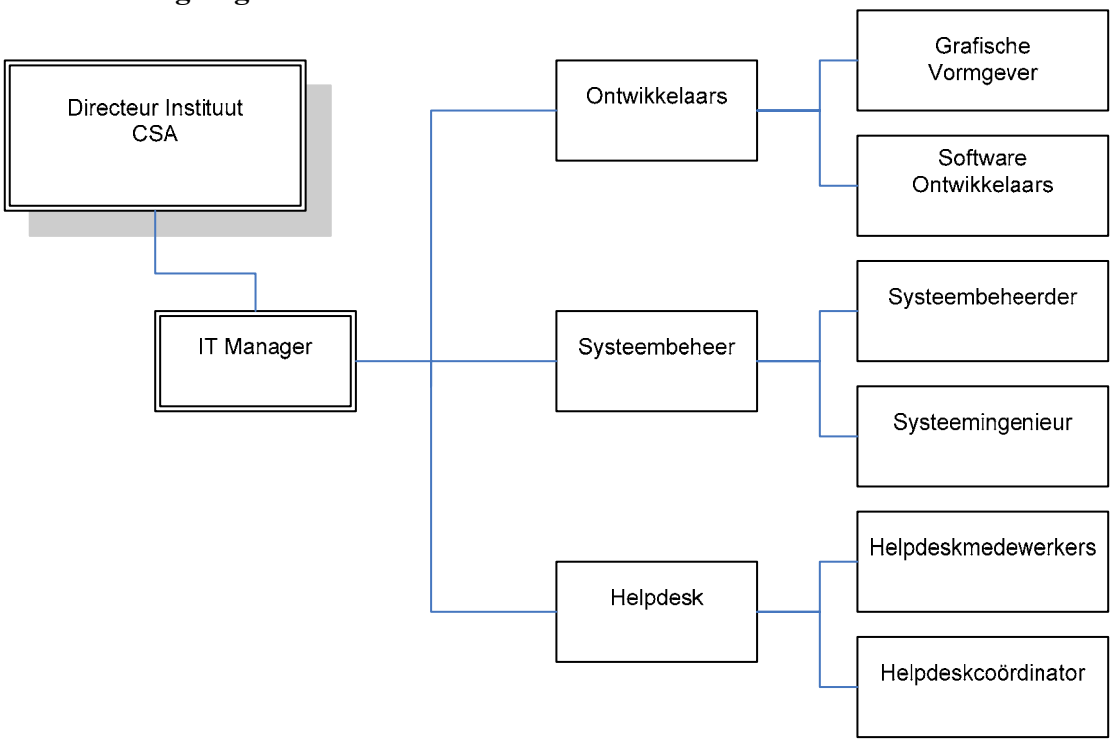
1.6 Infoplan

Infoplan is de informaticadienst van het SCK•CEN. Met hen ben ik het meest in contact gekomen in verband met problemen met software, installatie van nieuwe software, …

1.6.1 Algemeen

Information and Communication technology (ICT), genaamd Infoplan, behoort tot het instituut CSA (Communication, Support services, Administration).
Infoplan is georganiseerd in 3 groepen: de Helpdesk, Systeembeheer en Ontwikkeling.
Infoplan is ontstaan in 1992 en in de loop der jaren uitgegroeid tot een volwaardige centrale informaticadienst die alle aspecten van informatica aanbiedt zoals helpdesk, systeembeheer, software ontwikkeling en advies.
De informatica-infrastructuur bestaat uit een zeer performant netwerk, een uitgebreid server park met grote opslagcapaciteit, en een modern pc-park. Eindgebruikers werken met de nieuwste Microsoft Windows en Microsoft Office. Voor het technisch en wetenschappelijk werk is een groot aantal softwaretoepassingen in gebruik.

1.6.2 Organigram



Figuur 1.6.2.1: Organigram Infoplan

1.6.3 Taakomschrijving

Deze dienst verzorgt alle informaticabehoeften op het gebied van:

- aankopen van hardware, software en diensten;
- beveiligen van gegevens;
- onderhouden van het computerpark;
- toegang tot het lokaal netwerk en internet beheren;
- organiseren van informatica-opleidingen;
- softwareontwikkeling voor beheer en administratie.

2 DE STAGEOPDRACHT

Het SCK•CEN werkt met projecten die lopen tussen departementen, gebouwen, personen. Per project zijn er documenten, afspraken, ideeën, … die men per team met elkaar wil delen. Een perfecte oplossing voor zo'n centraal systeem is een website op het intranet van het SCK•CEN.
Na de reorganisatie in 2006 vindt men de tijd dat het intranet vernieuwd moet worden. Men heeft ervoor gekozen om een SharePoint oplossing te implementeren. Momenteel slaat men per project de documenten vanaf de server lokaal op. Via SharePoint kan dit veel eenvoudiger en mijn stagebegeleider heeft mij dan ook gevraagd hier een oplossing voor te vinden.
Dankzij de technologie van Web Parts kan ik onderdelen schrijven die men op de SharePoint omgeving kan plaatsen. Ik moest 6 Web Parts ontwikkelen. Welke dit zijn, en hoe ze worden gemaakt kan je verderop vinden.
Een medewerker van het SCK•CEN kan zo'n Web Part op zijn eigen SharePoint pagina toevoegen om zo eenvoudig aan alle relevante data te bekijken. Men kan dan deze Web Part gaan aanpassen zodat hij gepersonaliseerd wordt.
De eerste 2 opdrachten die kreeg dienen als een soort document beheer systeem om berekenings- & technische nota's te downloaden(CO Calculation Notes & CO Technical Notes).
Web Parts 3 & 4 & 5 zijn gelijkaardig aan de eerste 2. Hierbij moet men verschillende types van technische tekeningen kunnen downloaden. Deze Web Parts moeten ook andere netwerkschijven doorzoeken naar tekeningen(DEO Mechanical Drawings, DEO Instrumentation Drawings & BR2 Drawings).
Web Part 6 (Bidasse) laat wetenschappelijke data zien vanuit het DOS-Programma "Bidasse".
Naast het schrijven & ontwerpen van deze Web Part moest ik ook een oplossing vinden om deze Web Parts eenvoudig te installeren op de SharePoint Server van het SCK•CEN.
De Web Parts werden geschreven met Engels als voertaal. Dit omdat het SCK•CEN een internationaal karakter heeft.

3 GEBRUIKTE TECHNOLOGIEËN

In dit hoofdstuk beschrijf ik de technologieën die ik gebruikt heb om mijn stage tot een goed einde te brengen. Omdat SharePoint de rode draad door mijn stage is geweest geef ik meer uitleg over wat SharePoint is in 3.1.
Zelfgeschreven Web Parts maken gebruik van de .NET Technologie van Microsoft. Meer informatie over wat de .NET Technologie is, kan je vinden 3.2.
Om de deployment (Web Parts op de SharePoint server krijgen) eenvoudiger te maken heb ik ook een aantal BATCH bestanden geschreven (3.3).
Web Parts dienen geschreven te worden in een Web Custom Control, meer informatie over een Web Custom Control vind je in 3.4.
Om mijn opdrachten te analyseren heb ik gebruik gemaakt van UML. Een algemene uitleg over UML vind je in paragraaf 3.5.
Gegevens die in een database staan moeten gelezen worden door de Web Part. Om zulke gegevens op te halen, maakt men gebruik van een andere programmeertaal. Namelijk SQL (3.6)
Om bepaalde zaken te vergemakkelijken heb ik ook een webservice geschreven die steunt op XML. Info over webservices vind je in 3.7, 3.8 is toegewijd aan XML.

3.1 Microsoft SharePoint

Microsoft Office SharePoint Server 2007 of MOSS 2007 is een server toepassing ontwikkeld door Microsoft. MOSS 2007 is een betalende uitbreiding op WSS (Windows SharePoint Services). Microsoft Office SharePoint Server maakt dus gebruik van de basis faciliteiten van WSS en bouwt hierop verder om grotere bedrijven meer functionaliteiten te bieden. WSS maakt deel uit van Windows Server 2003 en 2008 en wordt er "gratis" bijgeleverd.

3.1.1 Algemeen

Omdat MOSS 2007 een uitbreiding is op WSS zijn de functionaliteiten van Windows SharePoint Services automatisch terug te vinden in Microsoft Office SharePoint Server. De volgende functionaliteiten gelden dus voor beide pakken. Ik zal voor dit deeltje het woord "SharePoint" gebruiken om beide pakketten te benoemen.

Portal environment

SharePoint zorgt voor een veilige, aanpasbare en "enterprise-level" portaalomgeving waarin een team van personen kan werken. Omdat SharePoint alle informatie op 1 plaats beheert en verzameld is het perfect om op zoek te gaan naar deze informatie. Doordat SharePoint gebruik maakt van het internet kan een systeembeheerder gebruikers binnen Active Directory toegang geven tot een bepaald deel van het Sharepoint-portaal, onafhankelijk van de fysieke locatie waar men zich bevindt.

Team work

Dankzij het feit dat men in team op dit SharePoint portaal kan werken, is het mogelijk om samen aan projecten, documenten, taken, … te werken. SharePoint is een kant en klare omgeving (mits enkele configuraties) en perfect past in het intranet van een bedrijf.

.NET Framework

SharePoint maakt volledig gebruik van het sterke Microsoft .NET 2.0 Framework. Het .NET Framework is een Microsoft only feature. Het heeft een voorgeprogrammeerde bibliotheek met oplossingen en programmeervereisten. Zoals bijvoorbeeld een textbox of een gridview waarmee men gegevens uit een databank eenvoudig kan weergeven.

Web Parts

Een unieke feature voor SharePoint en de .NET omgeving is de Web Part. Web Parts zijn

kleine deeltjes van een internetpagina. Ze kunnen afzonderlijk worden geprogrammeerd en geïmplementeerd worden in een website of een SharePoint omgeving. Voorbeelden van Web Parts zijn: een geïntegreerde zoekmachine, een prikbord waar men mededelingen op kan posten, document beheer systeem,… Handig aan deze Web Parts is dat men ze eventueel kan sluiten of minimaliseren. Op deze manier hoef je dus niet de volledige pagina af te scrollen en kan je kiezen welke informatie op de pagina relevant is voor jou. Elke Web Part kan afzonderlijk beheerd worden over heel de SharePoint portal. Het handige hieraan is, dat men bijvoorbeeld 2 dezelfde Web Parts met 2 verschillende instellingen op dezelfde pagina kan plaatsen.

3.1.2 Microsoft Office Sharepoint Server 2007

Kort samengevat kan men dus dankzij WSS een intranet website opzetten, laten beheren en gebruiken door gebruikers uit Active Directory en zelfgemaakte Web Parts ontwikkelen die voor een bedrijf nodig zijn. Waarom zou men dan nog een de investering doen naar MOSS 2007?
Wel, Microsoft Office SharePoint Server zorgt voor een aantal exclusieve functionaliteiten die zeker en vast de moeite waard zijn. Elke feature die ik beschrijf past in het taart diagram hieronder. Je ziet dat "Windows SharePoint Platform" centraal staat. Elk feature kan dus onafhankelijk van elkaar geïmplementeerd worden op het SharePoint portaal. Ik beschrijf ze kort hieronder.

Figuur 3.1.2.1: Taardiagram MOSS 2007

My Site – Collaboration
MOSS zorgt ervoor dat elke gebruiker die toegang heeft een eigen website kan creëren. My site is een soort "Homepage" die ervoor zorgt dat een gebruiker automatisch zijn informatie vindt die relevant is voor hem. Hier staat ook de gebruiker zijn profiel en kan hij/zij foto's of bestanden delen met andere gebruikers. Sommigen zien dit als een "Professional MySpace".
User Profiles – Collaboration, Portal, Search
Dankzij de Active Directory integratie worden profielen van werknemers op het portaal geplaatst. Elke individuele gebruiker kan kiezen welke informatie van zichzelf hij wil laten zien aan andere collega's. Search opdrachten lopen automatisch door deze user profiles.
Site Manager – Portal, Content Management
MOSS zorgt voor een simpele "Drag-and-drop" interface voor het beheren van navigatie onderdelen van het portaal. Beheren van links, site hiërarchieën is veel eenvoudiger en mogelijk gemaakt op plaatsen waar dat niet kon in Windows SharePoint Services.
Search – Search
Zoeken naar relevante informatie is nog nooit zo krachtig geweest. Microsoft Office

SharePoint Server wordt geleverd met een zoekmachine die niet enkel SharePoint pagina's doorzoekt, maar ook andere niet-Sharepoint pagina's binnen het intranet.

Niet enkel een information search maar ook een people search is ingebouwd. Stel dat ik op duizenden websites sta met VB.NET inhoud. Dan zal de people search functie van MOSS 2007 mijn profiel opgeven en aanduiden als lokaal expert.
Zoek resultaten zijn gericht naar "enterprise and line-of-business" data.
Business Data Catalog – Business Intelligence, Business Processes
Ik denk dat dit een van de meest belangrijke functionaliteiten van MOSS 2007 is. Microsoft Office Sharepoint Server zorgt ervoor dat men informatie van externe programma's kan integreren in pagina's of Web Parts. Bijvoorbeeld kan men gegevens uit een SQL Server database halen en deze op de pagina plaatsen. Niet enkel databases maar ook andere SAP & BI tools kunnen door SharePoint aangesproken worden.
Document Workflow – Business Processes
Door deze functie kan men op een eenvoudige wijze hele workflows & whitepapers rond bepaalde documenten of projecten schrijven.
Excel Services – Business Intelligence, Business Processes
Dankzij de "Shared Services" faciliteit kan men data van een Excel bestand plaatsen in een Web Part, men kan dus over heel de organisatie in 1 en dezelfde Excel Spreadsheet werken zonder gebruik te maken van versiebeheer of andere workarounds.
Dashboard and Report centre – Business Intelligence, Business Processes
Via deze functionaliteit kan men ingebouwde reporten en dashboards in Web Parts implementeren. Op deze manier kan men bekijken welke documenten en functies de gebruikers veel gebruiken. Men kan dan via deze informatie relevante documenten sneller beschikbaar maken of functies uit de SharePoint omgeving verwijderen om werknemers productiever te maken.

3.1.3 Versies

Net zoals Microsoft Office 2007 verschillende versies heeft naargelang het gebruik en de professionaliteit van gebruikers (studenten en thuisgebruikers, professional, enterprise,..) heeft Microsoft Office SharePoint Server 2007 ook verschillende versies naar gelang de omvang en de impact op het bedrijf. Om Microsoft Office Sharepoint Server 2007 te mogen gebruiken heeft men 1 of meerdere licenties nodig.
MOSS 2007 heeft buiten WSS 2 versies:

- **Microsoft Office SharePoint Server 2007 for Internet Sites**
- **Microsoft Office SharePoint Server 2007, Server License**

Als men voor de MOSS 2007 for Internet Sites kiest dan betaalt men voor een pakket dat op het internet geplaatst wordt (niet voor het intranet). Gebruikers van SharePoint mogen dan geen werknemers zijn.
Als men MOSS 2007 voor een intranet wil gebruiken dan moet men de Server License aanschaffen. Enkel op deze manier kan MOSS 2007 in Client/Server mode worden gebruikt. Naargelang de CAL's of Client Access License heeft een gebruiker meerdere rechten of functies om te gebruiken. Een CAL geeft een gebruiker ook het recht om MOSS 2007 te gebruiken. Deze Client Access License dient men alleen te kopen als men Sharepoint als intranet omgeving wil gebruiken. Microsoft heeft de volgende CAL's in de aanbieding:

- **Microsoft Office SharePoint Server 2007 CAL, Standard Edition**
- **Microsoft Office SharePoint Server 2007 CAL, Enterprise Edition**

De Standard Edition van de CAL geeft de gebruiker recht tot het gebruiken van de basis functionaliteiten zoals het beheren van de inhoud en business processes, zoek functies, My Site, …
Als men de hogere BI functies en de Excel Services functionaliteiten wil gebruiken, dient de gebruiker zowel een Standard als een Enterprise CAL te kopen.

3.2 .NET Framework

Bij het verschijnen van het .NET Framework is het door Microsoft gepresenteerd als een "Web Services platform". Hoewel het zeer geschikt is voor het maken en gebruiken van Web Services, is het .NET Framework veel meer. Het .NET Framework is ontwikkeld om vele problemen op te lossen die we tegenkomen bij het ontwikkelen van moderne applicaties. De strategie die Microsoft heeft gebruikt om te komen tot een idee als het .NET Framework kan samengevat worden in 3 puntjes:

- Beschikbaarheid op ieder moment van de dag (24x7)
- Beschikbaarheid op allerlei locaties
- Uit te voeren op allerlei verschillende hardware

Samengevat bestempelt Microsoft dit met de slogan: "Anytime, anywhere, and on any device".
Het .NET Framework is een omhulsel rond het operating systeem (bvb: Windows XP), dat ontwikkelaars kunnen gebruiken om moderne applicaties te ontwikkelen.

3.3 BATCH bestanden

Batch bestanden zijn tekstbestanden die commando's bevatten om het operating systeem te sturen. Een batch bestand bundelt commando's die je anders manueel, commando per commando moet gaan intypen. Batch bestanden zijn oorspronkelijk afkomstig vanuit het MS-DOS tijdperk, maar worden hedendaags veel gebruikt om processen te automatiseren. Een batch bestand kan je herkennen aan de .bat extensie en dient niet gecompileerd te worden. Je hebt dus geen extra software nodig, je kan ze via kladblok openen, aanpassen, …

3.4 Web Custom Control

Microsoft Visual Studio 2008 is de programmeer omgeving die ik heb gebruikt om mijn Web Parts te schrijven. Omdat een Web Part gedeployed moet worden in de Global Assembly Cache (zie Hx : Deployment) heeft Microsoft een template project gemaakt, namelijk de Web Custom Control.
Deze is veel gecompliceerder om te coderen als een Web User Control omdat men geen grafische designer heeft en men alle controls (textboxen, labels, …) via code moet aanmaken.
Bij het aanmaken van een nieuw Web Custom Control project, worden automatisch Manifest.xml & .dwp bestanden aangemaakt. Deze bestanden zijn nodig om onze Web Part in SharePoint te krijgen.

3.5 UML

De Unified Modelling Language (UML) is een van de meest gebruikte methodes om de visies van systeemontwikkelaars begrijpelijk en gestandaardiseerd vast te leggen. Het laat je toe om diagrammen te tekenen die begrijpelijk zijn voor de klant, de analist en de programmeur. Het is ontstaan begin jaren '90 en sindsdien is het een gestandaardiseerd & onmisbaar begrip geworden in de softwareindustrie.
UML bestaat uit een aantal grafische elementen die tot diagrammen worden gecombineerd. Omdat het een taal is, bevat UML regels voor het combineren van deze elementen.

De diagrammen die ik gebruikt heb om mijn Web Parts te verduidelijken zijn:

- Het Use Case diagram
- Het Activity diagram

3.5.1 Use Case diagram

Een Use Case diagram is een voorstelling van de interacties tussen de gebruiker en het systeem. De Use Case maakt duidelijk welke handelingen er nodig zijn om een gebeurtenis / functie te initialiseren.

3.5.2 Activity diagram

Een Activity diagram is een voorstelling van handelingen die het systeem doet aan de hand van interacties met de gebruiker of andere programma's.

3.6 SQL

SQL staat voor Structured Query Language. Met een SQL-statement kun je gegevens in een database gaan benaderen. SQL heeft de kracht om via code (zonder grafische interface) bepaalde rijen aan te passen, selecteren, verwijderen, …
Enkele simpele voorbeelden van SQL statements:

```
SELECT naam, straat, huisnummer FROM Klanten WHERE
achternaam='caeyers'
```

Deze SQL-Statement geeft een lijst terug met daarin de naam, straat en huisnummer van alle records in de tabel "klanten" met een achternaam "caeyers".

```
DELETE FROM Klanten WHERE achternaam="caeyers"
```

Bovenstaand statement zorgt ervoor dat alle records met de achternaam "caeyers" worden verwijderd uit de tabel Klanten.

```
INSERT INTO Klanten (naam,achternaam,straat,huisnummer)
VALUES ('denis','caeyers','Egelsvennen','157')
```

Deze SQL-Statement zorgt ervoor dat er een nieuwe record wordt aangemaakt in de tabel Klanten. De waarden 'denis','caeyers','Egelsvennen','157' worden ingevuld in de respectievelijke kolommen naam, achternaam, straat en huisnummer.
Omdat mijn Web Parts geen functionaliteiten moeten bevatten om gegevens aan te passen, verwijderen, toevoegen zal je enkel de SELECT-statements tegenkomen in mijn code.

3.7 Webservices

Webservices zijn kleine applicaties die over het internet beschikbaar zijn. Ze zijn dus van overal toegankelijk.
In tegenstelling tot gewone webapplicaties beschikken webservices niet over een grafische interface. Meestal worden webservices gebruikt en kleine functies in te schrijven zodat ontwikkelaars deze functies telkens opnieuw kunnen gebruiken.

In plaats van grote databanken, gebruiken webservices XML bestanden om hun gegevens op te halen.

3.8 XML

Op het eerste zicht lijkt XML op HTML. HTML is een taal om websites mee te schrijven, met nadruk op de lay-out en de manier waarop gegevens worden weergegeven.
XML daarentegen, heeft niets te maken met hoe gegevens worden weergegeven. XML bestaand aandacht aan wat het gegeven tussen de tag is. Met als gevolg dat XML bestanden kunnen gebruikt worden in plaats van een duur database programma te kopen.

4 ANALYSE

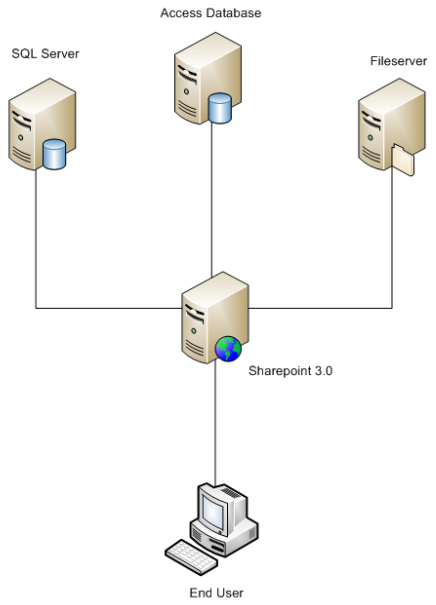
Een analyse wordt gemaakt voor de programmacode uiteindelijk wordt geschreven. Ze bepaald eigenlijk hoe het programma zich gedraagt en welke functies, componenten er nodig zijn.
Als eerste heb ik een paar globale analyses gemaakt (4.1) om alles een beetje beter in kaart te brengen en om beslissingen te maken.
Hierna komen de analyses van we Web Parts aanbod.(4.2, 4.3, 4.3)

4.1 Globale analyses

Voor ik begonnen ben met het analyseren & programmeren van de Web Parts heb ik eerst beknopte analyses gemaakt die voor mij noodzakelijk waren om bepaalde dingen beter te begrijpen.
Allereerst heb ik de componenten die betrekking hadden tot mijn Web Parts even in kaart gebracht. Dit was handig omdat ik zo een beter overzicht had over hoe alles met elkaar communiceert.Deze analyse kan je vinden in paragraaf 4.1.1
Na mijn eerste informatiezoektocht kreeg ik de keuze uit 2 programmeer talen,namelijk C# en VB.NET. Om een keuze te maken heb ik deze 2 talen even tegenover elkaar afgewogen (4.1.2).
De derde globale analyse handeldt over de Web Parts, hoe zijn deze opgebouwd? En welke bestanden heeft men nodig om zelf een Web Part te schrijven? (4.1.3)

4.1.1 Componenten analyse

De Web Parts die ik moest ontwikkelen maken gebruik van verschillende servers en services. Om meer zicht te krijgen op de verschillende servers, wat ze zijn, wat ze doen, lijkt het mij interessant om ze even in kaart te brengen en een korte uitleg te geven.



Figuur 4.1.1.1: Componenten diagram

SQL Server

De SQL server is een op het netwerk aangesloten machine die het programma "Microsoft SQL Server 2005" draait. Dit programma is een databankprogramma. Een databank is een groot bestand met gegevens. Het kan honderden gigabyte groot zijn en kan miljoenen records bevatten. Een databank bestaat uit tabellen. Tabellen zijn een soort "kasten" waarin bepaalde gegevens bewaard worden. Bijvoorbeeld, de databank "SCK" heeft een tabel

"personen" die gegevens bevat over Denis Caeyers, met zijn adres, e-mail, .. en gegevens over Philippe Gouat met zijn adres, e-mail, ...
In mijn omgeving werkte ik met de databank "SCKCEN". In deze databank vond men de tabellen "projects" en "tasks". De tabel "projects" bevat informatie over de proeven, uitvindingen, onderzoeken die gedaan worden op het SCK. De tabel "tasks" bevat onderdelen van projecten. Bijvoorbeeld, bij het project "Nuclear Waste Disposal" hoort de taak "Transfer container A to Storagehousing B". De reden waarom ik deze database nodig had, is omdat sommige Web Parts documenten moeten tonen die behoren tot een bepaalde taak.
Microsoft SQL Server 2005 is server-based. Dit wil zeggen dat men vanaf elke PC via de "Management Console" (programma om gegevens te bekijken, toevoegen,..) kan inloggen op deze server zonder al de gegevens op zijn PC te hebben.

Access Database

Access is een programma dat inbegrepen zit in "Microsoft Office". Net zoals SQL Server is Access een databankprogramma. In tegenstelling tot "Microsoft SQL Server 2005" is een Access Database client-based. Dit wil zeggen dat de persoon die deze gegevens wil bekijken de databank (die soms enkele gigabytes groot kan zijn) op zijn PC moet opslaan.
Een mogelijke oplossing om deze database te openen zonder de gegevens op zijn PC te hebben is door middel van een "Web applicatie". Een Web applicatie zorgt ervoor men de inhoud van de databank online of op het intranet kan bekijken.
Per Web Part heb ik een andere Access Database gebruikt. 5 van de 6 Web Parts die ik moest ontwikkelen dienen om documenten te downloaden naar de gebruiker zijn PC. De documenten die men moet kunnen downloaden staan niet in deze databank maar op een externe "File Server".

File Server

Een fileserver is een machine die dient om via gedeelde mappen, documenten en bestanden te bewaren. Deze machine is net zoals de "SQL Server" aangesloten op het netwerk, waardoor iedereen (met de juiste rechten) hierop kan connecteren en bestanden of documenten afhalen. De "Calculation Notes" en "Technical Notes" staan bijvoorbeeld op een fileserver.

SharePoint

Hoe houden deze 3 servers nu verband met elkaar? De oplossing hiervoor is Sharepoint en de Web Parts die ik moest ontwikkelen. Door middel van de centraliserende eigenschap kan men deze 3 componenten met elkaar verbinden.
Documenten horen bij een taak. Taken horen bij een project. In de Access database staat in de tabellen het veldje "Task Code". Deze Task Code verwijst naar de code die aan een bepaalde taak (in de SQL Server database) wordt gegeven. Om te weten welk document bij welke taak hoort, dien ik de gegevens op te vragen waarbij de Task Code in de Access Database hetzelfde is als de Task Code in de SQL Server Database.
SharePoint zal er dus voor zorgen dat men met 1 enkele Web Part per Project & Taak (SQL Server + Access Database) documenten vanaf de File server kan downloaden (Fileserver + Access Database).

4.1.2 C# vs VB.NET

Na mijn eerste informatie zoektocht op het internet naar informatie over SharePoint en zelf Web Parts schrijven, bleek al snel dat ik de keuze had tussen 2 programmeer talen. Namelijk C# en VB.NET. Web Parts die zelf geschreven zijn worden gecompileerd tot een .dll (Dynamic Link Library) bestand. Deze .dll zal SharePoint moeten kunnen lezen. SharePoint heeft zelf een achterliggende structuur van ASP.NET. ASP.NET kan op zich ook in C# of VB.NET geschreven worden. Vandaar dat ik de keuze had tussen deze 2 talen. De

functionaliteiten van beide talen zijn hetzelfde. C# heeft geen toegevoegde waarde over VB.NET. Ondanks dat C# sterk op VB.NET lijkt, zijn er toch een aantal verschillen.
Om een keuze te maken tussen deze 2 talen, heb ik een kleine analyse gemaakt over deze 2. Een vergelijking tussen C# en VB.NET vanuit mijn standpunt.

C#	VB.NET
+ Veel voorbeelden + Nieuwe ervaring, toegevoegde waarde op CV + Standaard ontwikkelingstaal op SCK • Geen ervaring met C# • Minder leesbaar • Meer kans op Syntaxfouten	• Niet zoveel voorbeelden • Geen toegevoegde waarde op CV + Meer ervaring door de lessen in VB.NET op de KHK + Beter leesbaar. + Minder kans op Syntaxfouten

Ervaring speelde voor mij een grote rol. C# is niet in het lessenpakket van de opleiding Toegepaste Informatica opgenomen, dus ik had nog nooit een letter code C# gezien, laat staan geschreven. Langs de andere kant, als ik mijn Web Parts in C# zou schrijven, zou dit een nieuwe ervaring zijn en een extra toegevoegde waarde zijn op mijn CV. Dit liet mij weer zeer hard twijfelen.
Een groot verschil tussen C# en VB.NET is dat VB.NET makkelijker leesbaar is. Een voorbeeld:

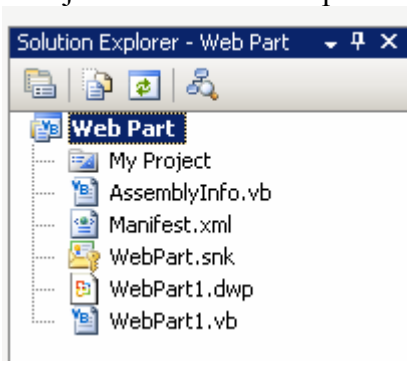
C#	VB.NET

if (age < 20)
{
greeting = "What's up?";
}
else
{
greeting = "Hello";
}
	If age < 20 **Then**
	greeting = "What's up?"
	Else
	greeting = "Hello"

Zoals je kan zien, kan je VB.NET bijna lezen als een zin. C# daarentegen, maakt het lezen iets moeilijker door de ; en de vele {. Doordat C# deze karakters veel gebruikt, is de kans dat men een; vergeet of een haakje vergeet te sluiten zeer groot. In kleine stukken code zoals deze, maakt het niet veel uit. Maar bij meer gecompliceerde code, is dit toch een groot verschil.
Bij het opzoeken naar hoe ik moest beginnen aan het schrijven van een Web Part, vond ik bijna enkel C# code. Voorbeelden werden bijna telkens in C# geschreven. Dit liet mijn keuze weer meer afwijken naar C#.
Uiteindelijk heb ik gekozen om mijn code te schrijven in VB.NET. De reden waarom is omdat de ervaring toch een grote rol speelde. C# zou het schrijven van mijn Web Parts enkel moeilijker maken, en ik zou misschien in tijdsnood komen. Op het internet zijn C# naar VB.NET converters te vinden. Deze converters vertalen C# code naar VB.NET code. De talrijke voorbeelden gingen dus niet verloren.
De SharePoint community is zeer groot, dit heeft mij ook overhalen om VB.NET te gebruiken. Tijdens mijn stage heb ik een Blog opgericht, waarin ik basisbeginselen uitleg en tips geef over hoe men zelf Web Parts moet schrijven, en hoe men met SharePoint werkt. Alle code die ik daarop plaats, is geschreven in VB.NET. Ik hoop op die manier mensen die ook liever met VB.NET werken, te helpen.

4.1.3 Web Parts

In dit onderdeeltje leg ik uit welke bestanden men nodig heeft om zelf een Web Part te schrijven. De solution explorer van Visual Studio ziet er als volgt uit:



Figuur 4.1.3.1: Solution Explorer Visual Studio 2008

My Project

Eerst in het lijstje staat "My Project". Dit is enkel en niet meer een verwijzing naar de eigenschappen van het project. Hierin kan men bijvoorbeeld de naam van de uiteindelijke DLL instellen, de naam van het project, of men een XML bestand moet aanmaken of niet, …

AssemblyInfo.vb

Onder "My Project" staat "AssemblyInfo.vb" hierin staan eigenschappen die betrekking hebben tot de assembly die later in de Global Assembly Cache word weggeschreven. Men kan hier bijvoorbeeld een beschrijving of een naam aan de assembly toevoegen.

Manifest.xml

Een ander bestand dat nodig is om een assembly in de GAC te deployen is "Manifest.xml". Zoals de extensie al zegt, is manifest.xml een XML bestand. Het bevat informatie & verwijzingen naar de naam van het DLL bestand, het DWP bestand, …

WebPart.snk

Om een DLL in de GAC te deployen, is het nodig om een project een Strong Name te geven. Een Strong Name is een aantal karakters die door Visual Studio worden gekozen. De Strong Name zorgt ervoor dat ieder project uniek is zodat er geen fout wordt veroorzaakt als 2 verschillende projecten dezelfde naam hebben.

WebPart1.dwp

Een DWP bestand is net zoals "Manifest.xml" een XML bestand. De inhoud heeft echter niets met elkaar te maken. Een DWP bestand bevat informatie die SharePoint gebruikt om de Web Part te tonen. Hier kan men bijvoorbeeld een beschrijving toevoegen, of de uiteindelijke naam die SharePoint laat zien aan de gebruikers. Het bevat dus enkel informatie over hoe de Web Part op het scherm getoond wordt.

WebPart1.vb

Het VB bestand bevat de uiteindelijke code en zorgt er voor dat een Web Part werkt.

4.2 CO Calculation Notes – CO Technical Notes

De eerste opdracht die ik kreeg was voor het Calculation Office zelf. Zoals eerder gezegd, beschikt het rekenbureau over talloze technische informatie & berekeningen in verband met materialen, warmteoverdracht, …

Deze 2 Web Parts dienen om team-leden van bepaalde projecten toegang te geven tot die informatie, via hun My Site ,homepagina of project hoofdpagina van het vernieuwde intranet.

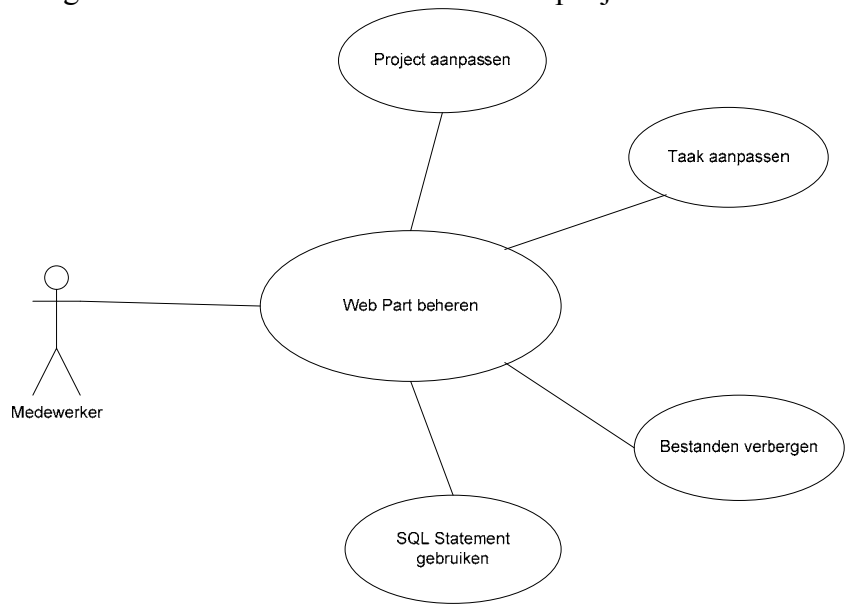
Omdat CO Calculation Notes en CO Technical Notes bijna een identieke kopie zijn van elkaar, geldt deze analyse voor beide Web Parts.

4.2.1 Use Case diagrammen

Deze 2 Use Case diagrammen geven de functionele vereisten van de Web Part aan. Ze verduidelijken wat de Web Part moet kunnen.

4.2.1.1 Web Part beheren

Web Part beheren verduidelijkt wat de instellingen zijn die de gebruiker moet kunnen configureren om de Web Part af te stellen op zijn voorkeur.

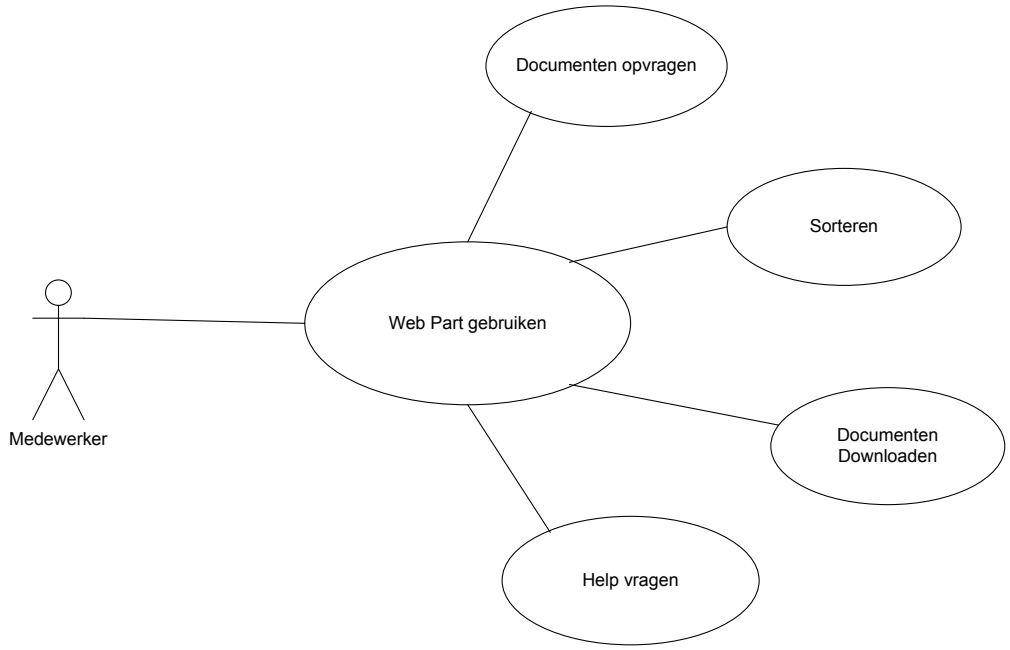


Figuur 4.2.1.1.1: Use Case Web Part beheren

Als enige Actor hebben we de medewerker. Deze medewerker is de persoon die de Web Part op zijn My Site of Portal plaatst.
De medewerker moet zijn eigen Web Part kunnen beheren en dus aanpassen naar eigen voorkeur. Hiervoor moet de gebruiker dus een Project kunnen aanpassen, hij moet het standaard project dat wordt aangegeven bij het initialiseren van de Web Part kunnen aanpassen aan het Project waar aan hij, samen met zijn team aan het werken is. Na het aanpassen van het project moet de gebruiker een Taak kiezen waarvan hij bepaalde documenten wil kunnen bekijken.
Om niet relevante informatie tijdelijk te verbergen, moet de medewerker bestanden kunnen verbergen.
Indien een medewerker documenten nodig heeft van meerdere taken of van meerdere verschillende projecten, kan de medewerker een SQL Statement ingeven die als bron van de documententabel gebruikt zal worden.

4.2.1.2 Web Part gebruiken

Naast het beheren moet de gebruiker ook de Web Part kunnen gebruiken. Deze Use Case geeft de functies weer die de Web Part moet kunnen nadat ze geconfigureerd is.

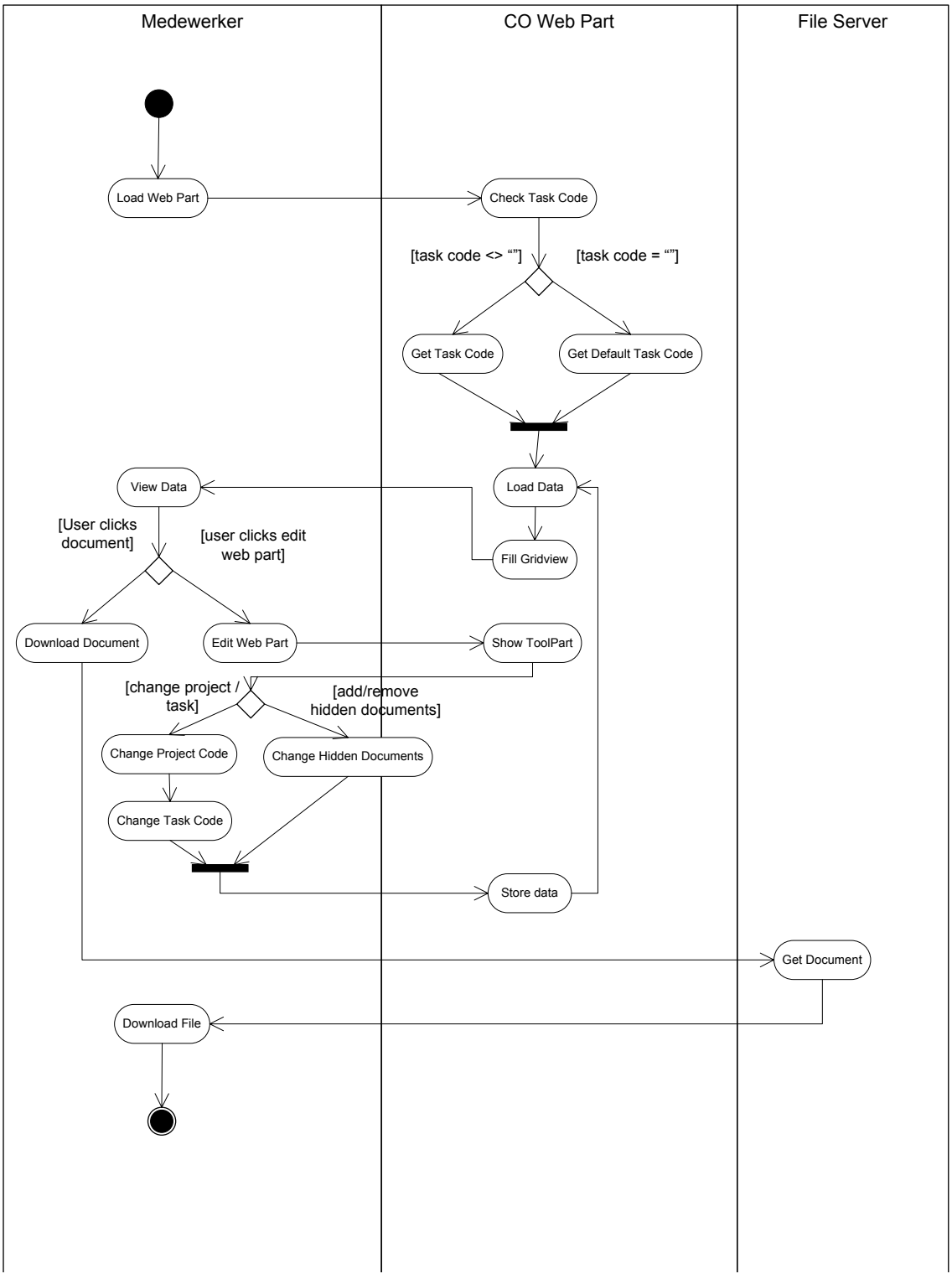


Figuur 4.2.1.2.1: Use Case Web Part gebruiken

Ook hier hebben we dezelfde medewerker als in de Use Case "Web Part Beheren". De medewerker neemt dus 2 rollen in, als gebruiker van de Web Part en als beheerder van de Web Part.
Om de Web Part te kunnen gebruiken moet hij dus de documenten kunnen opvragen. Deze documenten worden dus weergegeven in de tabel van documenten op de SharePoint pagina. De medewerker moet ook in de tabel kunnen sorteren, zowel opwaarts als neerwaarts.
Als hij zijn relevant document heeft gevonden, moet de gebruiker de mogelijk hebben om dit bestand te kunnen downloaden.Tenslotte moet de medewerker een Help-bestand kunnen opvragen.

4.2.2 Activity diagram

Het activity diagram verduidelijkt welke stappen het programma moet doen aan de hand van de verschillende componenten die gebruikt, aangesproken worden.



4.2.3 Databases

Zoals hiervoor al gezegd, moet de Web Part documenten weergeven naargelang een gekozen project & taak. De Projecten & Bijhorende taken worden bewaard in een SQL Server database. Het enige wat de gebruiker moet kiezen is de code van het project, en de code van de taak. Uit deze SQL Server database heb ik dus maar 2 tabellen nodig. Namelijk de tabel "projects" en de tabel "tasks". 1 project heeft meerdere taken, dit wil dus zeggen dat er een 1 op veel relatie tussen de tabellen ligt.

Op het SCK•CEN is dit niet het geval. Men kan de projectcode afleiden aan de hand van de taak code. De eerste 4 karakters van de taak code, is de project code. Bvb: een taak met als code: "A001457" hoort bij het project: "A001". Hieronder ziet u 2 screenshots van de tabellen die ik nodig heb.

Table - dbo.projects | Summary

project	name	Division	divName	Departement	depName	Section
A000	WET.PROGR.: S...	DG	ALGEMENE DIRE...	BSU	BEDRIJFSONDE...	BSU
A001	SURETE BR2 ...	NMS	WETENSCH.NUC...	BR2	BELGISCHE REA...	BR2
A002	VEILIGHEID LHM...	NMS	WETENSCH.NUC...	IOP	UITBATING INF...	IOP
A003	VEILIGHEID BR1...	ANS	GEAVANCEERDE...	RRO	UITBAT ONDERZ...	RRO
A004	VORMING PERS...	BR2	BR2 / no mo...	99999	default DEP ...	BR2EXP
A005	VERHUIZINGEN ...	ANS	GEAVANCEERDE...	ANS	GEAVANCEERDE...	ANS
A006	PROGR.INTERN...	ANS	GEAVANCEERDE...	RTD	ONTWERP REAC...	RTD

Figuur 4.2.3.1: Screenshot table "projects"

Table - dbo.tasks | Summary

Task	TaskName	StartYear	StartMonth	EndYear	EndMonth
A001000	BR2 V&K : MANA...	1993	1	9999	99
A001010	BR2 BR2 V&K : TECHN.SEKRETARI		1	9999	99
A001011	BR2 V&K : VEILI...	1993	1	9999	99
A001012	BR2 V&K : BRAN...	1993	1	9999	99
A001013	BR2 V&K : INDU...	1993	1	9999	99
A001020	BR2 : ONDERHO...	1993	1	9999	99
A001021	BR2 ONDERH.ST...	1993	1	9999	99

Figuur 4.2.3.2: Screenshot table "tasks"

Naast het gebruik van een SQL Server database, heb ik ook gebruik moeten maken van een Microsoft Access database. Deze database bevat alle informatie in verband met de Calculation Notes (tabel Calculation Notes) en de Technical Notes (tabel Technical Notes). De tabellen bevatten volgende informatie:

Calculation Notes:

Field Name	Data Type
Cell	Number
Task Code	Text
ID	Number
Revision	Text
Author	Text
Date	Date/Time
Title	Text
Counter	Number
Folder	Number

Figuur 4.2.3.3: Screenshot Calculation Notes

Technical Notes:

Field Name	Data Type
Cell	Number
Task Code	Text
ID	Number
Revision	Text
Author	Text
Date	Date/Time
Title	Text
Counter	Number
Folder	Number

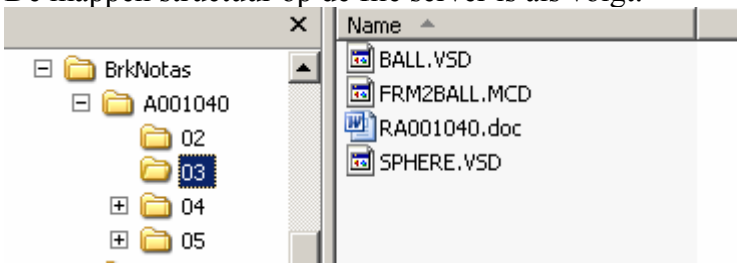
Figuur 4.2.3.4: Screenshot Technical Notes

Je ziet dat deze 2 tabellen identiek zijn. Het enige verschil tussen deze 2 tabellen is de informatie die in de kolommen is ingevuld.
Uit deze tabellen zijn volgende kolommen toepasselijk voor onze 2 Web Parts:

- Task Code: Bevat de taak code die verwijst naar de taak
- ID: Bevat het volgnummer van de nota, per project taak
- Revision: Bevat de recentste versienummer van de nota
- Date: De datum waarop het document laatst is aangepast
- Title: De titel van de nota
- Folder: De plaats waar het bestand fysiek staat.

4.2.4 Mappenstructuur

Om documenten te downloaden heeft men de naam van het document nodig. Een document begint met de letter "R", daarna de Task Code en daarna de extensie van het bestand. Bijvoorbeeld een document dat hoort bij de taak "0147" heeft als volgt de naam "R0147.doc". Dankzij het veldje Task Code in de Access database kunnen we de naam van het document afleiden.
De mappen structuur op de file server is als volgt:



Figuur 4.2.4.1: Screenshot mappenstructuur

De map "BrkNotas" staat voor de BerekeningsNotas of "Calculation Notes". Deze map is onderverdeeld in submappen met de code van elke taak als naam ("A001040"). Hieronder vindt men weer enkele submappen met cijfers. Deze cijfers staan in voor de versies van de documenten. In de inhoud zien we dan het overeenkomstig document genaamd "RA001040.doc" (R voor Report).
De Map "03" staat voor het ID van de nota.

4.3 DEO Mechanical Drawings – DEO Instrumentation Drawings – BR2 Drawings

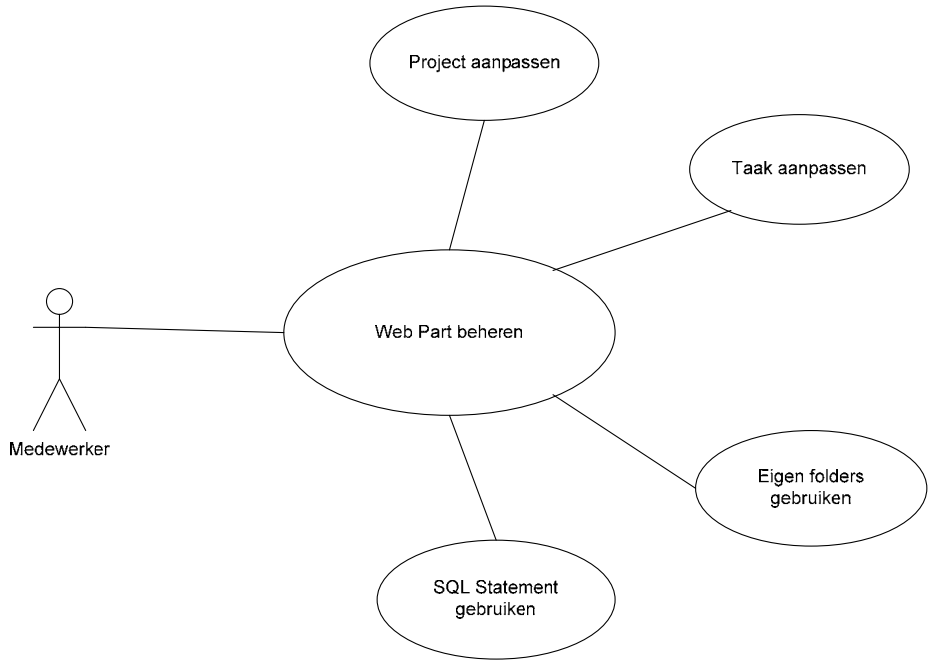
De tweede opdracht die ik kreeg was voor het Tekenbureau. Het tekenbureau tekent de plannen / ontwerpen die de ingenieurs bedenken in verband met experimenten.
Deze 3 Web Parts dienen om team-leden van bepaalde projecten toegang te geven tot tekeningen. De DEO Web Parts hebben een extra functie om gebruikers hun eigen tekeningen bij in de Tabel te stoppen.
Omdat DEO Mechanical Drawings, DEO Intrumentation Drawings en BR2 Drawings bijna een identieke kopie zijn van elkaar, geldt deze analyse voor de 3 Web Parts.

4.3.1 Use Case diagrammen

Deze 2 Use Case diagrammen geven de functionele vereisten van de Web Part aan. Ze verduidelijken wat de Web Part moet kunnen.

4.3.1.1 Web Part beheren

Web Part beheren verduidelijkt wat de instellingen zijn die de gebruiker moet kunnen configureren om de Web Part af te stellen op zijn voorkeur.

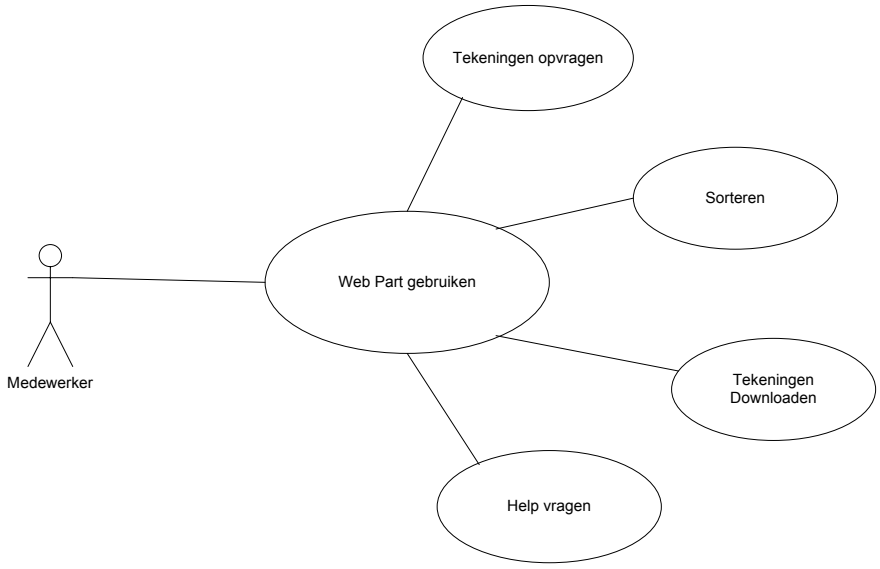


Figuur 4.3.1.1.1: Use Case Web Part beheren

Als enige Actor hebben we de medewerker. Deze medewerker is de persoon die de Web Part op zijn My Site of Portal plaatst.
De medewerker moet zijn eigen Web Part kunnen beheren en dus aanpassen naar eigen voorkeur. Hiervoor moet de gebruiker dus een Project kunnen aanpassen, hij moet het standaardproject dat word aangegeven bij het initialiseren van de Web Part kunnen aanpassen aan het Project waar hij, samen met zijn team aan het werken is. Na het aanpassen van het project moet de gebruiker een Taak kiezen waarvan hij bepaalde documenten wil kunnen bekijken.
Om eigen tekeningen in de tabel te integreren moet de gebruiker zijn eigen folders kunnen opgeven. Men kan 3 verschillende folders specificeren. Namelijk folders voor PDF-bestanden, TIF-bestanden en DWF-bestanden. Deze folders kunnen gedeelde mappen over het netwerk zijn, of de Document List binnen de huidige SharePoint Portal.
Deze functie is enkel mogelijk bij de DEO Web Parts. Voor BR2 Drawings was dit niet nodig.
Indien een medewerker tekeningen nodig heeft van meerdere taken of van meerdere verschillende projecten, kan de medewerker een SQL Statement ingeven die als bron van de documenten tabel gebruikt zal worden.

4.3.1.2 Web Part gebruiken

Naast het beheren moet de gebruiker ook de Web Part kunnen gebruiken. Deze Use Case geeft de functies weer die de Web Part moet kunnen nadat ze geconfigureerd is.



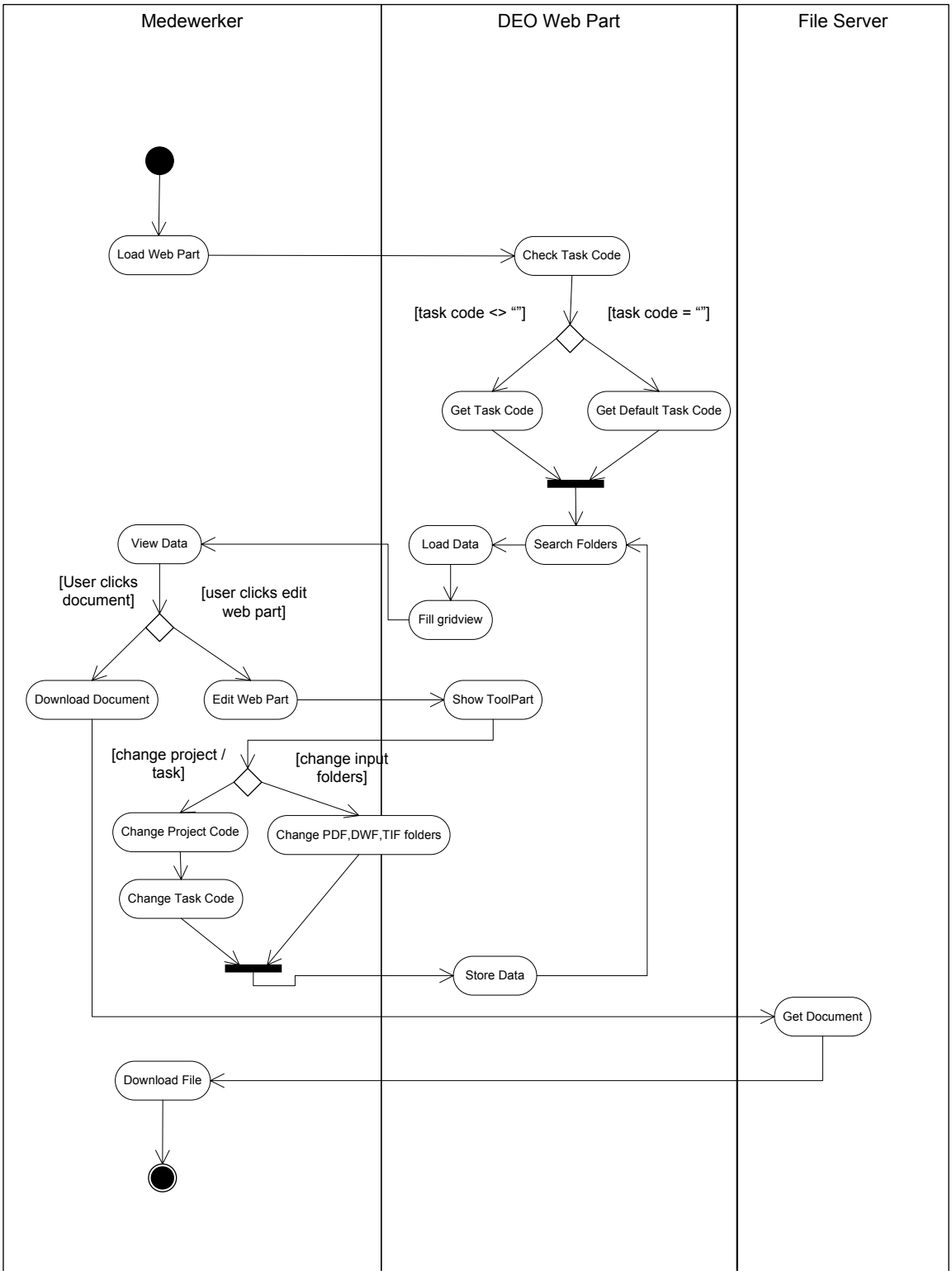
Figuur 4.3.1.1.1: Use Case Web Part gebruiken

Ook hier hebben we dezelfde medewerker als in de Use Case "Web Part Beheren". De medewerker neemt dus 2 rollen in, als gebruiker van de Web Part en als beheerder van de Web Part.
Om de Web Part te kunnen gebruiken moet hij dus de tekeningen kunnen opvragen. Deze tekeningen worden dus weergegeven in de tabel van documenten op de SharePoint pagina. De medewerker moet ook in de tabel kunnen sorteren, zowel opwaarts als neerwaarts.
Als hij zijn relevante tekening heeft gevonden, moet de gebruiker de mogelijk hebben op dit bestand te kunnen downloaden. Ook hier heeft men de keuze om de 4 verschillende tekeningen te kunnen downloaden (DWF, PDF, TIF & DWG).
Tenslotte moet de medewerker een Help-bestand kunnen opvragen.

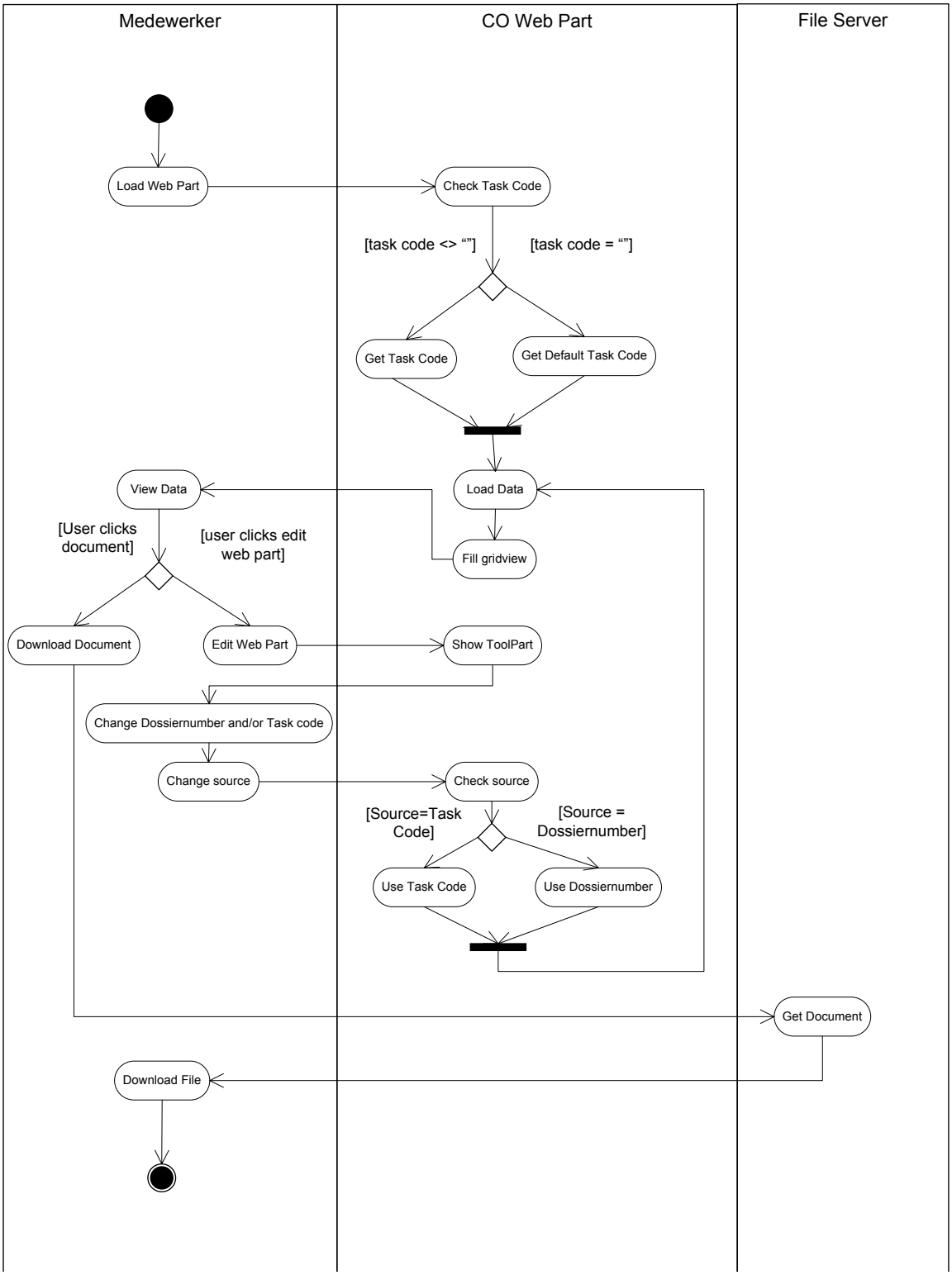
4.3.2 Activity Diagrams

Omdat BR2 Drawings een iets andere manier van werken heeft, en omdat er geen eigen folder gespecificeerd moeten worden, heeft BR2 Drawings een ander Activity Diagram dan de DEO Web Parts.

4.3.2.1 DEO Mechanical Drawings – DEO Instrumentation Drawings



4.3.2.2 BR2 Drawings



4.3.3 Databases

Ook deze Web Parts gebruiken de Microsoft SQL Server Database om de project- en taakcodes op te halen. Zie paragraaf 4.2.3 voor meer uitleg.

Naast het gebruik van een SQL Server database, heb ik ook gebruik moeten maken van een Microsoft Access databases. Deze databases bevat alle informatie in verband met de Mechanische tekeningen en de Instrumenten Tekeningen.
De tabellen bevatten volgende informatie:

Mechanical Drawings:

Field Name	Data Type
Plannummer	Text
Revisie	Text
Dossier	Text
Taaknummer	Text
Titelhoek 1	Text
Titelhoek 2	Text
Getekend	Text
Datum getekend	Text
Schaal	Text
Formaat	Text
Samenzicht	Text
Stuklijst	Text
Dienst	Text
Nazicht	Text
Datum nazicht	Text
Goedkeuring	Text
Datum goedkeuring	Text
Coordinator	Text
Datum coordinator	Text
Status	Text
Revisie A	Text
Datum revisie A	Text
Wijziging revisie A	Text
Revisie B	Text
Datum revisie B	Text
Wijziging revisie B	Text
Revisie C	Text
Datum revisie C	Text
Wijziging revisie C	Text
File	Text

Instrumentation Drawings:

Field Name	Data Type
Plannummer	Text
Revisie	Text
Taaknummer	Text
Dossier	Text
Getekend	Text
Goedkeuring	Text
Formaat	Text
Datum Getekend	Text
Titelhoek 1	Text
Titelhoek 2	Text
Nazicht	Text
Datum Nazicht	Text
Datum Goedkeuring	Text
Dienst	Text
Schaal	Text
Status	Text
Revisie A	Text
Datum Revisie A	Text
Wijziging Revisie A	Text
Revisie B	Text
Datum Revisie B	Text
Wijziging Revisie B	Text
Revisie C	Text
Datum Revisie C	Text
Wijziging Revisie C	Text
File	Text

Figuur 4.3.3.1: Screenshot database Mechanical Drawings
Figuur 4.3.3.2: Screenshot database Instrumentation Drawings

Je ziet dat deze 2 tabellen bijna identiek zijn. De velden die ik nodig had, zijn hetzelfde genaamd in beide tabellen, ik beschrijf ze kort:
Uit deze tabellen zijn volgende kolommen toepasselijk voor onze 2 Web Parts:

- Plannummer: Bevat het nummer van het tekenplan
- Revisie: Bevat de recentste versie nummer van de nota
- Dossier: Bevat een verwijzing naar een dossier. Bvb: Experimenten beginnen hun dossiernummer met een "X"
- Taaknummer: De code van de taak waar dit plan bijhoort.
- Titelhoek1: De eerste titel van het plan
- Titelhoek2: De tweede titel van het plan
- File: Verwijzing naar de fysieke plaats waar het .DWG bestand staat.

4.3.4 Samenhang tussen verschillende bestanden

Om er voor te zorgen dat bijvoorbeeld een PDF bestand van een gebruiker op dezelfde plaats in de tabel wordt weergegeven als het overeenkomstige DWG bestand, moeten beide bestanden dezelfde naam hebben.

Om dus gaan na te kijken of een DWG bestand bijhorende PDF, DWF of TIF bestanden heeft. Moest ik dus uit de lange string 'File' in de database, enkel de naam uitfilteren. En deze gaan vergelijken met elk bestand dat in de door de gebruiker opgegeven map staat. Bvb: uit de string '\\server1\tekeningen\tch\projectcode\tekening1.dwg' moet ik naam van het bestand 'tekening1' halen.
Dit is niet toepasselijk voor BR2 Drawings

4.4 Bidasse

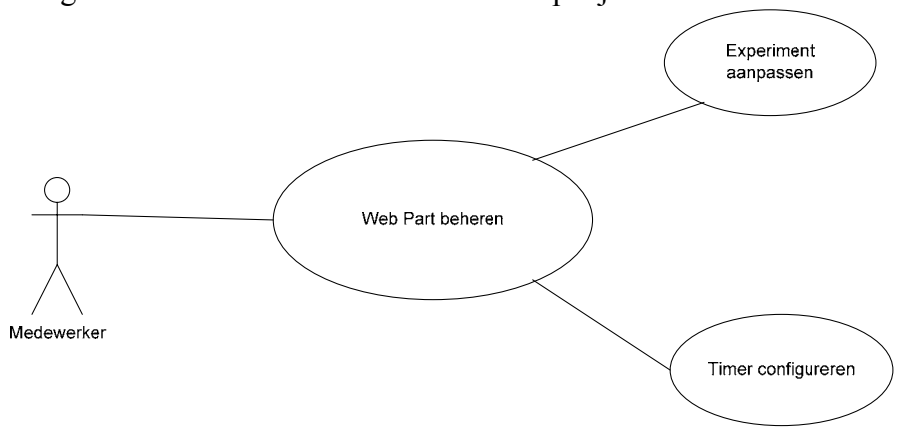
Om bepaalde metingen van een experiment op te vragen, gebruikt het SCK•CEN een MS DOS programma genaamd "Bidasse". Bidasse geeft een real-time weergave van metingen die gedaan worden op experimenten.
De Web Part "Bidasse" moet de waarden die worden weggeschreven door het MS DOS programma "Bidasse" uitlezen, en deze tonen op een SharePoint pagina.

4.4.1 Use Case Diagrammen

Deze 2 Use Case diagrammen geven de functionele vereisten van de Web Part aan. Ze verduidelijken wat de Web Part moet kunnen.

4.4.1.1 Web Part Beheren

Web Part beheren verduidelijkt wat de instellingen zijn die de gebruiker moet kunnen configureren om de Web Part af te stellen op zijn voorkeur.



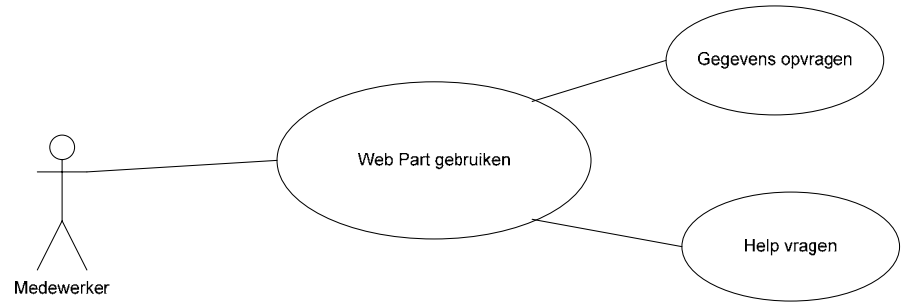
Figuur 4.4.1.1.1: Use Case Web Part beheren

Als enige Actor hebben we de medewerker. Deze medewerker is de persoon die de Web Part op zijn My Site of Portal plaatst.

De medewerker moet zijn eigen Web Part kunnen beheren en dus aanpassen naar eigen voorkeur. Hiervoor moet de gebruiker dus een experiment kunnen aanpassen, hij moet het standaardexperiment dat wordt aangegeven bij het initialiseren van de Web Part kunnen aanpassen aan het experiment waar hij de waarden van wil opvragen.
Hiernaast, kan de medewerker ook de timer configureren. De timer zorgt ervoor dat de SharePoint pagina elke minuut automatisch wordt gerefreshed zodat telkens de nieuwste waarden worden ingeladen.

4.4.1.2 Web Part gebruiken

Naast het beheren moet de gebruiker ook de Web Part kunnen gebruiken. Deze Use Case geeft de functies weer die de Web Part moet kunnen nadat ze geconfigureerd is.

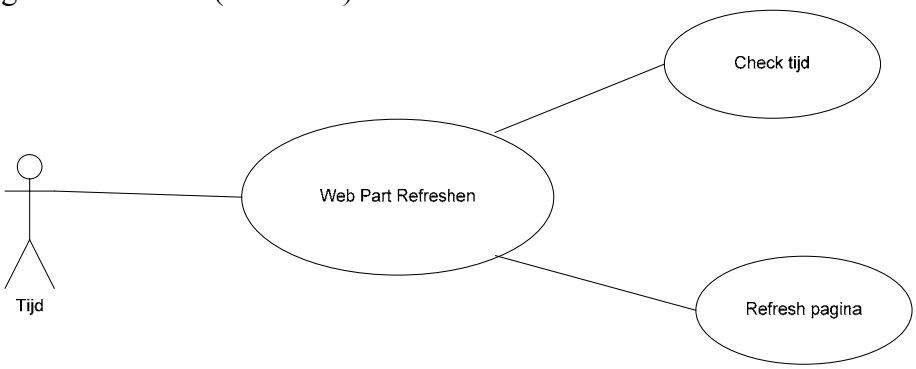


Figuur 4.4.1.1.2: Use Case Web Part gebruiken

Ook hier hebben we dezelfde medewerker als in de Use Case "Web Part Beheren". De medewerker neemt dus 2 rollen in, als gebruiker van de Web Part en als beheerder van de Web Part.
Om de Web Part te kunnen gebruiken moet hij dus de gegevens van het project kunnen opvragen. Deze gegevens worden dus weergegeven in de tabel op de SharePoint pagina. Tenslotte moet de medewerker een Help-bestand kunnen opvragen.

4.4.1.3 Web Part refreshen

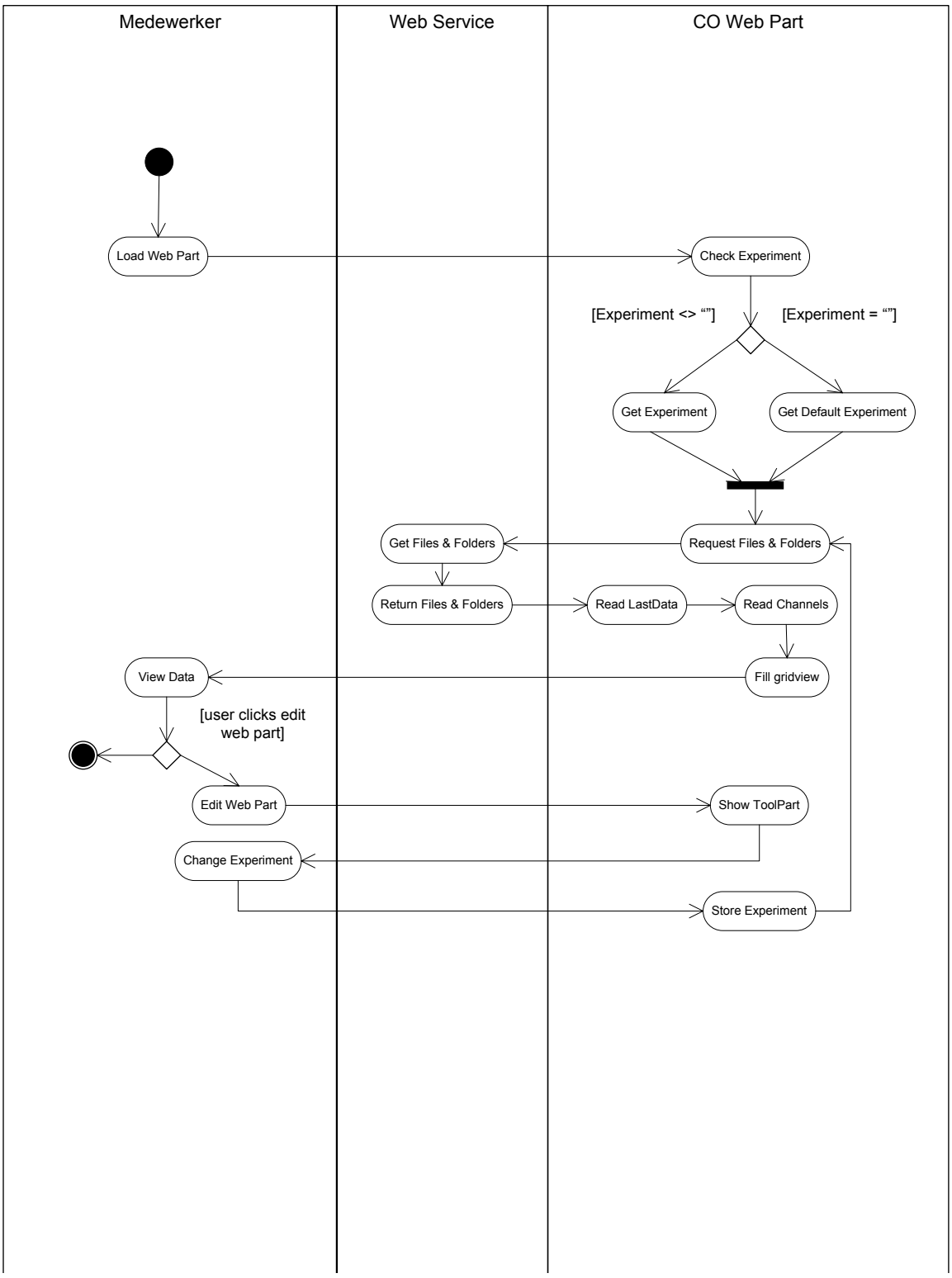
Om de gegevens real-time te houden moet de SharePoint pagina elke minuut opnieuw geladen worden (refreshen).



Figuur 4.4.1.1.3: Use Case Web Part refreshen

De Actor tijd is dus de teller die per seconde optelt, als hij aan 60 zit, wordt de SharePoint pagina opnieuw geladen.

4.4.2 Activity Diagram



4.4.2 Bestanden

Zoals bovenaan gezegd, werkt "Bidasse" niet met een database, maar met losse bestanden. De bestanden die gebruikt worden beschrijf ik hier onder.

Lastdata.dat
Om de laatst gemeten gegevens weg te schrijven, gebruikt "Bidasse" het binair bestand "lastdata.dat". Elk gemeten item, word dus in binaire vorm weggeschreven.
Experimentname.cha
Cha bestanden beschrijven de kanalen van de gemeten waarden. Hierin vind je bijvoorbeeld de naam van waarde, de eenheid & gegevens om later grafieken te tekenen.

5 CODE

In dit hoofdstuk wordt de uiteindelijke programmacode beschreven. Omdat alle Web Parts qua opbouw bijna hetzelfde zijn, beschrijf ik eerst de gemeenschappelijke programmacode die door de Web Parts wordt gebruikt.
Nadien beschrijf ik de code van elke Web Part.

5.1 Gemeenschappelijke Code

Vele stukken code zoals het aanmaken van een gridview, sorteer procedure van een gridview, dataconnectie, .. zijn hetzelfde in de verschillende Web Parts. Hieronder kan je de uitleg van deze stukken code vinden. Ik maak het onderscheid tussen Klassen, Procedures en Snippets.

5.1.1 Klassen

Een klasse is een soort van container die code bevat. Men kan in de hoofdcode deze klasse implementeren en zo de procedures en functies in de klasse gebruiken. Het is een typisch object binnen een OO-Programmeer omgeving zoals VB.NET.

5.1.1.1 DBSQLServer

Deze klasse heb ik geschreven om connectie te maken met de SQL Server van het SCK•CEN.

Gebruik

Om gebruik te maken van deze klasse moet men eerst deze klasse bekend maken aan de hoofdcode.Dit kan men doen door de volgende code:

```
Dim sqlserver As DBSQLserver
```

Vervolgens moet men eerst de connectiestring binnen deze klasse definiëren. Dit kan men bijvoorbeeld doen door de volgende code:

```
sqlserver.strConnection = "…"
```

In bovenstaande code wordt … vervangen door de connectie string die men nodig heeft om connectie met de SQL Server te maken.
Nadien kan met via de functie GetSQLResult gegevens opvragen via een SQL statement. Deze procedure geeft een DataTable terug met hierin de opgehaalde gegevens.

```
Dim table as new DataTable
table = sqlserver.GetSQLResult("select * from table")
```

Code

Elke code geschreven in VB.NET begint met het importeren van een aantal bestaande namespaces. Deze namespaces geven ons de mogelijkheid om specifieke methodes en functies te gebruiken binnen een project. Om de klasse DBSQLserver te schrijven moest ik de volgende namespaces importeren.

```
Imports System
Imports System.Data.SqlClient
```

```
Imports System.Data
```

System is de algemene klasse die ons de mogelijkheid geeft om een aantal algemene zaken te gebruiken.
Buiten de algemene klasse, importeerde ik ook System.Data en System.Data.SqlClient. Deze 2 klassen zorgen ervoor dat ik connectie kan maken met een Microsoft SQL Server en er gegevens kan uitlezen. Naast het maken van de connectie, geven deze klassen ook de mogelijkheid om een DataTable te gebruiken. De DataTable bevat de gegevens van een Database zoals MS SQL Server.
Na het importeren van namespaces begint de eigenlijke code van de klasse. Onderstaande regel code geeft weer dat de klasse vanaf hier begint, en dat ze de naam "DBSQLserver" krijgt.

```
Public Class DBSQLserver
```

Variabelen die over heel de klasse gebruikt moeten kunnen worden declareren we bovenaan de code, en beginnen met "Private" in plaats van met "Dim". Doordat ze gedeclareerd worden met "Private", kunnen ze niet door een andere klasse worden aangesproken. Onze uiteindelijke hoofdcode, kan dus niet de variabele "da" in onderstaande code aanspreken. Ik licht de variabelen kort toe:

- _connection: Deze variabele zorgt voor het gebruik van de Property strConnection later
- cnn: cnn is een SqlConnection vanuit de namespace System.Data.SqlClient en zorgt voor de connectie met de MS SQL Server.
- da: is de SqlDataAdapter die de gegevens uit de Database haalt en ze doorgeeft aan de code.

```
'declaratie van variabelen om deze verder te gebruiken
   Private _connection As String
   Private cnn As SqlConnection
   Private da As SqlDataAdapter
   Public Sub DBSQLserver()
   End Sub
   'property instellen om strConnection te setten & getten
   Public Property strConnection() As String
       Get
           Return _connection
       End Get
       Set(ByVal value As String)
           _connection = value
       End Set
   End Property
```

Het laatste deeltje van bovenstaande code is het aanmaken van de property strConnection. Een property word aangemaakt om gegevens van de ene klasse naar de andere klasse door te geven. Om connectie te maken met een database, heb je een connection string nodig. Deze string bevat informatie over waar de database staat, welke beveiliging men moet toepassen, … De connectionstring kan dankzij de Property strConnection van de hoofdklasse naar de klasse DBSQLserver doorgebracht worden.

Naast het leggen van de connectie met de database, moet deze klasse ook gegevens kunnen ophalen en terug doorgeven aan de hoofdklasse.
Onderstaande functie "GetSQLResult" verwacht dat men een string doorgeeft. Deze string moet de SQL-Statement zijn die zegt welke gegevens men uit welke tabel wil halen en geeft nadien een DataTable terug, met hierin de opgehaalde gegevens.
Indien de connectie mislukt, of er bevinden zich geen gegevens in de tabel, wordt de DataTable leeg terug gegeven.

```
    'Deze functie legt de connectie met de database en geeft
    een DataTable met waarden terug
        Public Function GetSQLResult(ByVal sql As String) As
    DataTable

             Dim dt As New DataTable
             Try
           'connectie maken met de database in strConnection
                cnn = New SqlConnection(strConnection)
             'Gegevens ophalen via de sql-statement, over de
    connectie en gegevens in de DataTable stoppen
                da = New SqlDataAdapter(sql, cnn)
                da.Fill(dt)
           'Gevulde DataTable terugsturen
                Return dt
           'Foutopvanging
             Catch ex As Exception
                Return dt
             End Try
     End Function
```

De tweede procedure die deze klasse bevat, is "CloseConnection". CloseConnection zorgt ervoor dat de connectie wordt gesloten, en dat de DataAdapter die de gegevens verstuurd wordt leeg gemaakt.

```
Public Sub CloseConnection()
        da.Dispose()
        cnn.Close()
End Sub
```

5.1.1.2 DBAccess

VB.NET maakt verschil tussen een connectie met een MS SQL Server database en een MS Access database. In gebruik zijn deze 2 klassen net hetzelfde, ze hebben allebei dezelfde procedures, variabelen en property's maar deze klasse maakt connectie met een MS Access database in plaats van met een MS SQL Server database.

Gebruik

Om gebruik te maken van deze klasse moet men eerst deze klasse bekend maken aan de hoofdcode. Dit kan men doen door de volgende code:

```
Dim access as DBAccess
```

Vervolgens moet men eerst de connectie string binnen deze klasse definiëren. Dit kan men bijvoorbeeld doen door de volgende code:

```
access.strConnection = "..."
```

In bovenstaande code word … vervangen door de connectiestring die men nodig heeft om connectie met de SQL Server te maken.
Nadien kan met via de functie GetSQLResult gegevens opvragen via een SQL statement. Deze procedure geeft een DataTable terug met hierin de opgehaalde gegevens.

```
Dim table as new DataTable
table = access.GetSQLResult("select * from table")
```

Code
Doordat DBAccess bijna identiek is aan DBSQLserver, geef ik de verschillen in de code hieronder weer.
Zoals in iedere klasse begint men met het importeren van de namespaces. System & System.Data worden ook geïmporteerd in deze klasse. Enkel wordt System.Data.SqlClient vervangen door de namespace die mogelijkheden geeft tot dataconnectie met een Access database.

```
Imports System.Data.OleDb
```

Variabelen verschillen ook. In plaats van de SQL server klassen, gebruikt men de OleDb varianten.

```
Private cnn As OleDbConnection
Private da As OleDbDataAdapter
```

Ook in de functie "GetSQLResult" zijn de OleDb varianten aangepast.

```
cnn = New OleDbConnection(strConnection)
da = New OleDbDataAdapter(sql, cnn)
```

5.1.2 Procedures

Een procedure bevat een aantal regels code die ervoor zorgen dat dubbel werk wordt voorkomen. Of gewoon om de duidelijkheid en de leesbaarheid van de code te verbeteren. Men kan dus een aantal regels code afzonderen en enkel verwijzen naar de naam van de procedure. Bvb:

```
Public sub PrintHelloLabel()
     Dim HelloLabel as new Label
     HelloLabel.text = "Hello"
     Me.controls.add(HelloLabel)
End sub

Protected Sub Page_Load(ByVal sender As Object, ByVal e As
System.EventArgs)
```

```
            PrintHelloLabel()
End sub
```

De Procedure "PrintHelloLabel" maakt een nieuw label aan, plaatst de tekst "Hello" in het label, en voegt daarna het label toe aan het uiteindelijke scherm.
Door te werken op deze manier hoeft men later niet de code opnieuw te schrijven, maar kan men verwijzen naar "PrintHelloLabel".

5.1.2.1 Gridview sorteren

Een van de basisvereisten van elke Web Part is dat men de gridview die op het scherm getoond wordt, kan sorteren via bepaalde kolom koppen.
Om dit mogelijk te maken heb ik een procedure geschreven die deze taak voor zich neemt.
Eerst en vooral, moet er in de code waar de gridview word aangemaakt, een handler aan de gridview worden toegevoegd.

```
'handler toevoegen om sorteren mogelijk te maken
AddHandler grid.Sorting, AddressOf Me.grid_Sorting
```

De handler behandelt het Sorting event van de gridview. Dit wil zeggen, als men dit Event aanstuurt (door bijvoorbeeld te klikken op een kolomkop) word de procedure "grid_Sorting" uitgevoerd.
De code maakt gebruik van ViewStates. Een ViewState bewaart de toestand van de objecten in een pagina.
Een duidelijk voorbeeld van een ViewState is een online aankoop formulier. Je wil een product kopen en geeft je naam, adres, .. in. Je klikt op de "Koop" knop en je merkt dat je een typfout hebt gemaakt in je adres. Als je nu op de "Back-knop" van je browser klikt zie je dat je informatie nog steeds is ingevuld en dat je ze simpelweg kan aanpassen. Dit is waar de ViewState zijn ding doet. Vroeger was dit niet zo, en moest je heel het formulier terug invullen.
De procedure zelf, ziet er als volgt uit.
Eerst en vooral moet de procedure worden gestart door volgende regel:

```
'Deze methode zorgt ervoor dat men in 2 richtingen kan
sorteren
'Ze handelt de Handler af die vooraf werd gedefinieerd.
Private Sub grid_Sorting(ByVal sender As Object, ByVal e As
GridViewSortEventArgs)
```

Nadien word de variabele lastExpression aangemaakt, en gaat men kijken of men hiervoor al gesorteerd heeft. Als dit zo is, word de naam van de lastExpression opgevuld met de SortExpression ViewState.

```
Dim lastExpression As String = ""
If Not ViewState("SortExpression") Is Nothing Then
       lastExpression = ViewState("SortExpression").ToString
End If
```

Nu gaat men kijken of de vorige expression, gelijk is met de expression van de kolom waarop men heeft geklikt. "e" geeft de kolomkop van de gridview weer.

Als de SortExpression gelijk is aan lastExpression. Dus als men voor een 2de maal op de kolomkop klikt. En de huidige sorteer richting "DESC" (voor neerwaarts sorteren). Dan word de nieuwe sorteer richting "ASC" (voor opwaarts sorteren).
Als dit niet het geval is, gaat men kijken of de SortExpression gelijk is aan de lastExpression en of de huidige sorteer richting "ASC" is. Als dit zo is, wordt de sorteer richting "DESC".
Als geen van beide mogelijkheden het geval is. Dus als men op een andere kolomkop klikt dan de vorige. Wordt de sorteer richting automatisch terug opwaarts.

```
If e.SortExpression = lastExpression And newDirection =
"desc" Then                    newDirection = "asc"
ElseIf e.SortExpression = lastExpression And newDirection =
"asc" Then
            newDirection = "desc"
Else
            newDirection = "asc"
End If
```

Als laatste wordt de nieuwe ViewState SortExpression gelijk aan de kolomkop, en de sorteer richting aan de nieuwe sorteer richting.

```
ViewState("SortExpression") = e.SortExpression
ViewState("SortDirection") = newDirection
```

De DataView die de gegevens bevat wordt gesorteerd volgens bovenstaande expressies en de gridview wordt opnieuw geladen volgende de huidige waarden.

```
dv.Sort = (e.SortExpression + (" " + newDirection))
grid.DataBind()
End Sub
```

5.1.2.2 CreateDefaultLabel

Om een standaard tekst met een uitleg in over hoe men de Web Part moet configureren weer te geven heb ik een procedure "CreateDefaultLabel" geschreven. Deze procedure zorgt er voor dat als de gekozen taakcode op "Default" staat, er een label wordt weergegeven.
Allereerst moet men dus gaan kijken of de taakcode op "Default" staat. Als dit zo is, wordt de procedure "CreateDefaultLabel" opgeroepen. Als dit niet zo is, gaat men verder met de procedure "CreateGrid" die de SPGridView aanmaakt en configureert.
In beide gevallen wordt de procedure "CreateHelpLink" aangeroepen. Deze procedure genereert een label dat verwijst naar de Help-pagina.

```
'Om andere controls weer te geven moeten we de
CreateChildControls() procedure overschrijven
    Protected Overrides Sub CreateChildControls()
        MyBase.CreateChildControls()
        'Procedure oproepen om het gridview aan te maken
        If Text = "Default" Then
            CreateDefaultLabel()
        Else
            CreateGrid()
```

```
        End If
        CreateHelpLink()
    End Sub
```

De programmacode van "CreateDefaultLabel" ziet er als volgt uit:

```
Private Sub CreateDefaultLabel()
        Dim lbl As New Label
        lbl.Text = "To set up this Web Part, follow these
steps:"
        lbl.Text += "<br/>"
        lbl.Text += " - Enter edit mode: Site Actions > Edit
Page > Modify Shared Web Part"
        lbl.Text += "<br/>"
        lbl.Text += " - Choose a project & task code from the
Gridview Source ToolPart"
        lbl.Text += "<br/>"
        lbl.Text += " - Close and re-enter Edit Mode to
choose documents you want to hide. (optional)"
        lbl.Text += "<br/>"
        lbl.Text += " - Apply changes & exit edit mode"
        lbl.Text += "<br/>"
        lbl.Text += "<br/>"
        Me.Controls.Add(lbl)
    End Sub
```

5.1.3 Snippets

Een snippet is een stukje code dat je in klassen, procedures, … kan gebruiken. Het is niet meer dan ruwe code die voor verschillende projecten kan gebruikt worden.
De snippets die ik ga uitleggen is het opvangen van de Data in de DataTable die wordt terug gegeven door de DBAccess klasse. En het aanmaken van een SPGridView.

5.1.3.1 DataView / DataTable

Zoals eerder gezegd, bevat de namespace System.Data een aantal klassen die specifiek handelen naar dataconnectie. Zoals bijvoorbeeld de DataTable die de klasse DBAccess terug geeft aan de hoofdcode. Hoe gaan we in onze hoofdcode nu deze gegevens opvangen en bruikbaar maken?
Eerst en vooral moeten we in onze hoofdcode ook een DataTable declareren. Hier wordt ook een DataView gedeclareerd die we later gebruiken.

```
Private dt As DataTable
Private dv As DataView
```

Je ziet dat deze DataTable als Private wordt gedeclareerd. Dit heeft als reden omdat de grid_Sorting procedure deze DataTable moet kunnen aanspreken.
Om de DBAccess klasse te kunnen aanspreken moet ook deze gedeclareerd worden.

```
Dim db As New DBAccess
```

De volgende stap is de DBAccess klasse initialiseren door de strConnection property op te vullen met de connectionstring. De functie GetSQLResult van de klasse DBAccess geeft

zoals hiervoor gezegt een DataTable terug. Deze DataTable wordt opgevangen door onze "Private" DataTable

```
'connectiemaken en records ophalen
db.strConnection = ConnectionString
dt = db.GetSQLResult(SQL)
```

Om in een DataTable te kunnen sorteren, moet er een DataView op de DataTable gemaakt worden. Een DataView is vergelijkbaar met een View in de Database wereld. Je kan de gegevens aanpassen, anders laten tonen, zonder wijzigingen in de oorspronkelijke DataTable door te geven.

```
dv = New DataView(dt)
```

De gegevens in de DataTable zijn nu gekopieerd naar de DataView, waar bewerkingen kunnen uitvoeren zoals het sorteren, filteren en binden naar een SPGridView.

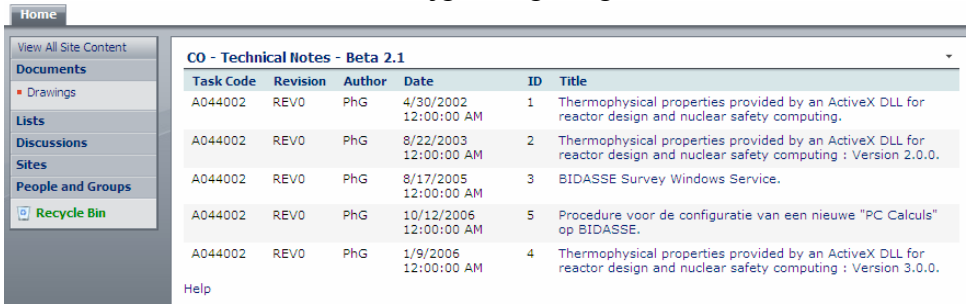
5.1.3.4 SPGridView

Een gridview is control die specifiek is voor de .NET Architectuur. In bijvoorbeeld PHP zal je dus nooit een GridView tegenkomen. Een Gridview is niet meer dan tabel met waarden. Je hoeft dus niet meer gebruik te maken van de HTML tags om een tabel te maken, maar je kan alle eigenschappen toewijzen aan de gridview.

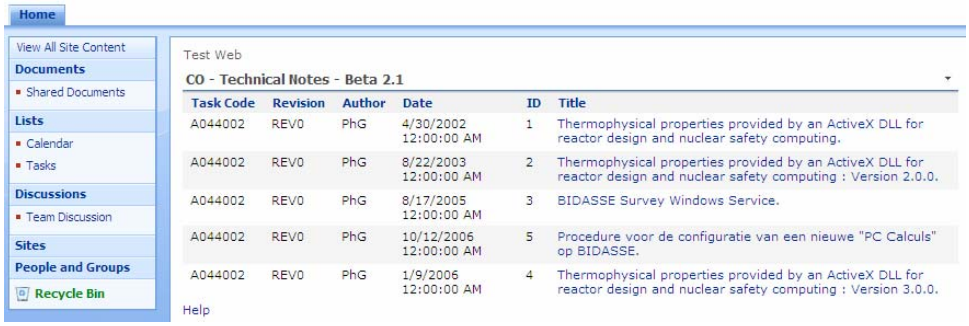
Omdat al de Web Parts die ik geschreven heb informatie moeten laten zien aan de eindgebruiker, was een Gridview de perfecte oplossing.

Microsoft heeft ook gedacht aan de ontwikkelaars van SharePoint Web Parts. Ze hebben een nieuwe soort van Gridview uitgebracht, namelijk de SPGridView. Een SPGridview is niet meer dan een gewone GridView. De reden waarom ik een SPGridView gebruik is omdat de lay-out van de GridView automatisch aanpast naargelang het thema dat een gebruiker kiest voor zijn SharePoint MySite.

In de onderstaande afbeeldingen kan je zien hoe dezelfde Web Part (met een SPGridview) een andere kleurencombinatie krijgt naargelang het thema van de SharePoint pagina.



Figuur: 5.1.3.4.1: SPGridView met zwarte lay-out



Figuur: 5.1.3.4.2: SPGridView met blauwe lay-out

Code

Een Gridview kan je op 2 manieren aanmaken. Je kan er voor zorgen dat je de gridview automatisch de kolommen van de DataTable overneemt. Of je kan je eigen kolommen specificeren zodat je niet relevante informatie eruit haalt.
Ik heb gekozen voor de 2de manier omdat de tabellen in de databases veel kolommen bevat die niet getoond dienen te worden.
Een gridview moet men net als alle andere controls eerst declareren. De SPGridView moet als "Private" gedeclareerd worden omdat de procedure grid_Sorting deze opnieuw gaat binden.

```
Private grid As New SPGridView
```

Als 2de stap moet je de SPBoundFields declareren. SPBoundFields stellen de kolommen van de SPGridView voor. Omdat de kolommen verschillen tussen de Web Parts, geef ik hier enkel een klein voorbeeld, de specifieke kolommen per Web Part leg ik uit in de beschrijving van de code van de Web Parts.

```
Dim bf1 As New SPBoundField
```

Volgende stap is het instellen van de eigenschappen van de SPBoundField Ik licht kort de ingestelde eigenschappen toe:

- .HeaderText: dit is de titel van de kolom in de GridView
- .DataField: deze eigenschap verwacht een string. De string stelt een kolom binnen de DataView voor.
- .SortExpression: ook deze eigenschap verwacht de naam van een kolom. Deze eigenschap wordt gebruikt door de procedure grid_Sorting om te sorteren op kolomkoppen.

```
With bf1
    .HeaderText = "Titel"
    .DataField = "Titel"
    .SortExpression = "Titel"
End With
```

Als alle SPBoundFields zijn ingesteld, worden de eigenschappen van de gridview ingesteld. Eerst en vooral, word de GridView duidelijk gemaakt dat hij de kolommen niet zelf moet aanmaken door de eigenschap AutoGenerateColumns op False te zetten.

```
With grid
    .AutoGenerateColumns = False
```

Nadien gaan we de SPBoundFields toevoegen aan de SPGridView, dit word gedaan voor elke kolom die in de gridview getoond moet worden.

```
    .Columns.Add(bf1)
```

De eigenschap DataSource krijgt de variabele dv mee, die staat voor de naam van de DataView die de gegevens bevat.

```
    .DataSource = dv
```

De volgende eigenschappen zijn niet noodzakelijk om een gridview te laten werken. Ik licht ze kort toe:

- .CellPadding: Geeft een spatie van 1 pixel tussen elke cell
- .EmptyDataText: Indien er geen gegevens in de DataView zitten word de tekst "No records found." weergegeven.
- .AllowSorting: Maakt de gridview duidelijk of er gesorteerd mag worden of niet.
- .AlternatingRowStyle.BackColor: Om een beetje meer duidelijkheid te geven stel ik de achtergrond kleur in van de oneven rijen.

```
    .CellPadding = 1
    .EmptyDataText = "No records found."
    .AllowSorting = True
    .AlternatingRowStyle.BackColor =
System.Drawing.Color.WhiteSmoke
```

Vervolgens moeten de gegevens in de DataView, aan de SPGridView worden gekoppeld.

```
    .DataBind()
End With
```

De laatste stap in het maken van een SPGridView, is de SPGridView toevoegen aan het scherm.

```
'gridview toevoegen aan scherm
Me.Controls.Add(grid)
```

5.2 CO Calculation Notes – CO Technical Notes

Het zelf schrijven van Web Parts voor een SharePoint omgeving is toch heel wat verschillend tegenover de werkwijze tijdens onze lessen. Daarom lijkt het mij verstandig om de code uitvoerig te beschrijven.
Omdat een Web Part moet geschreven worden via een Web Custom Control, moeten verschillende procedures overschreven in plaats van aangemaakt worden. Procedures die beginnen met "Protected Overrides Sub …" zijn dus procedures die nodig zijn om de Web Part te laten werken. Degene zonder "Overrides" zijn procedures die ik zelf heb aangemaakt.

5.2.1 Property's

Eerst en vooral worden er property's aangemaakt. Deze property's zorgen ervoor dat de code uit de algemene klasse kan worden aangestuurd door de ToolPart klassen.

```
'Declaratie van de variabelen en een constante begin waarde
plaatsen
    Private Const _defaultText As String = "B051021"
    Private Const _defaultSQL As String = ""
'declaratie van de property's
    Private _text As String = _defaultText
```

```
    Private _customSQL As String = _defaultSQL
    Private _array As List(Of String) = New List(Of String)
<Browsable(False), Category("Miscellaneous"),
DefaultValue(_defaultText), WebPartStorage(Storage.Personal),
FriendlyName("Insert Task Code"), Description("Text
Property")> Property [Text]() As String
        Get
            Return _text
        End Get

        Set(ByVal Value As String)
            _text = Value
        End Set
    End Property
    <Browsable(True), Category("SQL Statement"),
DefaultValue(_defaultSQL), WebPartStorage(Storage.Personal),
FriendlyName("Insert SQL Statement"), Description("Sql
Statement")> Property SQLFromToolPart() As String
        Get
            Return _customSQL
        End Get

        Set(ByVal Value As String)
            _customSQL = Value
        End Set
    End Property
    <Browsable(False), Category("Miscellaneous"),
WebPartStorage(Storage.Personal), FriendlyName("List"),
Description("List")> Property HideTable() As List(Of String)
        Get
            Return _array
        End Get

        Set(ByVal Value As List(Of String))
            _array = Value
        End Set
    End Property
```

Er zijn dus 3 property's aangemaakt."Text" om de project code in op te slagen, "SQLFromToolPart" om een eventuele SQL String te gebruiken en "HideTable" die de Notes bevat die verborgen moeten worden.
Elke Property krijgt ook nog een aantal parameters mee:

- Browsable: Als deze op "True" staat wordt er automatisch een ToolPart veldje aangemaakt.
- Category: Om de eventuele ToolPart naam in te stellen.
- DefaultValue: Om een standaard waarde mee te geven.
- WebPartStorage: Om de WebPart in aan te duiden.
- FriendlyName: Om een unieke naamgeving in weer te geven.
- Description: Om een beschrijving aan de property mee te geven.

5.2.2 ToolParts

In de ToolPart kan men eigenschappen ingeven die de weergave van de Web Part beïnvloeden. Om extra eigenschappen beschikbaar te maken, moeten ook deze zelf worden aangemaakt. ToolParts zijn klassen die worden gedeclareerd in de hoofdcode.

5.2.2.1 DropDownListToolPart

Deze ToolPart zorgt ervoor dat er 2 comboboxen worden weergegeven waaruit men de Project- & Taak Code kan kiezen.
In de methode New (die wordt aangeroepen als de ToolPart word geladen) worden alle controls geïnstancieerd:

```
Public Sub New()
        'Titel van de toolpart instellen
        Me.Title = "Gridview Source"
        'DropDown Lists configureren zodat ze telkens hun
veranderen terug sturen naar de server
        ddlProjects.AutoPostBack = True
        ddlTasks.AutoPostBack = True
        'Text instantiëren voor de labels (<br/> om lege
lijnen tussen de controls te laten)
        lblProjects.Text = "<br/>Select your project:<br/>"
        lblTasks.Text = "<br/><br/>Select your task:<br/>"
        spacer.Text = "<br/>"
        'Zorgen dat men geen lege dropdownlist te zien
krijgt.
        ddlTasks.AppendDataBoundItems = True
    End Sub
```

CreateChildControls is een procedure die overschreven moet worden. Ze zorgt ervoor dat de zelf aangemaakt controls zoals de DropDownLists op het scherm worden geplaatst.

```
Protected Overrides Sub CreateChildControls()
        MyBase.CreateChildControls()
        'Procedure aanroepen om de eerste combobox op te
vullen
        PopulateDropDownProjects()
        'controls toevoegen
        Me.Controls.Add(lblProjects)
        Me.Controls.Add(ddlProjects)
        Me.Controls.Add(lblTasks)
        Me.Controls.Add(ddlTasks)
        Me.Controls.Add(spacer)
        Me.Controls.Add(spacer)
    End Sub
```

Zoals je ziet, wordt de procedure "PopulateDropDownProjects()" aangeroepen. Deze Procedure zal de DropDownList "ddlProjects" opvullen met de waarden uit de Microsoft SQL Server Database.

```
Public Sub PopulateDropDownProjects()
        'declaratie
```

```
        Dim connectieStringSCK As String = "**********;"
        Dim connectieSCK As SqlConnection = New
SqlConnection(connectieStringSCK)
        Dim wp As COCN = Me.ParentToolPane.SelectedWebPart

        'connectie openen
        connectieSCK.Open()

        'commando & reader aanmaken, uitvoeren
        Dim command As SqlCommand = New SqlCommand("SELECT *
FROM projects", connectieSCK)
        Dim reader As SqlDataReader = command.ExecuteReader()

        'dropdownlist opvullen
        ddlProjects.DataSource = reader
        ddlProjects.DataValueField = "project"
        ddlProjects.DataTextField = "project"
        ddlProjects.DataBind()

        'handler toevoegen
        AddHandler ddlProjects.SelectedIndexChanged,
AddressOf Me.ddlProjects_SelectedIndexChanged

        'objecten sluiten
        reader.Close()
        connectieSCK.Close()
       'Standaard waarde ophalen vanuit de WebPart
        ddlProjects.SelectedValue = wp.Text.Substring(0, 4)
        ddlTasks.Items.Add(wp.Text)
        ddlTasks.Items.Add("Please select a project")
    End Sub
```

Belangrijk in deze code is dat er een handler wordt toegevoegd. Deze handler zorgt ervoor dat als men een Project kiest in de DropDownList "ddlProjects", de DropDownList "ddlTasks" wordt opgevuld.Daarnaast, wordt de hoofdcode ook aangesproken om de huidige Project Code als geselecteerde waarde weer te geven.
De code om de DropDownList "ddlTasks" op te vullen is gelijkaardig aan de vorige code. Enkel de SQL Statement verschilt:

```
Dim command As SqlCommand = New SqlCommand("SELECT * FROM
tasks WHERE SUBSTRING(Task,0,5)='" &
ddlProjects.SelectedValue.ToString & "'", connectieSCK)
        Dim reader As SqlDataReader = command.ExecuteReader()
```

In de WHERE clausule van de SQL Statement zie je dat de ik de eerste 5 karakters van de Task Code selecteer en deze vergelijk met de geselecteerde waarden van de DropDownList "ddlProjects".
Als laatste procedure hebben we de procedure "Apply Changes". Deze zorgt ervoor dat de eventuele wijzigingen worden opgeslagen en dat de Web Part wordt vernieuwd.

```
Public Overrides Sub ApplyChanges()
        MyBase.ApplyChanges()
```

```
        'De originele webpart aanspreken en de geselecteerde
waarde van de dropdown list "taken" naar de property [Text]
van de webpart sturen
        Dim wp As COCN = Me.ParentToolPane.SelectedWebPart
        wp.Text = ddlTasks.SelectedValue
    End Sub
```

5.2.2.2 HideListToolPart

Ook dit is een ToolPart klasse om de hoofdcode aan te sturen. Ze zorgt ervoor dat men eventueel bepaalde documenten kan verbergen uit de tabel.
Deze klasse maakt gebruikt van een CheckBoxList. Deze List geeft alle documenten weer met een CheckBox vooraan. Als men de CheckBox uitvinkt, mag dit document niet worden weergegeven in de gridview.
Om te weten welke documenten er zich bevinden in de gridview op de WebPart, heb ik in de hoofdcode 2 datatables aangemaakt. Een publieke DataTable om alle documenten in te bewaren. En een Private DataTable om de documenten in te bewaren die getoond moeten worden.
De HideTable wordt opgevuld met elk document en een verwijzing die aanduid of het document getoond moet worden of niet. Een item in de HideTable ziet er als volgt uit: "Reactor Model:True". De string wordt opgedeeld in 2 delen. Namelijk ID van het document ("Reactor Model"), gevolgd door een dubbel punt en de verwijzing of het document getoond moet worden of niet("True").
Door deze HideTable te doorlopen en de waarde na het dubbel punt op te vragen, weet ik dus welke records ik moet verwijderen uit de Private DataTable om ze te verbergen.
De procedure "New" bevat enkel de instellingen voor de titel van de toolpart en het label.

```
Public Sub New()
        Me.Title = "Hide / Show records"
        spacer.Text = "<br/>Documents selected will be shown
in the public gridview<br/><br/>"
 End Sub
```

CreateChildControls wordt weer overschreven & vult de CheckBoxList op. Daarna wordt de HideTable doorlopen en vinkt hij voor elk item de CheckBox uit waarbij er een waarde "False" na het dubbelpunt komt.

```
Protected Overrides Sub CreateChildControls()
        MyBase.CreateChildControls()
        'MyBase.EnsureChildControls()
        Dim BaseWebPart As COCN =
Me.ParentToolPane.SelectedWebPart
        Dim item As ListItem

        DocumentList.DataSource = BaseWebPart.pubdv
        DocumentList.DataTextField = "Title"
        DocumentList.DataValueField = "ID"
        DocumentList.DataBind()

        If BaseWebPart.HideTable.Count = 0 Then
            For Each item In DocumentList.Items
                item.Selected = True
```

```
            Next
        End If

        For Each Str As String In BaseWebPart.HideTable
            item =
DocumentList.Items.FindByValue(Str.Substring(0,
Str.IndexOf(":")))
            If Str.Substring(Str.IndexOf(":") + 1) = "False"
Then
                item.Selected = False
            Else
                item.Selected = True
            End If
        Next


        Me.Controls.Add(spacer)
        Me.Controls.Add(DocumentList)
    End Sub
```

De Procedure ApplyChanges maakt de HideTable leeg en vult ze terug op met de nieuwe waarden.

```
Public Overrides Sub ApplyChanges()
        MyBase.ApplyChanges()
        Dim BaseWebPart As COCN =
Me.ParentToolPane.SelectedWebPart

        Dim strVisible As String

        BaseWebPart.HideTable.Clear()

        For Each item As ListItem In DocumentList.Items
            If item.Selected = True Then
                strVisible = "True"
            Else
                strVisible = "False"
            End If

            BaseWebPart.HideTable.Add(item.Value.ToString &
":" & strVisible)
        Next
    End Sub
```

5.2.2.3 ToolParts toevoegen aan de Web Part

Nadat deze ToolParts geschreven zijn, moeten ze nog toegevoegd worden aan de hoofdcode. Dit wordt gerealiseerd door de Functie "GetToolParts" te overschrijven:

```
Public Overrides Function GetToolParts() As ToolPart()
        Dim toolParts(4) As ToolPart 'Array van de toolparts
```

```
        'toolpart met de algemene definities in
        Dim wptp As WebPartToolPart = New WebPartToolPart()
        'onze zelfgemaakte toolpart waarin we de SQL
commando's kunnen ingeven
        Dim custom As CustomPropertyToolPart = New
CustomPropertyToolPart()
        'de toolpart met de zelfgeschreven comboboxen in.
        Dim DataToolPart As New DropDownListToolPart
        Dim HideToolPart As New HideListToolPart

        'Toevoegen aan de array van toolparts in de juiste
volgorde
        toolParts(0) = DataToolPart
        toolParts(1) = HideToolPart
        toolParts(2) = custom
        toolParts(3) = wptp

        'terug sturen
        Return toolParts
    End Function
```

Er worden uiteindelijk 4 ToolParts weergeven:

- DataToolPart: Bevat de DropDownListToolPart
- HideToolPart: Bevat de HideListToolPart
- Custom: Custom Property's van Microsoft om bijvoorbeeld de hoogte van de WebPart in te stellen
- Wptp: Custom Property's van Microsoft om bijvoorbeeld de titel van WebPart in te stellen

5.2.3 CreateChildControls

De opbouw van het Default label kan je vinden in de Gemeenschappelijke Code. Hier wordt enkel de opbouw van de SPGridView beschreven.

5.2.3.1 CreateGrid

In deze procedure wordt dus de SPGridView aangemaakt en opgevuld. De DataTable wordt van een DataView voorzien en op deze DataView worden bewerkingen gemaakt zodat alles correct wordt weergegeven in de SPGridView.
Als eerst wordt gecheckt welke source men vraagt. Hiermee bedoel ik, als de SQL-Statement in de ToolPart is ingevuld, dan wordt de Taak Code buiten beschouwing gelaten.

```
If SQLFromToolPart.Equals("") Then
      SQL = "SELECT * FROM [Table] WHERE [Field] = '" &
[Text] & " '"
Else
      SQL = SQLFromToolPart
End If
```

Nadien worden de 2 DataTables (publiek in privé) aangemaakt volgens de manier die wordt beschreven in de Gemeenschappelijke Code (paragraaf 5.1)
Vervolgens worden nieuwe kolommen in de Private DataTable aangemaakt, en worden deze waarden opgevuld.

```
'Custom kolommen toevoegen aan de dataview
            dv.Table.Columns.Add("CustomRevision")
            dv.Table.Columns.Add("CustomID")
            dv.Table.Columns.Add("CustomTC")

            'dataview doorlopen
            For Each dr In dv
                'Als er meerdere revisies van een document
zijn, word de kolom customRevision opgevuld met de REV nr en
een slash voor de opbouw van de link
                If dr.Item("Revision").ToString = "REV0" Then
                    dr.Item("CustomRevision") = ""
                Else
                    dr.Item("CustomRevision") =
dr.Item("Revision").ToString & "/"
                End If
                'Als het ID kleiner is dan 2 karakters (vb: 2
of 5) dan wordt dit aangevult met een "0"
                If dr.Item("ID").ToString.Length < 2 Then
                    dr.Item("CustomID") = "0" &
dr.Item("ID").ToString
                Else
                    dr.Item("CustomID") =
dr.Item("ID").ToString
                End If

                'de link naar het document juist schrijven
                dr.Item("CustomTC") = "R" & dr.Item("Task
Code").ToString & Extension

                'items die niet mogen gezien worden,
verwijderen
                For Each item As String In HideTable
                    If dr.Item("ID").ToString =
item.Substring(0, item.IndexOf(":")) And
item.Substring(item.IndexOf(":") + 1) = "False" Then
                        dr.Delete()
                    End If
                Next

            Next
            'aanpassingen doorvoeren naar de dataTable
          dv.Table.AcceptChanges()
```

De nieuwe kolom "CustomRevision" zal waarden bevatten die nodig zijn om de link op te bouwen. Wanneer een document geen andere versie heeft, wordt deze gewoon leeg gelaten. Anderzijds, als er wel meerdere versies zijn, zal deze kolom een waarde krijgen in de vorm van: "02/".

Als tweede wordt het ID aangepast. Als de lengte van dit getal kleiner is dan 2, dus bijvoorbeeld 2 of 5, dan word er voor het getal een 0 geplaatst om de mappenstructuur te volgen.
"CustomTC" gaat de naam van het bestand bevatten zoals het op de FileServer staat, beginnende met een "R", dan de taak code en als laatste de extensie.
Nadien wordt er gekeken of er in de HideTable records staan die het zelfde ID hebben als een ID in de DataView en waarbij de waarde op "False" staat na het dubbelpunt. Als dit zo is, worden deze records uit de DataView verwijderd.
Als de DataView is opgebouwd, worden deSPBoundFields aangemaakt. Een SPBoundField kan je vergelijken met een kolom uit de SPGridView. Volgende SPBoundFields worden aangemaakt op de manier die staat beschreven in de Gemeenschappelijke code (paragraaf 5.1).

- Task Code
- Revision
- Author
- Date
- ID
- Title

Om de gebruikers de mogelijkheid te geven om een document te kunnen downloaden, wordt van "Titel" geen SPBoundField gemaakt, maar wel een SPMenuField. Een SPMenuField kan je ook vergelijken met een kolom uit een SPGridView, alleen krijgt deze kolom de vorm van een menu. Als je dit menu opent kan je op een MenuItem klikken en wordt het document gedownload. De code om een SPMenuField aan te maken is als volgt:

```
            Dim bf5 As New SPMenuField
          Dim titleMenu As New MenuTemplate
          Dim mnuDownloadFile As New MenuItemTemplate("Download this
     file", "_layouts\images\download.gif")
          With bf5
                .HeaderText = "Title"
                .TextFields = "Title"
                .SortExpression = "Title"
                .MenuTemplateId = "TitleMenu"
                .TokenNameAndValueFields = "TaskCode=Task
Code,IDNumber=CustomID,Rev=CustomRevision,CTC=CustomTC"
            End With

            'Property's instellen voor het MenuItemTemplate
            titleMenu.ID = "TitleMenu"
            mnuDownloadFile.ClientOnClickNavigateUrl =
"File://*******/Calculation/Brknotas/%TaskCode%/%IDNumber%/%Rev%%CTC%"
            'Menuitem toevoegen aan menu, menu toevoegen aan de
gridview
            titleMenu.Controls.Add(mnuDownloadFile)

            'menu toevoegen aan de web part
            Me.Controls.Add(titleMenu)
```

BF5 is van opbouw net hetzelfde als een SPBoundField, alleen krijgt men hier de mogelijkheid om een TokenNameAndValueFields eigenschap in te stellen. In deze

eigenschap worden kolommen uit de DataView ingevuld met een variabele achter. Deze variabele kan nadien gebruikt worden in de .ClientOnClickNavigateUrl eigenschap van het MenuItemTemplate. De waarden zullen dan per record juist ingevuld worden.
Vervolgens wordt de SPGridView aangemaakt en opgevuld volgens de manier die staat beschreven in de Gemeenschappelijke code (paragraaf 5.1).
Als laatste worden de connecties met de database gesloten.

```
'connectie sluiten
db.CloseConnection()
```

5.2.3.2 CreateLink

Er wordt een link aangemaakt die de gebruikt voorziet van een Help pagina

```
Public Sub CreateHelpLink()
        Dim link As New HyperLink

        With link
            .NavigateUrl =
"~/_layouts/CustomHelpFiles/COCN_help.htm"
            .Target = "_blank"
            .Text = "Help"
        End With
        Me.Controls.Add(link)
    End Sub
```

5.3 DEO Mechanical Drawings – DEO Instrumentation Drawings

Net zoals in het vorige hoofdstuk, wordt ook hier de code opgedeeld. Als eerst de verschillende property's die gebruikt worden, daarna de toolparts en tenslotte de CreateChildControls procedure.

5.3.1 Property's

Eerst en vooral worden er property's aangemaakt. Deze property's zorgen ervoor dat de code uit de algemene klasse kan worden aangestuurd door de ToolPart klassen.

```
'Declaratie van de de variabelen en een constante begin
waarde plaatsen
    Private Const _defaultText As String = "Default"
    Private Const _defaultSQL As String = ""
    Private Const _defaultPDF As String =
"\\Scksrv1\sckcen\projects"
    Private Const _defaultTIF As String =
"\\Scksrv1\sckcen\projects"
    Private Const _defaultDWF As String =
"\\Scksrv1\sckcen\projects"

'declaratie van de property's
     Private _text As String = _defaultText
    Private _customSQL As String = _defaultSQL
    Private _PDF As String = _defaultPDF
    Private _TIF As String = _defaultTIF
```

```
    Private _DWF As String = _defaultDWF

<Browsable(False), Category("Miscellaneous"),
DefaultValue(_defaultText), WebPartStorage(Storage.Personal),
FriendlyName("Insert Task Code"), Description("Text
Property")> Property [Text]() As String
        Get
            Return _text
        End Get

        Set(ByVal Value As String)
            _text = Value
        End Set
    End Property
    <Browsable(True), Category("Input Folders"),
DefaultValue(_defaultPDF), WebPartStorage(Storage.Personal),
FriendlyName("Select the folder where your PDF files are
located"), Description("Folder property")> Property PDF() As
String
        Get
            Return _PDF
        End Get

        Set(ByVal Value As String)
            _PDF = Value
        End Set
    End Property
    <Browsable(True), Category("Input Folders"),
DefaultValue(_defaultTIF), WebPartStorage(Storage.Personal),
FriendlyName("Select the folder where your TIF files are
located"), Description("Folder property")> Property TIF() As
String
        Get
            Return _TIF
        End Get

        Set(ByVal Value As String)
            _TIF = Value
        End Set
    End Property
    <Browsable(True), Category("Input Folders"),
DefaultValue(_defaultDWF), WebPartStorage(Storage.Personal),
FriendlyName("Select the folder where your DWF files are
located"), Description("Folder property")> Property DWF() As
String
        Get
            Return _DWF
        End Get

        Set(ByVal Value As String)
            _DWF = Value
        End Set
```

```
    End Property
    <Browsable(True), Category("SQL Statement"),
DefaultValue(_defaultSQL), WebPartStorage(Storage.Personal),
FriendlyName("Insert SQL Statement"), Description("Sql
Statement")> Property SQLFromToolPart() As String
        Get
            Return _customSQL
        End Get

        Set(ByVal Value As String)
            _customSQL = Value
        End Set
    End Property
```

Hier worden 5 property's aangemaakt. De property's PDF,TIF & DWF dienen om de gebruiker tekstvakken te geven om hun folders voor de betreffende bestanden op te geven. Text bevat ook hier de waarde van de gekozen Task Code. Als laatste property is er "SQLFromToolPart" die de gebruiker weer voorziet om een SQL query in te geven.

5.3.2 ToolParts

In de ToolPart kan men eigenschappen ingeven die de weergave van de Web Part beïnvloeden. Om extra eigenschappen beschikbaar te maken, moeten ook deze zelf worden aangemaakt. ToolParts zijn klassen die worden gedeclareerd in de hoofdcode.
Omdat de andere ToolPart eigenschappen automatisch door SharePoint worden aangemaakt, hebben we maar 1 zelfgemaakte ToolPart nodig, namelijk de DropDownListToolPart

5.3.2.1 DropDownListToolPart

Deze code is identiek aan de DropDownListToolPart in de CO Web Parts. Meer informatie hierover kan je dus vinden in paragraaf 5.2.2.1

5.3.2.2 ToolPart toevoegen aan Web Part

Nadat deze ToolParts geschreven zijn, moeten ze nog toegevoegd worden aan de hoofdcode. Dit wordt gerealiseerd door de Functie "GetToolParts" te overschrijven:

```
Public Overrides Function GetToolParts() As ToolPart()
        Dim toolParts(4) As ToolPart 'Array van de toolparts

        'toolpart met de algemene definities in
        Dim wptp As WebPartToolPart = New WebPartToolPart()
        'onze zelfgemaakte toolpart waarin we de SQL
commando's kunnen ingeven
        Dim custom As CustomPropertyToolPart = New
CustomPropertyToolPart()
        'de toolpart met de zelfgeschreven comboboxen in.
        Dim DataToolPart As New DropDownListToolPart

        'Toevoegen aan de array van toolparts in de juiste
volgorde
        toolParts(0) = DataToolPart
        toolParts(1) = custom
        toolParts(2) = wptp
        'terug sturen
```

```
        Return toolParts
    End Function
```

Er worden uiteindelijk 3 ToolParts weergeven:

- DataToolPart: Bevat de DropDownListToolPart
- Custom: Custom Property's van Microsoft om bijvoorbeeld de hoogte van de WebPart in te stellen
- Wptp: Custom Property's van Microsoft om bijvoorbeeld de titel van WebPart in te stellen

5.3.3 CreateChildControls

Net zoals bij de ToolParts moet hier ook de procedure "CreateChildControls" overschreven worden om de gridview op het scherm te krijgen.

```
 'Om andere controls weer te geven moeten we de
CreateChildControls() procedure overschrijven
    Protected Overrides Sub CreateChildControls()
        MyBase.CreateChildControls()
        'Procedure oproepen om het gridview aan te maken
        If Text = "Default" Then
            CreateDefaultLabel()
        Else
          ReadDirectory(PDF, "pdf", PDFList)
            ReadDirectory(TIF, "tif", TIFList)
            ReadDirectory(DWF, "dwf", DWFList)
            CreateGrid()

        End If
        CreateHelpLink()
    End Sub
```

Je ziet dat hier een extra procedure wordt opgeroepen vooralleer de SPGridView wordt aangemaakt.

5.3.3.2 ReadDirectory

Vooraleer de SPGridView wordt aangemaakt, worden eerst de opgegeven mappen uitgelezen en worden de bestanden in een List gestopt. Parameters die meegegeven worden zijn:

- Path: bevat het pad waar het bestand staat,
- Extension: bevat de extensie waarop gefilterd moet worden,
- List: bevat de List die wordt opgevuld

```
Public Sub ReadDirectory(ByVal path As String, ByVal
extension As String, ByRef List As List(Of String))
```

Om er voor te zorgen dat ook SharePoint Document Lists gebruikt kunnen worden, wordt er eerst gekeken of de eerste 2 karakters van de folder die de gebruiker opgeeft, gelijk is aan

"\\". Als dit niet zo is word de SharePoint Document List, waarvan de gebruiker de naam op geeft, doorlopen en worden de waarden in de list gestopt.

```
If Left(path, 2) <> "\\" Then
            Try
                Dim web As SPWeb =
SPControl.GetContextWeb(Context)
                Dim myFolder As SPFolder =
web.GetFolder(path)
                Dim myFiles As SPFileCollection =
myFolder.Files

                For Each myFile As SPFile In myFiles
                    If Right(myFile.Name, 3) = extension Then
                        List.Add(web.Url.ToString &
myFile.ServerRelativeUrl)
                    End If
                Next
            Catch ex As Exception
                Dim lbl As New Label
                lbl.Text = "Document List Error occured: " &
ex.Message
            End Try
```

Als de 2 eerste karakters wel gelijk zijn aan "\\" wordt de opgegeven folder doorlopen en worden de waarden in de list gestopt.

```
    Else
            'nakijken of het laatste karakter van path een \
is, als dit niet is word deze erbij gevoegd.
            If Not Right(path, 1) = "\" Then
                path += "\"
            End If
            'Deze procedurce leest de directory op het
aangegeven pad & stuurt een list met de link
            'naar deze bestanden in terug
            Try
                Dim di As New DirectoryInfo(path)
                Dim aryFi As FileInfo() = di.GetFiles("*." &
extension)
                Dim fi As FileInfo

                For Each fi In aryFi
                    List.Add("File:" &
fi.FullName.Replace("\", "/"))
                Next
            Catch ex As Exception
                Dim lbl As New Label
                lbl.Text = "File Error occured: " &
ex.Message
            End Try
        End If
```

```
    End Sub
```

5.3.3.3 CreateGrid

In deze procedure wordt dus de SPGridView aangemaakt en opgevuld. De DataTable wordt van een DataView voorzien en op deze DataView worden bewerkingen gemaakt zodat alles correct wordt weergegeven in de SPGridView.
Als eerste stap wordt er eerst gekeken of de SQLFromToolPart een waarde bevat. Als dit zo is, wordt deze waarde gebruikt om de SQL string op te bouwen.

```
If SQLFromToolPart.Equals("") Then
      SQL = "SELECT * FROM [Table] WHERE [Field] = '" &
[Text] & " '"
Else
      SQL = SQLFromToolPart
End If
```

Nadien worden de DataTable aangemaakt volgens de manier die wordt beschreven in de Gemeenschappelijke Code (paragraaf 5.1). Vervolgens worden nieuwe kolommen in de DataTable aangemaakt, en worden deze waarden opgevuld.

```
'zelf kolommen toevoegen
       dv.Table.Columns.Add("DWGTextField")
       dv.Table.Columns.Add("DWGFileField")
       dv.Table.Columns.Add("PDFTextField")
       dv.Table.Columns.Add("PDFFileField")
       dv.Table.Columns.Add("TIFTextField")
       dv.Table.Columns.Add("TIFFileField")
       dv.Table.Columns.Add("DWFTextField")
      dv.Table.Columns.Add("DWFFileField")

     'voor elke rij in de database word volgende regels code
uitgevoerd
     'Er word vergeleken tussen de naam van het bestand in de
     database & de naam van het bestand op de aangegeven
     directory
     'indien deze hetzelfde zijn, word de link naar het
     bestand in de dataview weggeschreven
            For Each dr As DataRowView In dv
                'enkel de naam (zonder folders / extensie)
ophalen
                Dim url As String = dr.Item("File")
                Dim aStr() As String
                Dim filename As String

                url = url.Replace("\", "/")
                aStr = url.Split("/")
                filename = aStr(aStr.Length - 1)
                filename = filename.Substring(0,
filename.IndexOf("."))
```

```
                'DWG Files toevoegen aan de dataset
                If Not filename.Equals("") Then
                    dr.Item("DWGTextField") = "DWG"
                End If
                dr.Item("DWGFileField") =
dr.Item("File").ToString.Replace("\", "/").Replace("M:",
"//admsrv1/Tekening")
                'PDF Files toevoegen aan de dataset
                For Each f As String In PDFList
                    If f.Substring(f.LastIndexOf("/") + 1,
f.IndexOf(".") - f.LastIndexOf("/") - 1) = filename And
filename <> "" Then
                        If Left(url, 2) <> "\\" Then
                            dr.Item("PDFFileField") =
f.Replace("\", "/")
                        Else
                            dr.Item("PDFFileField") =
f.Replace("\", "/").Replace("&", "and")
                        End If
                        dr.Item("PDFTextField") = "PDF"
                    End If
                Next
                'TIF files toevoegen aan dataset
                For Each f As String In TIFList
                    If f.Substring(f.LastIndexOf("/") + 1,
f.IndexOf(".") - f.LastIndexOf("/") - 1) = filename And
filename <> "" Then
                        If Left(url, 2) <> "\\" Then
                            dr.Item("TIFFileField") =
f.Replace("\", "/")
                        Else
                            dr.Item("TIFFileField") =
f.Replace("\", "/").Replace("&", "and")
                        End If
                        dr.Item("TIFTextField") = "TIF"
                    End If
                Next
                'DWF files toevoegen aan dataset
                For Each f As String In DWFList
                    If f.Substring(f.LastIndexOf("/") + 1,
f.IndexOf(".") - f.LastIndexOf("/") - 1) = filename And
filename <> "" Then
                        If Left(url, 2) <> "\\" Then
                            dr.Item("DWFFileField") =
f.Replace("\", "/")
                        Else
                            dr.Item("DWFFileField") =
f.Replace("\", "/").Replace("&", "and")
                        End If
                        dr.Item("DWFTextField") = "DWF"
                    End If
```

```
                    Next
            Next
```

Als eerste word de filename, zonder extensie uit de database gehaald. Dit is de referentie om te vergelijken of er gegevens zijn in de custom folders die bij een bepaald DWG bestand horen.
Nadien worden de DWG gegevens aangepast. In de database staat de drive letter van een gedeelde map op het netwerk (M:) deze moet vervangen worden door de juiste map waarin deze bestanden staan.
De volgende "For Each" structuren vullen de kolommen" TextField" en " FieldField" aan. "TextField" gaat niets bevatten als er geen bestand gevonden is dat hetzelfde is als de filename in de database. De kolommen "FileField" gaan de juiste link bevatten naar de plaats waar het bestand staat.
Als de DataView is opgebouwd, worden deSPBoundFields aangemaakt. Een SPBoundField kan je vergelijken met een kolom uit de SPGridView. Volgende SPBoundFields worden aangemaakt op de manier die staat beschreven in de Gemeenschappelijke code (paragraaf 5.1)**.** Volgende kolommen zullen verschijnen in de gridview:

- Titel 1
- Titel 2
- Drawing Number
- Index
- DWG
- DWF
- PDF
- TIF

Om de gebruikers de mogelijkheid te geven om een document te kunnen downloaden, wordt van "DWG", "DWF", "PDF" en "TIF" geen SPBoundField gemaakt, maar wel een SPMenuField. Een SPMenuField kan je ook vergelijken met een kolom uit een SPGridView alleen krijgt deze kolom de vorm van een menu. Als je dit menu opent kan je op een MenuItem klikken en wordt het document gedownload. De code om een SPMenuField aan te maken is als volgt:

```
Dim bfdwg, bfdwf, bftif, bfpdf As New SPMenuField
Dim mnuDwg, mnuPdf, mnuTif, mnuDwf As New MenuTemplate
Dim mnuOpenDwg As New MenuItemTemplate("Open",
"_layouts\images\open.gif")
With bfdwg
   .TextFields = "DWGTextField"
   .MenuTemplateId = "mnuOpenInst"
    .TokenNameAndValueFields = "dwgfile=DWGFileField"
    .ToolTipFields = "DWGFileField"
End With
mnuDwg.ID = "mnuOpenInst"
mnuOpenDwg.ClientOnClickNavigateUrl = "File:%dwgfile%"
mnuDwg.Controls.Add(mnuOpenDwg)
```

"bfdwg" is van opbouw net hetzelfde als een SPBoundField, alleen krijgt men hier de mogelijkheid om een TokenNameAndValueFields eigenschap in te stellen. In deze

eigenschap worden kolommen uit de DataView ingevuld met een variabele achter. Deze variabele kan nadien gebruikt worden in de .ClientOnClickNavigateUrl eigenschap van het MenuItemTemplate. De waarden zullen dan per record juist ingevuld worden.
Dit wordt gedaan voor zowel DWG, DWF, PDF en TIF bestanden.
Vervolgens wordt de SPGridView aangemaakt en opgevuld volgens de manier die staat beschreven in de Gemeenschappelijk code (paragraaf 5.1).

5.3.3.4 CreateLinks
Naast het gewonde Help hyperlink, bevat deze Web Part nog 2 extra hyperlinks. Deze hyperlinks bevatten een link naar het installatiebestanden om de DWF Viewer en Autocad Design Review (om DWG bestanden te bekijken) te installeren.
Omdat deze installatiebestanden op later tijdstip kunnen verplaatst of geüpdatet worden, heeft een vorige stagiair een webservice geschreven die de juiste link terug geeft. Ik spreek deze webservice dus aan om de link te krijgen.

```
Public Sub CreateLinks()
        'Declaratie van de webservice en de hyperlinks
        Dim webservice As New be.sck.intern.UpgradeServer
        Dim dwfLink, dwgLink As String
        Dim lnkHelp As New HyperLink
        Dim lnkDWF As New HyperLink
        Dim lnkDWG As New HyperLink
        Dim spacer As New Label
        Dim spacer2 As New Label

        'webservice aanroepen & de link naar de installatie
bestanden in een variabele wegschrijven
        dwfLink = webservice.checkVersionXML("Autodesk Design
Review", "").ToString
        dwgLink = webservice.checkVersionXML("DWG True View",
"").ToString

        'spacers instellen
        spacer2.Text = " - "
        spacer.Text = " - "

        'links instellen
        lnkHelp.Text = "<br/><br/>Help"
        lnkHelp.NavigateUrl =
"~/_layouts/CustomHelpFiles/DEO_Inst_help.htm"
        lnkHelp.Target = "_blank"
        lnkDWF.Text = "Download DWF Viewer"
        lnkDWF.NavigateUrl = dwfLink
        lnkDWG.Text = "Download DWG Viewer"
        lnkDWG.NavigateUrl = dwgLink

        'links & spacers toevegen aan scherm
        Me.Controls.Add(lnkHelp)
        Me.Controls.Add(spacer2)
        Me.Controls.Add(lnkDWF)
        Me.Controls.Add(spacer)
        Me.Controls.Add(lnkDWG)
```

```
    End Sub
```

5.4 BR2 Drawings

Net zoals in het vorige hoofdstuk, wordt ook hier de code opgedeeld. Als eerst de verschillende property's die gebruikt worden, daarna de toolparts en tenslotte de CreateChildControls procedure.

5.4.1 Property's

Ook hier hebben we eerst de property's die zorgen voor communicatie tussen de ToolPart en de hoofdklasse.

```
'Declaratie van de de variabelen en een constante begin
waarde plaatsen
    Private Const _defaultTaaknummer As String = ""
    Private Const _defaultDossiernr As String = "X210"
    Private Const _defaultSource As String = "Default"

'declaratie van de property's
    Dim _taaknummer As String = _defaultTaaknummer
    Dim _dossiernr As String = _defaultDossiernr
    Dim _source As String = _defaultSource

<Browsable(False), Category("Miscellaneous"),
DefaultValue(_defaultTaaknummer),
WebPartStorage(Storage.Personal), FriendlyName("Taaknummer"),
Description("Taaknummer")> _
    Property Taaknummer() As String
        Get
            Return _taaknummer
        End Get

        Set(ByVal Value As String)
            _taaknummer = Value
        End Set
    End Property

    <Browsable(False), Category("Miscellaneous"),
DefaultValue(_defaultDossiernr),
WebPartStorage(Storage.Personal),
FriendlyName("Dossiernummer"), Description("Dossiernummer")>
_
   Property Dossiernr() As String
        Get
            Return _dossiernr
        End Get

        Set(ByVal Value As String)
            _dossiernr = Value
        End Set
    End Property
    <Browsable(False), Category("Miscellaneous"),
```

```
DefaultValue(_defaultSource),
WebPartStorage(Storage.Personal), FriendlyName("Source"),
Description("Source")> _
  Property Source() As String
        Get
            Return _source
        End Get

        Set(ByVal Value As String)
            _source = Value
        End Set
    End Property
```

De 3 property's die worden aangemaakt dienen enkel en alleen om te kijken welke data men moet ophalen. "Taaknummer" zorgt voor de taak nummer die wordt opgegeven, "Dossiernummer" voor het eventuele dossiernummer. "Source" bevat de keuze die de gebruiker maakt welke van de 2 bovenstaande hij wil gebruiken.

5.4.2 ToolParts

In de ToolPart kan men eigenschappen ingeven die de weergave van de Web Part beïnvloeden. Om extra eigenschappen beschikbaar te maken, moeten ook deze zelf worden aangemaakt. ToolParts zijn klassen die worden gedeclareerd in de hoofdcode.
In deze Web Part hebben we maar 1 ToolPart nodig, namelijk de CriteriaToolPart

5.4.2.1 CriteriaToolPart

De CriteriaToolPart neemt de data keuze voor zich. Het zorgt voor 2 comboboxen en 2 radiobuttons. De comboboxen worden opgevuld met de Taaknummers en de dossiernummers beginnende met een "X" (voor experimenten). De Radiobutton zorgen ervoor dat de gebruiker kan kiezen of hij het taak- of dossiernummer wil gebruiken.
In de methode New (die wordt aangeroepen als de ToolPart word geladen) worden alle controls geïnstancieerd:

```
Public Sub New()
        'Titel van de toolpart instellen
        Me.Title = "Gridview Source"
        'Text instellen voor de labels
        lblTaaknummer.Text = "Select a Tasknumber:<br/>"
        lblDossiernr.Text = "<br/><br/>Select a
dossiernumber:<br/>"
        spacer.Text = "<br/><br/>"
        lblRadio.Text = "<br/>Select the source to use:<br/>"
        rbList.Items.Add("Use Tasknumber")
        rbList.Items.Add("Use Dossiernumber")
        spacer2.Text = "-----------------------<br/>"
    End Sub
```

In de CreateChildControls procedure worden de standaard waarden opgevuld en worden de controls toegevoegd op het scherm:

```
Protected Overrides Sub CreateChildControls()
```

```
        MyBase.CreateChildControls()
        'huidige webpart declareren
        Dim BaseWebPart As BR2_Drawings =
Me.ParentToolPane.SelectedWebPart
        'standaard waarde van de radiobuttons instellen
        If BaseWebPart.Source = "Dossier" Then
            rbList.SelectedValue = "Use Dossiernumber"
        Else
            rbList.SelectedValue = "Use Tasknumber"
        End If
        'procedure aanroepen om comboboxen op te vullen
        ComboBoxVullen(ddlTaaknummer, "Taaknummer",
BaseWebPart.Taaknummer.ToString)
        ComboBoxVullen(ddlDossiernr, "Dossiersnummer",
BaseWebPart.Dossiernr.ToString)
        'controls toevoegen
        Me.Controls.Add(lblRadio)
        Me.Controls.Add(rbList)
        Me.Controls.Add(spacer)
        Me.Controls.Add(spacer2)
        Me.Controls.Add(lblTaaknummer)
        Me.Controls.Add(ddlTaaknummer)
        Me.Controls.Add(lblDossiernr)
        Me.Controls.Add(ddlDossiernr)
        Me.Controls.Add(spacer)
    End Sub
```

Je ziet dat de procedure "ComboBoxVullen" wordt aangeroepen, ze zorgt ervoor dat de data uit de database wordt gelezen en dat de DropDownLists worden opgevuld. Deze procedure heeft enkele parameters:

- DropDownList: hier wordt de combobox meegegeven die wordt opgevuld.
- sqlDataField: maakt onderscheid tussen taaknummer & dossiernummer.
- DefaultValue: zet de standaardwaarde van de DropDownList.

```
Public Sub ComboBoxVullen(ByVal DropDownList As DropDownList,
ByVal sqlDataField As String, ByVal DefaultValue As String)
        'declaratie
        Dim connectieStringSCK As String =
"********************"
        Dim connectieSCK As OleDbConnection = New
OleDbConnection(connectieStringSCK)

        'connectie openen
        connectieSCK.Open()

        'commando & reader aanmaken, uitvoeren
        Dim command As OleDbCommand
        'indien men via het dossiernummer werkt, moeten de
nummers beginnen met een X
        If sqlDataField = "Taaknummer" Then
            command = New OleDbCommand("SELECT DISTINCT " &
sqlDataField & " FROM [tblplannen] WHERE " & sqlDataField & "
```

```
IS NOT NULL AND NOT " & sqlDataField & "='' AND NOT " &
sqlDataField & "='ONBEKEND' ORDER BY " & sqlDataField & "
ASC", connectieSCK)
        Else
            command = New OleDbCommand("SELECT DISTINCT " &
sqlDataField & " FROM [tblplannen] WHERE " & sqlDataField & "
IS NOT NULL AND NOT " & sqlDataField & "='' AND NOT " &
sqlDataField & "='ONBEKEND' AND " & sqlDataField & " LIKE
'X%' ORDER BY " & sqlDataField & " ASC", connectieSCK)
        End If
        Dim reader As OleDbDataReader =
Command.ExecuteReader()

        'dropdownlist opvullen
        DropDownList.DataSource = reader
        DropDownList.DataValueField = sqlDataField
        DropDownList.DataTextField = sqlDataField
        DropDownList.DataBind()
        'standaard waarde instellen
        DropDownList.SelectedValue = DefaultValue

        'objecten sluiten
        reader.Close()
        connectieSCK.Close()
    End Sub
```

5.4.2.2 ToolParts toevoegen aan Web Part

Nadat deze ToolParts geschreven zijn, moeten ze nog toegevoegd worden aan de hoofdcode. Dit wordt gerealiseerd door de Functie "GetToolParts" te overschrijven:

```
Public Overrides Function GetToolParts() As ToolPart()
        Dim toolParts(4) As ToolPart 'Array van de toolparts

        'toolpart met de algemene definities in
        Dim wptp As WebPartToolPart = New WebPartToolPart()
        'onze zelfgemaakte toolpart waarin we de SQL
commando's kunnen ingeven
        Dim custom As CustomPropertyToolPart = New
CustomPropertyToolPart()
        'de toolpart met de zelfgeschreven comboboxen in.
        Dim CriteriaToolPart As New CriteriaToolPart

        'Toevoegen aan de array van toolparts in de juiste
volgorde
        toolParts(0) = CriteriaToolPart
        toolParts(1) = custom
        toolParts(2) = wptp
        'terug sturen
        Return toolParts
    End Function
```

Er worden uiteindelijk 3 ToolParts weergeven:

- CriteriaToolPart: Bevat de CriteriaToolPart
- Custom: Custom Property's van Microsoft om bijvoorbeeld de hoogte van de WebPart in te stellen
- Wptp: Custom Property's van Microsoft om bijvoorbeeld de titel van WebPart in te stellen

5.4.3 CreateChildControls

Net zoals in de vorige Web Parts wordt eerst gekeken of de taakcode op "Default" staat. Deze werkwijze kan je vinden in de Gemeenschappelijke Code.

5.4.3.1 CreateGrid

In deze procedure wordt dus de SPGridView aangemaakt en opgevuld. De DataTable wordt van een DataView voorzien en op deze DataView worden bewerkingen gemaakt zodat alles correct wordt weergegeven in de SPGridView.
Deze procedure lijkt zeer hard op degene van de CO Web Parts. Er is geen SQL statement mogelijkheid voorzien, maar toch kan de SQL beïnvloed worden door de keuze van de Radiobutton in de ToolPart.

```
'SQL instellen
If Source = "Dossier" Then
    SQL = "SELECT * FROM [tblplannen] WHERE
[Dossiersnummer]='" & Dossiernr & "'"
Else
    SQL = "SELECT * FROM [tblplannen] WHERE [Taaknummer]='" &
Taaknummer & "'"
End If
```

Hierna worden zoals op de vorige manieren de SPBoundFields en SPMenuFields ingesteld. Kolommen die gebruikt worden zijn:

- Plannummer
- Revisie
- Titel

5.5 Bidasse

Net zoals in het vorige hoofdstuk, wordt ook hier de code opgedeeld. Als eerst de verschillende property's die gebruikt worden, daarna de toolparts en tenslotte de CreateChildControls procedure.
Bidasse is heel wat gecompliceerder dan de vorige Web Parts. Er wordt gebruik gemaakt van binaire bestanden die uitgelezen moeten worden.

5.5.1 Property's

Ook hier hebben we eerst de property's die zorgen voor communicatie tussen de ToolPart en de hoofdklasse.

```
'Declaratie van de de variabelen en een constante begin
waarde plaatsen
    Private Const _defaultProject As String = "Default"
    Private Const _defaultTimer As Boolean = False

'declaratie van de property's
```

```
    Dim _project As String = _defaultProject
    Dim _timer As Boolean = _defaultTimer

<Browsable(False), Category("Miscellaneous"),
DefaultValue(_defaultProject),
WebPartStorage(Storage.Personal), FriendlyName("Project"),
Description("Project Property")> _
    Property Project() As String
        Get
            Return _project
        End Get

        Set(ByVal Value As String)
            _project = Value
        End Set
    End Property
    <Browsable(False), Category("Miscellaneous"),
DefaultValue(_defaultTimer),
WebPartStorage(Storage.Personal), FriendlyName("Timer"),
Description("Timer")> _
    Property Timer() As Boolean
        Get
            Return _timer
        End Get

        Set(ByVal Value As Boolean)
            _timer = Value
        End Set
    End Property
```

Bidasse gebruikt weinig communicatie met de ToolPart, daarom zijn er maar 2 property's aanwezig. De eerste is "Project" waarin het gekozen project (of experiment) geplaatst wordt. Als tweede hebben we "Timer", dit is een boolean. Als deze op "True" staat, word de timer gestart en wordt de Web Part elke minuut vernieuwd.

5.5.2 ToolParts

In de ToolPart kan men eigenschappen ingeven die de weergave van de Web Part beïnvloeden. Om extra eigenschappen beschikbaar te maken, moeten ook deze zelf worden aangemaakt. ToolParts zijn klassen die worden gedeclareerd in de hoofdcode.
Bidasse gebruikt 2 ToolParts, 1 voor het Project te kiezen, een andere om de Timer te configureren.

5.5.2.1 ProjectToolPart

Het enige waar de ProjectToolPart voor zorgt, is dat men een DropDownList krijgt waarin men een bepaald experiment kan kiezen.
De experimenten (of projecten) worden ingeladen door de mappenstructuur van Bidasse te bekijken. Elk project krijgt zijn eigen map, door een aantal vaste mappen (zoals instellingen mappen, bestandsmappen) weg te filteren kan ik alle experimenten opvragen.
In de methode New (die wordt aangeroepen als de ToolPart wordt geladen) worden alle controls geïnstancieerd:

```
Public Sub New()
        'Titel van toolpart instellen
        Me.Title = "Choose Project"
        'spacer instellen
        spacer.Text = "<br/>"
        'label instellen
        lblProjects.Text = "<br/>Choose your project:<br/>"
    End Sub
```

In de CreateChildControls procedure worden de standaard waarden opgevuld en worden de controls toegevoegd op het scherm:

```
Protected Overrides Sub CreateChildControls()
        MyBase.CreateChildControls()
        'procedure oproepen om dropdownlist op te vullen
        FillDropDownList()
        'standaard waarde invullen
        Dim BaseWebPart As BR2_Bidasse =
Me.ParentToolPane.SelectedWebPart
        ddlProjects.SelectedValue = BaseWebPart.Project
        'controls toevoegen aan scherm
        Me.Controls.Add(lblProjects)
        Me.Controls.Add(ddlProjects)
        Me.Controls.Add(spacer)
    End Sub
```

Je ziet dat de procedure "FillDropDownList" wordt aangeroepen, ze zorgt ervoor dat de mappenstructuur wordt gelezen en dat de DropDownLists worden opgevuld.

```
Private Sub FillDropDownList()
        Try
            ' Code snippet
            ' make a reference to a directory
            Dim di As New IO.DirectoryInfo(BidassePath)
            Dim diar1 As IO.DirectoryInfo() =
di.GetDirectories()
            Dim dra As IO.DirectoryInfo

            ddlProjects.Items.Clear()
            'list the names of all files in the specified
directory
            For Each dra In diar1
                Select Case dra.Name.ToLower
                    Case "calor", "install", "logbook",
"logitime", "process", "report", "test"
                        'Those directories have no
LASTDATA.DAT file
```

```
                    Case Else
                        ddlProjects.Items.Add(dra.Name)
                End Select
            Next
            ddlProjects.Text = "List Of Available
Experiments"
        Catch ex As Exception
            MsgBox(ex.Source & " - " & ex.Message,
MsgBoxStyle.Critical, "Service Controller Alert")
        End Try
    End Sub
```

De procedure ApplyChanges wordt aangeroepen als men op "Ok" klikt en voer de wijzigingen door naar de hoofdcode.

```
Public Overrides Sub ApplyChanges()
        'waarden naar de property van de webpart sturen
        MyBase.ApplyChanges()
        Dim wp As BR2_Bidasse =
Me.ParentToolPane.SelectedWebPart
        wp.Project = ddlProjects.SelectedValue.ToString
    End Sub
```

5.5.2.2 TimerToolPart

Dit is een simpele ToolPart die enkel een Checkbox weergeeft. Als deze gecheckt is, word de waarde "true" naar de hoofdcode gegeven, indien deze unchecked is, wordt de waarde "false" terug gegeven.

```
Public Sub New()
        'Titel van toolpart instellen
        Me.Title = "Timer settings"
        'spacer instellen
        spacer.Text = "<br/>"
        'label instellen
        lblProjects.Text = "<br/>Enable auto refresh: "
    End Sub

    Protected Overrides Sub CreateChildControls()
        MyBase.CreateChildControls()
        'standaard waarde invullen
        Dim BaseWebPart As BR2_Bidasse =
Me.ParentToolPane.SelectedWebPart
        If BaseWebPart.Timer = True Then
            chkTimer.Checked = True
        Else
            chkTimer.Checked = False
        End If
        'controls toevoegen aan scherm
        Me.Controls.Add(lblProjects)
        Me.Controls.Add(chkTimer)
        Me.Controls.Add(spacer)
```

```
    End Sub
    Public Overrides Sub ApplyChanges()
        'waarden naar de property van de webpart sturen
        MyBase.ApplyChanges()
        Dim wp As BR2_Bidasse =
Me.ParentToolPane.SelectedWebPart
        If chkTimer.Checked = True Then
            wp.Timer = True
        Else
            wp.Timer = False
        End If
    End Sub
```

5.5.2.3 ToolParts toevoegen aan Web Part

Nadat deze ToolParts geschreven zijn, moeten ze nog toegevoegd worden aan de hoofdcode. Dit wordt gerealiseerd door de Functie "GetToolParts" te overschrijven:

```
Public Overrides Function GetToolParts() As ToolPart()
        Dim toolParts(4) As ToolPart 'Array van de toolparts

        'toolpart met de algemene definities in
        Dim wptp As WebPartToolPart = New WebPartToolPart()
        'onze zelfgemaakte toolpart waarin we de SQL
commando's kunnen ingeven
        Dim custom As CustomPropertyToolPart = New
CustomPropertyToolPart()
        'de toolpart met de criteria in.
        Dim ProjectToolPart As New ProjectToolPart
        Dim TimerToolPart As New TimerToolPart

        'Toevoegen aan de array van toolparts in de juiste
volgorde
        toolParts(0) = ProjectToolPart
        toolParts(1) = TimerToolPart
        toolParts(2) = custom
        toolParts(3) = wptp

        'terug sturen
        Return toolParts
    End Function
```

Er worden uiteindelijk 4 ToolParts weergeven:

- ProjectToolPart: Bevat de ProjectToolPart
- TimerToolPart: Bevat de TimerToolPart
- Custom: Custom Property's van Microsoft om bijvoorbeeld de hoogte van de WebPart in te stellen
- Wptp: Custom Property's van Microsoft om bijvoorbeeld de titel van WebPart in te stellen

5.5.3 CreateChildControls

Ook hier wordt er eerst gekeken of de property "project" op default staat. Ik beschrijf dus enkel de procedure "CreateGrid".

5.5.3.1 CreateGrid

Om de juiste mappen van de .cha en lastdata.dat bestanden op te vragen, heb ik een webservice geschreven. Deze kan je vinden in 5.5.4
Eerst en vooral wordt de webservice aangesproken, en via de desbetreffende methodes de juiste bestanden opgehaald.

```
Dim websrv As New be.sck.intern.BidasseWebService
Dim LastDataPath As String =
websrv.GetLastDataFolder(Project).ToString
Dim ChannelsPath As String =
websrv.GetChaFile(Project).ToString
```

Als de Property "Timer" op True staat, word er een HTMLGenericControl toegevoegd. Deze Control zorgt ervoor dat er Javascript kan gebruikt worden in de code. Onderstaande code zorgt ervoor dat de pagina elke minuut vernieuwd wordt.

```
If Timer = True Then
   Dim timer As HtmlControls.HtmlGenericControl
   'Timer instellen & javascript implementeren
   timer = New HtmlControls.HtmlGenericControl("script")
   timer.Attributes.Add("language", "javascript")
   timer.InnerText = "setTimeout('refresh_location()',
1000*60);"
   timer.InnerText += "function
refresh_location(){location.reload(true);}"
   Me.Controls.Add(timer)
End If
```

In de volgende stap, wordt via de ondertussen vertrouwde wijze een DataTable en DataView aangemaakt waarin we zelf kolommen maken.

```
'Kolommen aan DataTable Toevoegen
dt.Columns.Add("naam")
dt.Columns.Add("waarde")
dt.Columns.Add("eenheid")

'Juiste LastData ophalen
GetLastDataFile(LastDataPath)
'Kanalen inlezen
ReadChannels(ChannelsPath, AantalKanalen
```

Nadien wordt het juist LASTDATA.DAT bestand opgehaald en worden de kanalen ingelezen via volgende functies:

- GetLastDataFile
- ReadChannels

GetLastDataFile is een functie die de experiment map doorloopt om te kijken of er meerdere LASTDATA.DAT bestanden instaan. Indien dit zo is, wordt het bestand genomen met het hoogste cijfer achteraan in de extensie. De Parameter die wordt meegegeven is de experiment map.

```
Public Sub GetLastDataFile(ByRef LastDataPath As String)
        Dim storefile As Directory

        Dim Files() As String
        Dim file As String

        Files = storefile.GetFiles(LastDataPath,
"LASTDATA.*")

        If Files.Length = 1 Then
            LastDataPath += "LASTDATA.DAT"
        Else
            For Each file In Files
                If Right(file, 1) = Files.Length.ToString
Then
                    LastDataPath = file
                End If
            Next
        End If
    End Sub
```

De tweede procedure die wordt aangeroepen is "ReadChannels". Deze procedure leest het kanalen bestand en stopt alle kanalen in een List. Dit is nodig omdat de eenheden en de namen van de waarden die in LASTDATA.DAT staan, niet kunnen terug vinden in LASTDATA.DAT. Het .cha is geen binair bestand, dus ik maak gebruikt van een StreamReader in plaats van een BinaryReader.

```
Public Sub ReadChannels(ByVal ChannelsPath As String, ByRef
AantalKanalen As Integer)
        Dim sr As New StreamReader(ChannelsPath)
        Dim line As String
        Dim array() As String
        Dim eenheid As String
        Dim naam As String

        AantalKanalen = 0

        Do
            line = sr.ReadLine
            array = line.Split(",")
            eenheid = array(6).ToString
            naam = array(3).ToString
            naamList.Add(naam)
            eenheidList.Add(eenheid)
            AantalKanalen += 1
        Loop Until sr.EndOfStream
```

```
    End Sub
```

Nadat alle kanalen zijn ingelezen, wordt het LASTDATA bestand uitgelezen. Elk LASTDATA bestand bevat een hoofding die exact 32 bytes groot is. Deze hoofding bevat de naam van het experiment, en de datum waarop deze meting is uitgevoerd. LASTDATA is wel een binair bestand en wordt dus uitgelezen met een binaryreader.

```
'Filestraam & binaryreader instantïeren
            fs = New FileStream(LastDataPath, FileMode.Open,
FileAccess.Read)
            reader = New BinaryReader(fs)

            'eerste regel uitlezen -> bevat de datum
            reader.Read(raw_data, 0, 30)
            'Datum ophalen & omvormen naar een meer leesbare
vorm.
            datum = raw_data
            datum = datum.Substring(16)
            datum = datum.Substring(4, 2) & "/" &
datum.Substring(2, 2) & "/" & datum.Substring(0, 2) & " " &
datum.Substring(6, 2) & ":" & datum.Substring(8, 2) & ":" &
datum.Substring(10, 2)
```

Na dat de hoofding is ingelezen, worden de echte waarden in LASTDATA uitgelezen. Een waarde begint vanaf de 32ste byte en is telkens 4 bytes lang.

```
'LastData uitlezen & aantal rijen aan DataTable toevoegen
For i = 0 To (AantalKanalen * 4) - 1
    reader.BaseStream.Seek(32 + i, SeekOrigin.Begin)
    b = reader.ReadSingle
    b = Round(b, 1)
    waardeList.Add(b)
    dt.Rows.Add()
    i += 3
Next
```

Nadat LASTDATA.DAT uitgelezen is, wordt de DataView aangemaakt en worden de waarden die in de Lists staan, in de DataView geplaatst.

```
'DataView aanmaken
dv = New DataView(dt)

'Waarden uit de lists halen en in de dataview stoppen
For Each dr As DataRowView In dv
    dr("naam") = naamList(teller)
    dr.Item("eenheid") = eenheidList(teller)
    dr.Item("waarde") = waardeList(teller)
    teller += 1
Next
```

Als deze For Each structuur is doorgelopen, is de DataView opgebouwd en wordt de SPGridView opgebouwd via de ondertussen vertrouwde wijze. De SPBoundFields die worden ingesteld zijn:

- Naam: naam van het kanaal
- Waarde: huidige waarde van het kanaal
- Eenheid: eenheid waarin de waarde gelezen moet worden.

5.5.3.2 Webservice

Het Bidasse systeem is qua naamgeving niet consequent. Bijvoorbeeld, het BACCHANAL experiment heeft bijhorende .cha bestanden genaamd "Chips.cha". Om zulke moeilijkheidheden niet rechtstreeks in de code te schrijven, heb ik een webservice geschreven die van vanaf code kan aangesproken worden.
Als men bijvoorbeeld "Chips.cha" terug wil veranderen naar "bacchanal.cha" hoeft men enkel het achterliggende XML-bestand aan te passen op de Web Part terug te laten werken.

OpbouwBidasseExperiments.XML

BidasseExperiments.XML is het XML-bestand dat alle verwijzingen naar de benodigde Bidasse bestanden bevat. Het ziet er als volgt uit:

```
<?xml version="1.0" encoding="utf-8" ?>
<experiments>
  <experiment>
    <name>BACCHANAL</name>
    <settingsfile>\\*****\Chips.set</settingsfile>
    <channelsfile>\\*****\Chips.cha</channelsfile>
    <lastdatafolder>\\*****\</lastdatafolder>
  </experiment>
  <experiment>
    <name>ASTIR</name>
```

Figuur 5.5.3.2.1: Voorbeeld uit XML-bestand

Functies

De Webservice bevat 3 functies:

- GetChaFile: geeft de locatie en de naam van het overeenkomstige .cha bestand terug
- GetSetFile: geeft de locatie en de naam van het overeenkomstige .set bestand terug
- GetLastDataFolder: geeft de map van het LASTDATA.DAT bestand terug.
- De code van deze 3 functies lijken zeer sterk op elkaar.

Het enige verschil is dat "`.ChildNodes.Item(2).InnerText`" een andere waarde krijgt naar gelang de positie in het .XML bestand

```
<WebMethod()> Public Function GetChaFile(ByVal ExperimentName As String)
As String

Dim xmlDoc As New XmlDocument()
Dim i As Integer
Dim resultCha As String = ""

xmlDoc.Load(Server.MapPath("") & "\BidasseExperiments.xml")
For i = 0 To xmlDoc.ChildNodes.Item(1).ChildNodes.Count - 1
    If ExperimentName.ToUpper =
xmlDoc.DocumentElement.ChildNodes.Item(i).ChildNodes.Item(0).InnerText
Then
        resultCha =
        xmlDoc.DocumentElement.ChildNodes.Item(i).ChildNodes.Item(2).Inne
        rText
```

```
    End If
Next

If resultCha = "" Then
      Return "No items found"
Else
      Return resultCha
End If

End Function
```

6 DEPLOYMENT

Het ontwikkelen van een Web Part gebeurt allemaal lokaal op je computer. Om de Web Part te gaan testen, moet men deze op de SharePoint Server zetten. Dit proces noemt men "Deployment". In dit hoofdstuk leg ik uit welke manier ik gebruikt heb om een Web Part te deployen.
In 6.1 som ik de benodigdheden op die noodzakelijk zijn om een Web Part in gebruik te nemen. De GAC of Global Assembly Cache licht ik toe in 6.2. De uitleg van het .cab bestand en de inhoud hiervan kan je vinden in 6.3. Het gebruik van stsadm.exe staat beschreven in 6.4. In 6.5 beschrijf ik de batch bestanden die ik heb geschreven om de installatie te vergemakkelijken. Als laatste (6.6) beschrijf ik de stappen die je moet volgen om een Web Part te deployen.

6.1 Benodigdheden

Om een zelfgeschreven Web Part te installeren of te testen moet je voldoen aan de volgende punten:

- Een geldig account met de juiste permissies tot de SharePoint server
- Een Web Part om te testen
- Een .cab bestand
- Een kopie van stsadm.exe
- Eventueel de scripts die ik geschreven heb

6.2 Global Assembly Cache

Iedere computer waar het Microsoft .NET framework op geïnstalleerd is heeft een code cache genaamd Global Assembly Cache. De GAC bewaart samenstellingen van code & klassen (assembly's) die bedoeld zijn om door verschillende programma's gebruikt te worden.
Er zijn verschillende manieren om assembly's te installeren in de Global Assembly Cache:

- Een installer die ontworpen is om met de GAC te werken
- Via een bijgeleverd tooltje genaamd "gacutil.exe"
- Drag & Drop via Windows Explorer

De GAC bevindt zich in de map "C:\WINDOWS\assembly". Omdat dit een nogal delicate map is, wordt deze vaak beschermd door Access Control Lists (ACL's) om gewone gebruikers geen toegang te geven tot deze map.
Om een assembly te installeren moet men dus een Administrator zijn of een account hebben met Administrator rechten. Wat verder noodzakelijk is dat de inhoud van de assembly getekend is met een Strong Name Key. De Strong Name zorgt ervoor dat de assembly uniek wordt gemaakt.

6.3 Cabinet File

Een Cabinet File of .cab bestand is een verzameling van gecomprimeerde bestanden herwerkt naar 1 bestand. Ongeveer zoals een .zip bestand. Verschillend aan een .zip is dat een .cab bestand installatiebestanden bevat die dienen om gekopieerd te worden naar het systeem van de gebruiker. Terwijl een .zip bestand elk soort bestanden kan bevatten.

De inhoud van een .cab bestand om een SharePoint Web Part te installeren is als volgt:

- Primary Output from Web Part
- Manifest.xml
- WebPart.dwp

De primary output from Web Part omvat de .dll die wordt aangemaakt als de code van je Web Part wordt gecompileerd. Hierin staan dus alle klassen en code die nodig zijn om je Web Part te laten werken.
Manifest.xml bevat informatie over de .dll. De naam ervan, de namespace kan je hierin terug vinden en instellen. Niet enkel informatie over de .dll maar ook de naam van het .dwp bestand vind je hierin terug
WebPart.dwp is een xml bestand dat informatie bevat over de Web Part zelf. De naam, beschrijving, informatie over de assembly is hierin terug te vinden. Niet enkel deze maar ook interface gebonden items zoals bijvoorbeeld een icoontje kan je hierin instellen.

6.4 stsadm.exe

stsadm.exe is een commando tool die bij Microsoft SharePoint geleverd wordt. Dit programma maakt verschillende processen beschikbaar die niet beschikbaar zijn via de "Central Administration" van Microsoft SharePoint.
Ik heb stsadm.exe gebruikt om mijn Web Parts te deployen door middel van een .cab bestand (hierboven beschreven). Het zorgt automatisch voor handelingen die je anders manueel moet doen. Enkele voorbeelden hiervan zijn:

- Het .cab bestand kopiëren naar de GAC
- Assembly toevoegen aan SharePoint
- Assembly definiëren als veilig in de web.config van de centrale Sharepoint server

Stsadm.exe is een commando tool, wat wil zeggen dat het uitgevoerd moet worden via een command prompt of commando venster in Windows. Dankzij deze eigenschap kan men dus scripts schrijven die deze commando's verwerken. Het script kan uitgevoerd worden door een dubbele klik.

6.5 Deployment Scripts

Om de deployment te vergemakkelijken en te automatiseren heb ik 3 BATCH bestanden geschreven. BATCH bestanden zijn tekstdocumenten die commando's bevatten. Deze commando's worden door Windows herkend en kunnen dus uitgevoerd worden. BATCH zijn herkenbaar door de extensie .bat. Omdat deze scripts sterk op elkaar lijken leg ik de eerste uitvoerig uit en verklaar ik voor de andere scripts enkel de verschillen met de eerste.

6.5.1 Install.bat

Install.bat zorgt ervoor dat een Web Part geïnstalleerd wordt op de Sharepoint server.
Install.bat heeft de volgende code:

```
@ECHO OFF
set dir=C:\Program Files\Common Files\Microsoft Shared\web
server extensions\12\BIN
set filename=WebPartCab.CAB
ECHO -- Installing new Web Part --
stsadm -o addwppack -force -globalinstall -filename
%filename%
ECHO.
ECHO -- Forcing web service to reset --
iisreset /noforce
ECHO.
ECHO.
```

```
ECHO Script ran successfully.
ECHO If no errors are present in the above text, your new Web
Part has been installed!
ECHO.
PAUSE
```

@ECHO OFF zorgt ervoor dat men de commando's zoals `stsadm –o …` niet ziet en dat men enkel de output op het scherm te zien krijgt. Hierna worden 2 variabelen aangemaakt. De variabele `"dir"` bevat de directory waarin stsadm.exe zich bevindt. Dit om later in de echte omgeving te implementeren in een groot BATCH bestand. De andere variabele `"filename"` bevat de naam van het eerder aangemaakt Cabinet bestand.
ECHO zorgt ervoor dat de tekst achter het commando `(-- Installing new Web Part --)` wordt getoond op het scherm.
Hierna wordt stsadm.exe opgeroepen `(stsadm -o addwppack -force -globalinstall -filename %filename%)`. Zoals je ziet zijn er verschillende parameters die aangeven wat stsadm moet doen. Ik licht ze kort even toe:

- `-o` : Deze Parameter kondigt aan dat er een operatie aankomt
- `Addwppack` : Parameter om aan te duiden dat er een Web Part geïnstalleerd moet worden
- `-force` : Dankzij –force worden vorige versies overschreven door de nieuwe
- `-globalinstall` : zorgt ervoor dat de assembly in de GAC wordt geschreven
- `-filename %filename%` : bepaalt de bestandsnaam van het .cab bestand. %filename% is een verwijzig naar de parameter "filename" bovenaan.

Volgende regels in het script laten een lege lijn zien `(ECHO.)` en de regel `"—Forcing web service to reset –"`
Om er zeker van te zijn dat de Web Part wordt weergegeven kan men IIS herstarten. Handig hieraan is dat men kan kijken in de log bestanden van IIS wanneer de Web Part geïnstalleerd werd. Dit wordt mogelijk gemaakt door het commando `iisreset /noforce`
Nadien worden er weer enkele lege lijnen en tekst afgedrukt. Tot slot zorgt het commando PAUSE ervoor dat het commando venster niet wordt afgesloten, maar dat men op een toets dient te drukken om door te gaan.

6.5.2 WPList.bat

Wplist.bat zorgt ervoor dat de gebruiker een overzicht te zien krijgt van de reeds geïnstalleerde Web Parts. De code ziet er als volgt uit:

```
@ECHO OFF
ECHO Current Web Parts installed:
ECHO.
set dir=C:\Program Files\Common Files\Microsoft Shared\web
server extensions\12\BIN
stsadm -o enumwppacks
ECHO.
ECHO.
PAUSE
```

Hierbij wordt de parameter enumwppacks gebruikt om de lijst op te vragen.

6.5.3 Uninstall.bat

Zoals de naam zegt, zorgt uninstall.bat ervoor dat een Web Part van de SharePoint server wordt verwijderd en zo onbeschikbaar gemaakt wordt voor alle SharePoint Sites. De code ziet er als volgt uit:

```
@ECHO OFF
set dir=C:\Program Files\Common Files\Microsoft Shared\web
server extensions\12\BIN
set filename=WebPartCab.cab
ECHO -- Uninstalling the Web Part --

stsadm -o deletewppack -name %filename%
ECHO.
ECHO.
ECHO Script ran successfully, if no errors are present in the
above text, your new Web Part has been uninstalled!
ECHO.
PAUSE
```

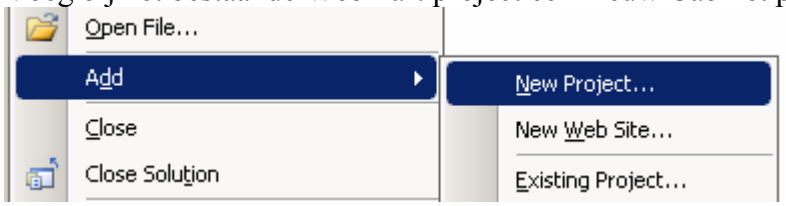
Ook bij dit script moet eerst de filename van het geïnstalleerde .cab bestand worden opgegeven.
Dankzij de parameter deletewppack van stsadm wordt de Web Part verwijderd. –name %filename% verwijst naar de bovenaan opgegeven bestandsnaam.

6.6 Stappenplan

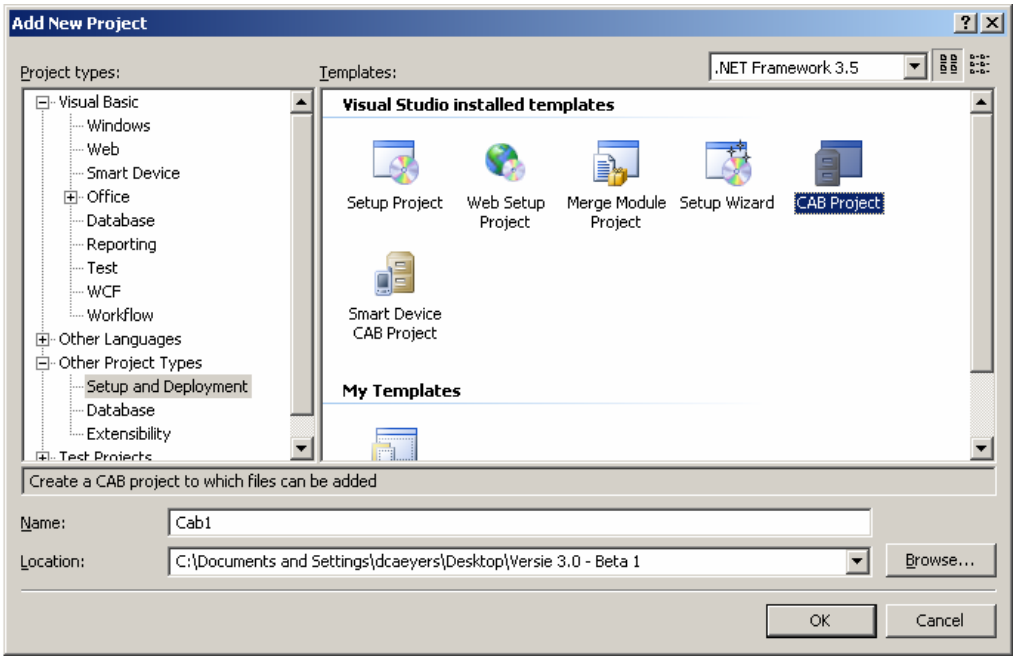
Om de samenhang te verduidelijken geef ik een kort stappenplan weer.

Stap 1: Cabinet project toevoegen aan huidig project

Voeg bij het bestaande Web Part project een nieuw Cabinet project toe.



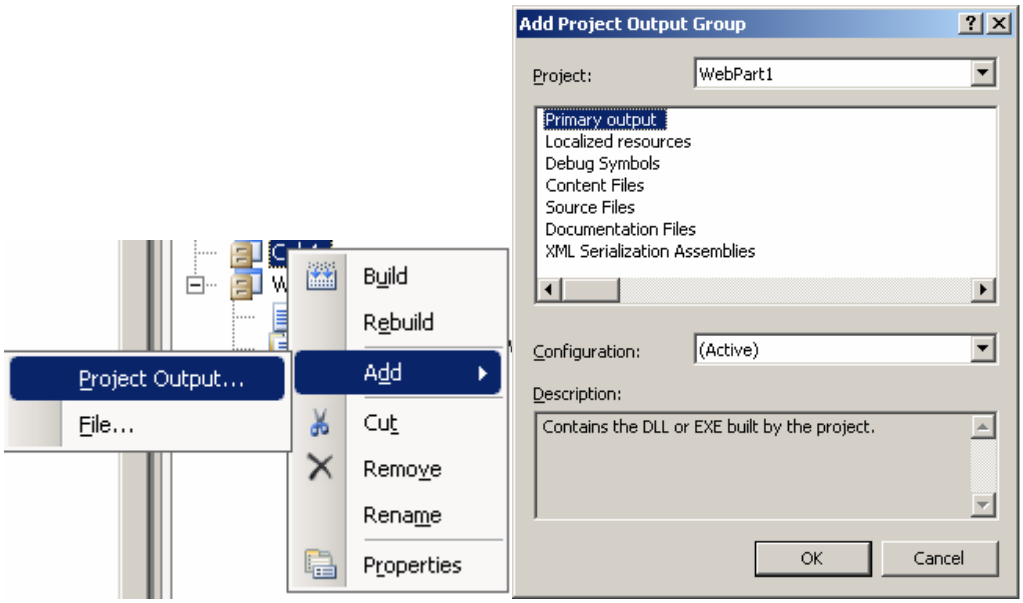
Figuur 6.6.1: Add new project – Visual Studio 2008



Figuur 6.6.2: Choose CAB Project – Visual Studio 2008

Stap 2: Items toevoegen aan het CAB project

1) Project Output



Figuur 6.6.3: Add Project output – Visual Studio 2008

2) Manifest.xml & WebPart.dwp

Rechterklik op Cab Project -> Add -> File

Stap 3: Projecten Builden, stsadm en scripts kopiëren

Compileer beide projecten door middel van:

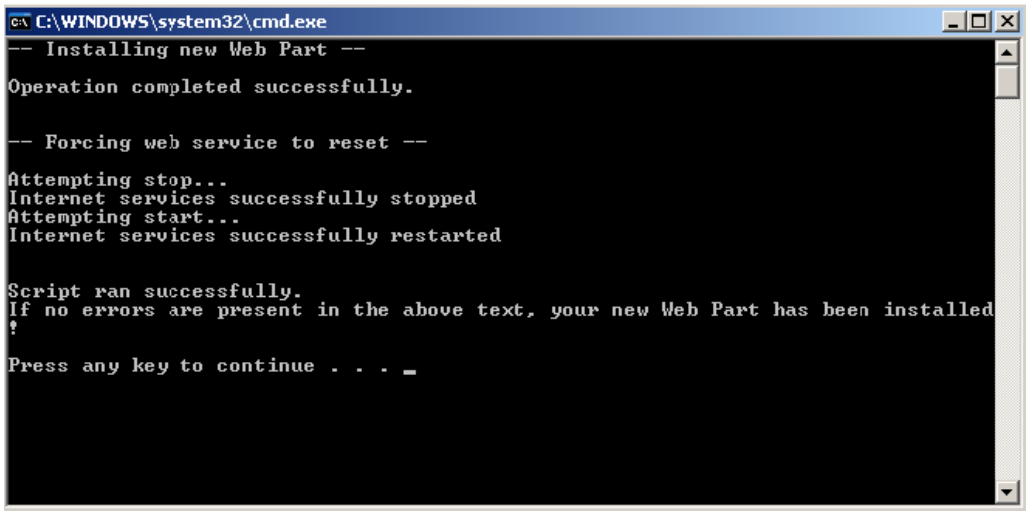
Rechterklik op projectnaam -> Build

Belangrijk bij het builden van de projecten is dat je eerst het project van je Web Part moet builden en nadien pas je CAB project. Anders kan het gebeuren dat je CAB project de vorige .dll van je Web Part project gebruikt, met als gevolg dat je geen wijzigingen zal zien. Log aan op je SharePoint Server Machine als administrator en kopieer het .cab bestand naar een map naar keuze. Het .cab bestand kan je vinden in de Bin / Release map van je CAB project.

Stsadm kan je kopiëren van "`C:\Program Files\Common Files\Microsoft Shared\web server extensions\12\BIN`" op je Sharepoint server naar de map van je .cab bestand. Kopieer tevens mijn scripts hier naartoe.

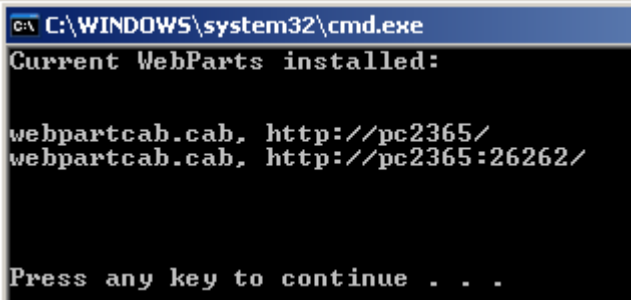
Stap 4: Install.bat uitvoeren

Dubbelklik op install.bat, als er geen errors voorkomen, is je Web Part geïnstalleerd op je server.



Figuur 6.6.4: Weergave van Install.bat

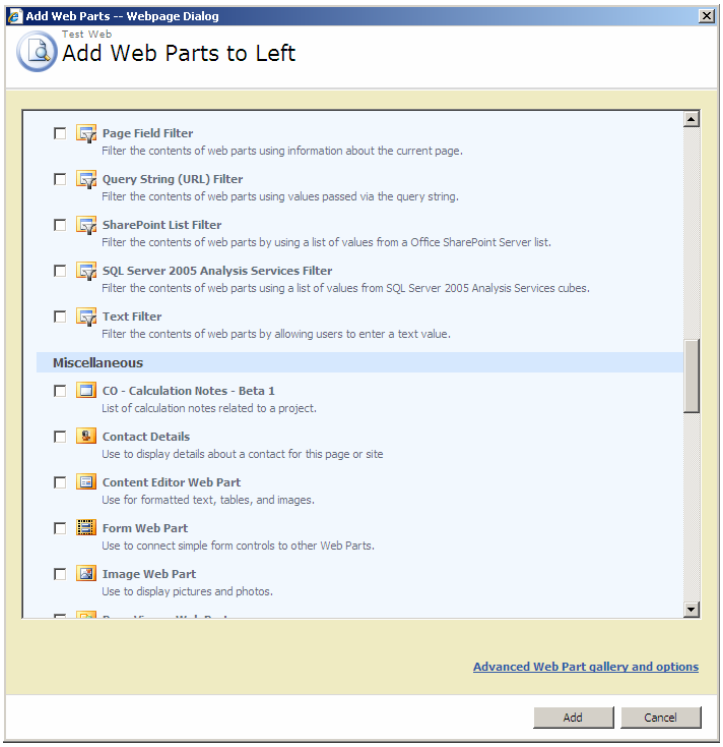
Als je wblist.bat uitvoert kan je nakijken of je Web Part effectief geïnstalleerd is.



Figuur 6.6.5: Weergave van wplist.bat

Stap 5: Web Part aan je Sharepoint pagina toevoegen.

Start je Internet Explorer en ga naar het adres van je Sharepoint Server. Via Edit Page kan je je pagina aanpassen en een Web Part toevoegen. Je pas geïnstalleerde Web Part zal nu verschijnen in de lijst.



Figuur 6.6.6: Add Web Part in Microsoft SharePoint

BESLUIT

De bedoelding van dit stageproject was om zelf Web Parts te schrijven voor het nieuwe intranet van het SCK•CEN. Deze Web Parts dienen om de expertisegroep "Reactor Technology Design" te ondersteunen. Dit door Access Databases en data acquisitie bestanden uit te lezen en zo de informatie online beschikbaar maken.
De ontwikkeling is gebeurd via VB.NET en de Web Custom Control. Deze control maakt het mogelijk om het projectje te compileren tot .DLL bestand, om het zo op de Microsoft SharePoint Server te implementeren.
De werknemers van het SCK•CEN zullen dankzij mijn Web Parts veel sneller en eenvoudiger aan hun relevante informatie geraken.
Hiernaast kunnen mijn Web Parts ook als een soort van blauwdruk gebruikt worden. Zodat de software ontwikkelaars ook meer inzicht krijgen in het ontwerpen van Web Parts voor Microsoft SharePoint.
Dankzij de Help-bestanden en de procedures die ik geschreven heb, kunnen de medewerkers de Web Parts zonder moeite gebruiken. En kunnen de medewerkers van Infoplan mijn Web Parts op een eenvoudige manier installeren.
Dankzij deze stageopdracht heb ik kennis kunnen maken met Microsoft SharePoint. Het heeft mij veel geleerd over het effect van een drastische verandering zoals een nieuw intranet.
Mijn technische kennis over het programmeren, vooral OO-programmeren is er ook sterk op vooruit gegaan.
Dankzij deze unieke kans om op het SCK•CEN stage te lopen, ben ik ondertussen klaargestoomd voor het bedrijfsleven.

BIBLIOGRAFIE

WWW-sites
Ken Cox. (2004). Building a SharePoint Web Part
Gevonden op 02 April 2008 op het internet:
http://www.ftponline.com/vsm/2004_en/magazine/features/kcox/default.aspx
Paul Robinson. (2007). SPGridview and SPMenufield
Gevonden op 15 April 2008 op het internet:
http://blogs.msdn.com/powlo/default.aspx

MICROSOFT CORP. (2003). Technet
Gevonden op 28 Maart 2008 op het internet:
http://technet.microsoft.com/

CD-ROM
SCK•CEN Information Package 2007-2008

BIJLAGE1: COMPETENTIES

Gegevens behandelen

Het behandelen van gegevens en deze op een zo gebruikersvriendelijke manier weergeven is iets wat zeer belangrijk is. Vooral als het over grote databanken gaat die gebruikt worden in grote ondernemingen. Zulke databanken moeten zo snel en consistent blijven ook toekomst gericht.

In het vak "Systeemanalyse" van het eerste jaar hebben we geleerd hoe we gegevens moeten modelleren en bewaren via de manier van Codd. Op deze manier zorgen we ervoor dat er geen dubbele gegevens in de databank terecht komen en dan er via relaties overbodige gegevens worden uitgehaald. Dankzij vele oefeningen hebben we via deze manier de basis van het "ASP-Project" in het tweede jaar gelegd. Door deze stevige basis kunnen applicaties beter en efficiënter werken.

Naast het modelleren hebben we in het derde jaar het vak "Beheer van databanken" gekregen. Dit vak heeft ons leren werken met "Microsoft SQL Server". Hierin hebben we geleerd hoe we databanken moeten onderhouden, back-ups nemen, administratieve handelingen uitvoeren, … Dit heeft mij sterk geholpen tijdens mijn stage, omdat op het SCK•CEN ook gebruik wordt gemaakt van Microsoft SQL Server databanken.

Ook dankzij het vak "XML" heb ik gebruik gemaakt van een XML-bestand om gegevens te bewaren en te beheren. In de webservice die ik gebruik voor "Bidasse" schrijf ik mappen weg om deze nadien gemakkelijk te kunnen aanpassen en te gebruiken.

Als laatste hebben we in het vak "SQL" geleerd hoe we efficiënt gegevens kunnen uitlezen zonder al te veel geheugen van de servers te gebruiken.

Analyseren

De analyse vormt de basis van een goede applicatie. Met een sterke en goede analyse wordt de tijd om een programma te schrijven sterk verkort en wordt deze op een efficiëntere ontwikkeld.

Dankzij de Unified Modelling Language (UML) die we in het vak "Systeemanalyse" geleerd hebben, kunnen we correcte analyses maken. Met behulp van verschillende diagrammen kunnen we de funtionele eisen van een programma in kaart brengen en zo beter programmeren. Dankzij deze diagrammen kunnen we ook beter contact hebben met de opdrachtgevers om hun meer duidelijkheid te geven in het systeem dat ontwikkeld wordt. UML is immers een taal die door elke programmeur gelezen kan worden.

Dankzij het ASP-Project in het tweede jaar hebben we dit in de praktijk kunnen toepassen door onze analyses met de opdracht gevers te bespreken.

In het tweede jaar "Systeemanalyse" hebben we kennis gemaakt met patronen die de analyse van een programma veel korter maakt. We hebben businessanalyses gemaakt om de huidige bedrijfsprocessen in kaart te brengen om zo aan te duiden wat de toegevoegde waarde van een nieuw programma in het bedrijf kan zijn.

Ook hebben we kennis gemaakt met programma's zoals "Rational Rose" en "StarUML" om de analyses te digitaliseren en gemakkelijker uit te werken.

Niet alleen bij het programmeren, maar ook bij het opstellen van netwerken tijdens het vak "CISCO" hebben we analyses moeten maken. Voorop neerschrijven van hoe het netwerk er gaat uitzien, welke PC welk IP-adres krijgt, …

Oplossingen uitwerken

Het uitwerken van oplossingen is ruim aan bod gekomen door de verschillende projecten die we moete maken hebben.

Als eerste kleine voorsmaak hebben we in het eerste jaar verschillende kleine opdrachtjes gekregen in het vak "VB.NET". Dit heeft ons geleerd om efficiënt te programmeren en te documenteren. Het schrijven van commentaarlijnen en handleidingen heeft veel meer zin gekregen dankzij deze opdrachtjes.

In het tweede jaar zijn we een stap verder gegaan met het ASP-Project. Dit project is was verspreid over het hele jaar. Beginnende met de analyse en eindigen met het afleveren van een goed werkende ASP website. Hierbij moesten we uitgebreide documenten maken in verband met de analyse, een volledige handleiding uitwerken en deze analyses bespreken met de opdrachtgever. Daarnaast moesten we ook testprocedures schrijven en andere studenten hun applicaties testen en beoordelen.

In het laatste jaar hebben we een veel uitgebreidere manier van werken gezien. In het vak "Businessproject" hebben we een school uit Bree geholpen met het uitwerken van een nieuw netwerk. Deze opdracht hebben we een team gemaakt. We hebben hier technieken geleerd om een Plan Van Aanpak te maken en dit te bespreken met de netwerkbeheerders van de school. Daarnaast hebben we de componenten van het netwerk moeten opzoeken, verschillende prijsanalyses moeten maken en zelfs contacten leggen met leveranciers.

Bij het ASP-Project en het project van de school in Bree, hebben we telkens een uitgebreide presentatie moeten geven om alles duidelijk uit te leggen aan de opdrachtgevers.

Oplossingen beheren
Hierbij wil ik vooral verwijzen naar onze opleiding in het programma "System Management Server". Dit programma zorgt ervoor dat men vanop afstand programma's over het netwerk van distribueren en installeren.
Om zo'n programma werkende te krijgen is er heel wat configuratie nodig. We hebben verscheidene lessen bezig geweest met het configureren van dit programma voor we een echt resultaat zagen.
Ook tijdens mijn stage is dit aan bod gekomen. De configuratie van Microsoft SharePoint is niet zo uitgebreid als die van "System Management Server" maar zonder opleiding is dit toch een heel wat werk.
Om dit te tot een goed einde te brengen heb ik raad gekregen van een Nederlands bedrijf dat zicht enkel en alleen bezig houd met Microsoft SharePoint. Ze hebben me via e-mail ondersteund om Microsoft SharePoint op te stellen op mijn eigen PC om zo alles te kunnen testen.
Hierbij hoort ook dat ik voor elke Web Part een "uninstall" script heb geschreven. Dit script zorgt ervoor dat een Web Part van de Microsoft SharePoint server verwijderd kan worden. Zodat er na verloop van tijd geen onnodige bestanden blijven staan.

Projectmatig werken

Werken in een project omgeving is iets wat in elke IT-job voorkomt. Het splitsen in taken en deeltaken, maken van planningen en werken in een team is hierbij zeer belangrijk. Werken in vorm van een project met een team is immers zeer verschillend dan alles alleen doen.

Ook hier verwijs ik naar het ASP-Project. Hierbij moesten we in het begin van het project, een planning opstellen en deze op en zo correct mogelijke wijze volgen. Deze planning werd dan opgevolgd door de docent die ons hierop ook beoordeelde. Dankzij deze planningen hebben we geleerd hoe we een groot project kunnen splitsen in afzonderlijke taken zodat we niet alles dooreen gaan doen en sommige dingen misschien over het hoofd gaan kijken. Dit project werd gemaakt in teams van 2 tot 3 personen. Hierdoor hebben we leren overleggen en aan elkaar rapporteren.

Het vak "Businessproject" past ook perfect in deze context. Net zoals bij het ASP-Project werden we ook opgedeeld in teams. Dit keer van 5 tot 6 personen. We moesten elk een taak kiezen, namelijk projectleider, teamlid, secretaris en PR. Dankzij dit project hebben we geleerd hoe het is om te werken in een team, en hoe belangrijk het wel niet is om elkaar te begrijpen en op elkaar in te spelen.

Bij het "Businessproject" hebben we ook een planning moeten maken die we moeten volgen hebben. We moesten elkaar en onszelf beoordelen volgens een vooropgestelde manier. Dit heeft ons geleerd om kritisch te zijn tegenover elk kaar, en tegenover onszelf.

Ook bij het maken van de presentaties is de teamstructuur van belang. Wie doet wat? Wie presenteert welk deel? En in sommige gevallen, wie doet er niets …

Communiceren

Communiceren is een van de belangrijkste eigenschappen om een team of een bedrijf draaiende te houden. Zonder communicatie is er geen realisatie.
Ook dit past perfect in de hiervoor vermelde projecten. Communiceren met elkaar over het project zelf, hoe gaan we dit aanpakken, wie doet wat, …
Niet enkel communiceren met elkaar, maar ook met de opdrachtgevers. Alles in de puntjes en perfect uitleggen, bespreken, bijsturen, … Dit omvat ook de interviews te de gedaan hebben bij de opdrachtgevers. Dankzij deze interviews konden we achterhalen wat de opdrachtegevers precies wilden en hoe we onze projecten daarop konden afstemmen.
Bij het "Businessproject" hebben we hierbij ook een Short List en een Long List moeten opstellen. Deze lijsten bevatten de opdrachten (of deeltjes ervan) die gaan realiseren, en welke we niet gaan uitwerken. De punten die niet gerealiseerd werden moesten we natuurlijk bespreken en duidelijk staven met voorbeelden waarom we het niet gingen doen.
Communiceren gebeurt niet in 1 taal. Dankzij de vakken "Engels" en "Frans" hebben we geleerd hoe we tegenover anderstaligen onze interesses moeten bespreken. Ook het schrijven van E-mail en brieven kwam hier aan bod.
Tijdens het vak "Nederlands" hebben we ook op een andere manier leren communiceren. Via brieven, e-mails, I-MAP handleidingen kunnen we nu op een duidelijkere en efficiëntere manier met elkaar omgaan. Hierbij hoorde ook een onderdeel "Telefoneren", hierin hebben we gezien hoe we op een correcte manier iemand verder helpen via de telefoonlijn.

Eigen gedrag aanpassen.
Het aanpassen van je eigen gedrag is essentieel bij het projectmatig werken en het werken in teamverband.
Ook hier verwijs ik naar de projecten. Bij het invullen van de evaluatieformulieren hebben we gemerkt in welke punten we zwak zijn, en in welke punten we ons kunnen verbeteren.
Daarnaast hebben we in het 2de jaar het vak "Ethiek" gekregen. Hierin hebben we geleerd hoe we onszelf kunnen beoordelen en hoe we moeten reageren op elkaar naargelang bepaalde omstandigheden.
Dit alles is vooral aan bod gekomen tijdens mijn stage op SCK•CEN. Ongeveer de helft van de dienst waar ik me bevond was franstalig, ik moest me dus aanpassen en frans spreken tegen de franstaligen. Daarnaast heb ik ook tijdig hulp van de helpdesk moeten inroepen wanneer er iets niet lukt. Ik heb mijn gedrag op deze manier aangepast door geen uren of dagen te blijven staren op hetzelfde probleem maar om tijdig externe hulp te roepen.
Na elke meeting of vergadering heb ik kort even nagedacht over hoe ik me heb gedragen, assertief, initiatief, meer op de achtergrond, … Door dat ik hier kort heb over nagedacht heb ik me mij de volgende vergadering anders gedragen als lid van de vergadering.

Kwalitatief handelen
Handelen in functie van het bedrijf. Hiermee bedoel ik je gedrag en je oplossingen afstellen op de bedrijfscultuur van het bedrijf.
Ook dit is op mijn stage hard aan bod gekomen. Tijdens het programmeren heb ik steeds in het oog moeten houden dat mijn applicatie in de toekomst ook nog kunnen gebruikt worden. Daarom heb ik de Webservice bij "Bidasse" ontwikkeld. Zodat als er een bestand verplaatst wordt, men simpel weg het achterliggende XML-bestand moet aanpassen om de applicatie te laten werken.
Bij elke Web Part heb ik ook een help-bestand geplaatst. Dankzij deze handleiding kunnen gebruikers van de applicatie steeds hun fouten opsporen of eventueel externe hulp inschakelen.
Zo heb ik ook bij elke Web Part een installatie procedure gemaakt om de medewerkers van de Helpdesk ook te ondersteunen.
Als laatste heb ik ook steeds bij elke Web Part nagedacht of het wel correct is, en of net niet iets beter kan. Op deze manier op ik minder gebruiksvriendelijke functies aangepast. Ik heb mij steeds in de voeten van de gebruiker geplaatst en vanuit zijn standpunt naar de applicatie gekeken.

BIJLAGE 2: SCREENSHOTS

CO Calculation Notes - CO Technical Notes

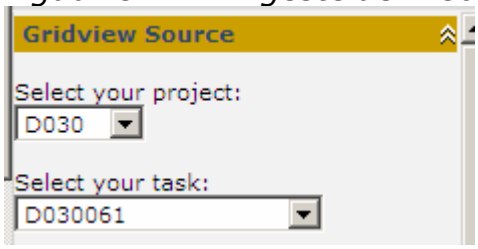
CO - Calculation Notes [2]

Task Code	Revision	Author	Date	ID	Title
D030061	REV0	PhG	03/07/2007 00:00:00	1	EVITA : dimensioneren van de overvoedingspomp.
D030061	REV0	PhG	04/07/2007 00:00:00	2	EVITA : dimensioneren van de zuigplug.
D030061	REV0	PhG	05/07/2007 00:00:00	3	EVITA : invloed van het dispositief op de BR2 hydraulica.
D030061	REV0	PhG	06/07/2007 00:00:00	4	EVITA : caracteristieken bij normaal en
D030061	REV0	PhG	31/07/2007 00:00:00	5	EVITA : caracteristieken in incidentele en toevallige werkwijzen.
D030061	REV0	PhG	01/08/2007 00:00:00	6	EVITA : schommeling van het noodzakelijke debiet om het temperatuurverschil tussen RJH en BR2 te compenseren.
D030061	REV0	PhG	02/08/2007 00:00:00	7	EVITA : spanningen in de aluminium plug.
D030061	REV0	PhG	07/08/2007 00:00:00	8	EVITA : belastingen op het brandstofelement.
D030061	REV0	PhG	06/09/2007 00:00:00	9	EVITA : weerstand van de BR2 manchet.

Download this file

Help

Figuur 8.1.1: Ingestelde Web Part



Figuur 8.1.2: Project Tool Part

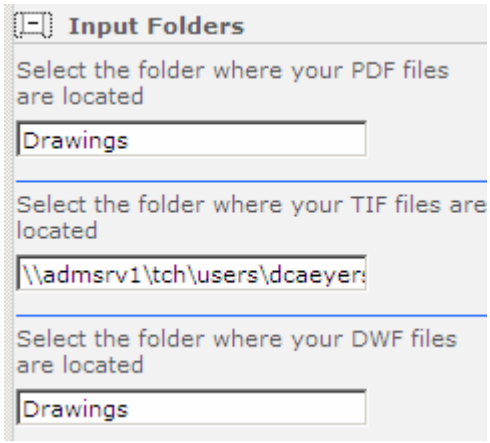
Figuur 8.1.3: Hide List Tool Part

DEO Mechanical Drawings – DEO Instrumentation Drawings

DEO Mechanical Drawings

Titel 1	Titel 2	Drawing number	Index	
Assembly fuel element TYPE S IV n/I (RJH)	EVITA	48464		DWG
Fuel element TYPE S IV n/I (RJH)	EVITA	48465		DWG
Details fuel element TYPE S IV n/I (RJH)	EVITA	48466		DWG
Assembly fuel element TYPE S V n/I (RJH)	EVITA	48470		DWG
Fuel element TYPE S V n/I (RJH)	EVITA	48471		DWG
Details fuel element TYPE S V n/I (RJH)	EVITA	48472		DWG
Koppeling brandstofelement met verlengbuis	EVITA	16952	D	DWG

Figuur 8.2.1: DEO Mechanical Drawings geconfigureerd



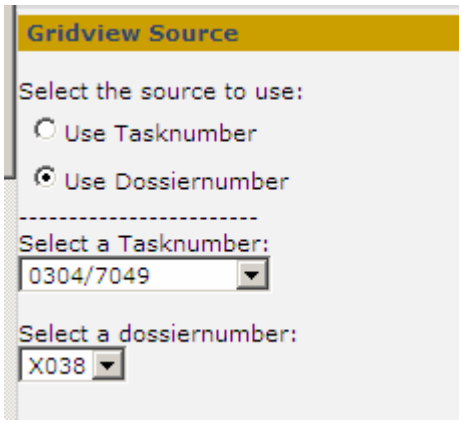
Figuur 8.2.2: Folder Tool Part

BR2 Drawings

BR2 - Drawings

Plan nummer	Revisie	Titel
60495	A	TEMPERATURE MEASUREMENT: T190 Bloc Diagram
60495-002A	J	CALLISTO - Analoge metingen en Controle Aansluitschema's - Inhoud
60495-002B	J	CALLISTO - Analoge metingen en Controle Aansluitschema's - Inhoud
60495-004	A	CALLISTO - Flowmeting F139 Aansluitschema
60495-007	A	CALLISTO - Flowmeting F144 Aansluitschema
60495-010	A	CALLISTO - Flowmeting F154 Aansluitschema
60495-013	A	CALLISTO - Flowmeting F164 Aansluitschema
60495-017	A	CALLISTO - Flowmeting F426 Aansluitschema

Figuur 8.3.1: BR2 Drawings geconfigureerd



Figuur 8.3.2: Gridview Source

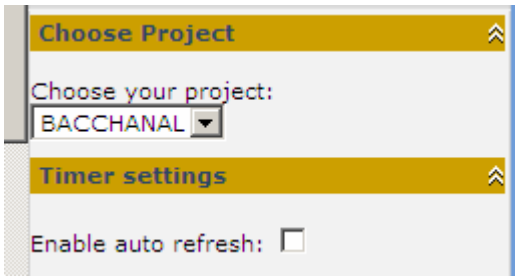
Bidasse

Bidasse

Experiment: BACCHANAL
Time: 26/05/08 09:28:10

Naam	Waarde	Eenheid
T171	23.3	□C
T172	23.3	□C
T173	23.3	□C
T11R	23.3	□C
T14R	23.3	□C
T177	23.3	□C
T178	23.3	□C
T179	23.2	□C
T174	23.8	□C
T175	23.5	□C
T176	24.1	□C
dT1-1	0	□C
dT4-1	0	□C
Qtot	-1039.2	W
Qloss	-548.6	W
QlossHead	-482.9	W

Figuur 8.4.1: Geconfigureerde Web Part



Figuur 8.4.2: Tool Parts