Een atomfeed en een monitoringapplicatie voor nucleaire experimenten

Softwareontwikkeling in .NET

Bachelor in de toegepaste informatica

Joren Proost

Academiejaar 2014-2015

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

LID VAN ASSOCIATIE KU LEUVEN

THOMAS MORE

VOORWOORD

Dit eindwerk is tot stand gekomen in het kader van een bedrijfsstage bij het SCK in Mol. De stage is het laatste onderdeel uit de bacheloropleiding 'toegepaste informatica' van de Thomas More Hogeschool Kempen.

Het zijn 13 leerrijke weken geweest voor mij, in een zeer boeiende omgeving. Het was een interessante ervaring om de kennis van drie jaar opleiding om te zetten in de praktijk. Naast de verwezenlijkingen op vlak van informatica heb ik ook kennis kunnen maken met de bedrijfswereld in een wetenschappelijke context, meer bepaald nucleair onderzoek.

Dit eindwerk zou niet gerealiseerd kunnen worden zonder de hulp van enkele mensen. Ik had hen dan ook graag bedankt voor de hulp en steun gedurende deze periode.

Allereerst wil ik mijn stagebegeleider bij het SCK, dhr. Philippe Gouat, bedanken. Hij heeft gezorgd voor de stageopdrachten en ik kon steeds bij hem terecht met vragen. Hij heeft mij gedurende de hele stageperiode gesteund om tot een goed resultaat te komen.

Ook de rest van het personeel verdient een woord van dank. Zij hebben ervoor gezorgd dat ik mij goed thuis heb gevoeld op het SCK en stonden altijd klaar voor een tasje koffie en een leuke babbel.

Verder wil ik ook de docenten van de Thomas More Hogeschool bedanken, zij hebben een groot aandeel in mijn vorming als informaticus. In het bijzonder zou ik Dirk De Peuter willen bedanken, mijn begeleidende docent tijdens de stage. Bij hem kon ik steeds terecht voor vragen en feedback i.v.m. de stage en het eindwerk.

Tenslotte zou ik ook nog mijn ouders willen bedanken, die me de kans hebben gegeven om deze studie te volmaken.

December 2014,

Joren Proost

SAMENVATTING

De stage bij het SCK bestond uit twee stageprojecten.

In het eerste project werd een Atomfeed ontwikkeld voor het rekenbureau van het SCK, dat een server beheert met meer dan 30 000 bestanden. Dit zijn wetenschappelijke artikels, rekenbladen, etc. die betrekking hebben tot de experimenten van het SCK. Het is nodig dat het hoofd van het rekenbureau op de hoogte blijft van de bestanden die op de server geplaatst worden.

De ontwikkelde feed stelt een lijst met de nieuwste 100 bestanden ter beschikking. Een feed is een lijst met gegevens die regelmatig vernieuwd wordt. Om deze feed te maken zijn een Windows service en een webpagina in ASP.NET ontwikkeld, beide in C#. De Atomfeed wordt opgebouwd zoals een XML-bestand. Gebruikers kunnen de feed bekijken via een webbrowser en de bijhorende bestanden downloaden.

De tweede stageopdracht hield in een webapplicatie te ontwikkelen voor de reeds bestaande toepassing 'BIDASSE_2008', die gebruikers toestaat om de experimenten te monitoren. Bij het SCK worden experimenten uitgevoerd in een onderzoeksreactor De tweede stageopdracht hield in een webapplicatie te ontwikkelen voor de reeds bestaande toepassing 'BIDASSE_2008', die gebruikers toestaat om de experimenten te monitoren. Voor deze experimenten worden voortdurend metingen uitgevoerd. De gemeten waarden moeten in real-time gemonitord kunnen worden door de ingenieurs.

Met de ontwikkelde webapplicatie is het voor het personeel mogelijk om via het intranet de experimenten op te volgen. Ook voor telewerkers en mobiele apparaten is de toepassing beschikbaar.

Voor elk experiment zijn de meetresultaten in real-time te bekijken en kunnen er grafieken gemaakt worden om gegevens over een langere periode te analyseren.

De webapplicatie is ontwikkeld in Visual Studio 2008 met versie 3.5 van het Microsoft .NET-framework. Als programmeertaal is C#gebruikt.

INHOUDSOPGAVE

LIJST VAN GEBRUIKTE AFKORTINGEN

.NET (Uitspraak: dot net) Een applicatieframework ontwikkeld door Microsoft.

ANS Advanced Nuclear Systems: een onderzoeksinstituut van het SCK.

ASP.NET Active Server Pages: een onderdeel van Microsoft's .NET-framework om een webserver webpagina's te laten aanmaken via programmacode.

BIDASSE BR2 Integrated Data Acquisition System for Survey and Experiments: de naam van een informaticatoepassing die gebruikt wordt binnen het SCK.

BR2 Belgian Reactor 2: kernreactor van het SCK waarin experimenten worden uitgevoerd.

C# (Uitspraak: C Sharp) Een programmeertaal van het Microsoft .NET-framework.

CEN Centre d'Etude de l'Energie Nucléaire: de Franstalige naam van het stagebedrijf.

CSV Comma-separated values: een manier om data op te slaan als platte tekst. De verschillende velden worden van elkaar gescheiden door een komma.

HTML HyperText Markup Language: een standaardtaal om webpagina's te maken.

MSS Microsoft Search Server: een zoek en indexeerplatform voor Windows-servers.

PDF Portable Document Format: een bestandsindeling voor elektronische documenten.

RSS Rich Site Summary, ook wel 'Really Simple Syndication' genoemd: een standaard om geüpdatete content automatisch aan gebruikers te tonen. Gestructureerd zoals een XML-bestand.

SCK Studiecentrum voor Kernenergie: de naam van het stagebedrijf.

SQL Structured Query Language: een standaardtaal om bewerkingen door te voeren of opzoekingen te doen in een database.

URL Uniform Resource Locator: een gestructureerde naam die verwijst naar data. Bijvoorbeeld het adres van een webpagina.

VB Visual Basic: een programmeertaal van het Microsoft .NET-framework.

XML Extensible Markup Language: een standaardformaat om gestructureerde gegevens weer te geven als platte tekst.

LIJST VAN GEBRUIKTE ILLUSTRATIES

INLEIDING

Dit eindwerk is het verslag van een stageperiode bij het SCK in Mol. Het rekenbureau van het SCK voorziet software en data voor ingenieurs die betrokken zijn bij de experimenten van het SCK. Tijdens de stage zijn er twee projecten ontwikkeld in opdracht van het rekenbureau. Beide projecten zijn in .NET ontwikkeld, met C# als programmeertaal.

Het eerste project is een Atomfeed die een overzicht geeft van de recente bestanden in de databank van het rekenbureau. Dit is handig om snel een beeld te hebben van de recente bestandenstroom op de server. Gebruikers die beroep doen op de databank kunnen eenvoudig de nieuwste bestanden downloaden, zonder ze telkens op te moeten zoeken.

Het tweede project is de BIDASSE webapplicatie die gebruikt kan worden voor het monitoren van experimenten in een onderzoeksreactor. Het is gebaseerd op BIDASSE_2008, een applicatie die al bestond op het SCK. Het data-acquisitiesysteem hiervoor was al ontwikkeld. Het voordeel van de webapplicatie is dat ze via het intranet gebruikt kan worden, zonder dat er een lokale installatie nodig is. Naar de toekomst toe kan ze ook beschikbaar worden gemaakt voor personeel dat van thuis uit werkt en voor het gebruik op mobiele toestellen.

Bij aanvang van beide projecten is een plan-van-aanpak en een analyse gemaakt, je kan deze terugvinden in dit verslag. Verder worden de realisatie en het resultaat van de opdrachten stap per stap beschreven. De stageopdracht omvat enkel de ontwikkeling van beide projecten. De implementatie zal pas gebeuren wanneer de stage al afgelopen is.

1 SCK•CEN

In dit hoofdstuk vind je een beknopte beschrijving van de stageplaats. Je komt te weten wat de activiteiten van het SCK zijn en welke afdelingen betrokken zijn bij de stageopdracht.

Het SCK•CEN of Studiecentrum voor Kernenergie (Centre d'Etude de l'Energie Nucléaire) is een Belgisch onderzoekscentrum met wereldwijde erkenning.

Figuur 1.1: logo sck-cen

Het hoofddoel van het SCK is om onderzoek te voeren dat betrekking heeft op de ontwikkeling van vreedzame toepassingen van radioactiviteit. Enkele voorbeelden zijn de veilige uitbating van kerncentrales, de berging van radioactief afval en de invloed van radioactiviteit op de mens. (SCK-CEN, sd)

Deze onderzoeken worden voor ongeveer de helft gefinancierd door overheidssubsidies. Om de rest van het budget rond te krijgen wordt er gezocht naar externe partners. Hoewel het SCK geen commerciële doelen voor ogen heeft, worden er dus wel enkele diensten aangeboden om geld vanuit de markt te verkrijgen. Enkele bronnen van inkomsten zijn o.a. materiaaltesten voor externe partners en de productie van radio-isotopen voor de medische wereld.

Het SCK omvat drie wetenschappelijke instituten met elk hun eigen verantwoordelijkheden:

- Instituut voor Nucleaire Materiaalwetenschappen
- Instituut voor Geavanceerde Nucleaire Systemen
- Instituut voor Milieu, Gezondheid en Veiligheid

(SCK-CEN, sd)

Het stageproject dat in dit eindwerk wordt beschreven hoort bij het Instituut voor Geavanceerde Nucleaire Systemen (Advanced Nuclear Systems, ANS). Dit instituut staat in voor het testen en ontwikkelen van innovatieve reactorconcepten. Hiervoor zijn samenwerkingsverbanden afgesloten met de industrie en andere onderzoeksgroepen, zowel nationaal als internationaal. (SCK-CEN, sd)

De onderzoekers van ANS kunnen beroep doen op de diensten van het Rekenbureau. Dit bureau biedt specifieke software aan voor onderzoeken van ANS. Het beheert ook de technische documentatie die bij de onderzoeken hoort (o.a. berekeningen, artikels en publicaties).(Gouat, sd)

2 ATOMFEED

Voor de eerste opdracht van de stage is een Atomfeed ontwikkeld. In dit hoofdstuk vind je een beschrijving van de opdracht, het plan-van-aanpak en de realisatie en het resultaat ervan.

2.1 Opdracht en plan-van-aanpak

In dit onderdeel vind je achtergrondinformatie over de opdracht en het plan-van-aanpak dat opgesteld is om tot een resultaat te komen.

2.1.1 Aanleiding en achtergrond

Het rekenbureau ondersteunt de projectingenieurs van het onderzoeksinstituut 'Geavanceerde Nucleaire Systemen'. Het is onder andere verantwoordelijk voor de verzameling van documentatie zoals wetenschappelijke artikels en rekenbladen die betrekking hebben tot de experimenten en onderzoeken van het SCK.

Deze documentatie bevindt zich in een datamap op een aparte server. Het personeel kan deze bestanden doorzoeken en downloaden via een webpagina op het intranet van het SCK.

Om de stroom van bestanden op de server overzichtelijk te houden zou er een lijst moeten komen waarin de 100 recentste, of laatst gewijzigde, bestanden terug te vinden zijn.

2.1.2 Verwacht resultaat

Er wordt een Atomfeed verwacht waarmee de laatste 100 gewijzigde of toegevoegde bestanden van de server getoond worden. Gebruikers moeten die bestanden via de feed kunnen openen.

Een Atomfeed is een lijst met gegevens in een XML-bestand dat regelmatig geüpdatet wordt. Feeds worden vaak gebruikt op nieuwssites, om de laatste nieuwe artikels bij te houden. In deze opdracht bevat de feed echter bestandsgegevens in plaats van links naar artikels.

Feedreaders zorgen ervoor dat het XML-bestand leesbaar en bruikbaar wordt voor eindgebruikers. In de meeste browsers zit standaard een feedreader ingebouwd, daarom kan de feed via een link beschikbaar worden gemaakt voor de gebruikers.

Het is dus de bedoeling dat er een webpagina met deze Atomfeed op het intranet komt die gemakkelijk toegankelijk en bruikbaar is.

2.1.3 Business case

Via de Atomfeed kan snel een overzicht getoond worden van de recente activiteit op de bestandserver van het rekenbureau.

De gebruikers hebben ook snel toegang tot de bestanden. Via de feed kan je in enkele muisklikken een bestand openen, terwijl er eerst op het intranet gezocht moest worden naar de bestanden. Dit kostte veel meer tijd.

2.1.4 Fasering

De opdracht is opgedeeld in verschillende fasen. Allereerst wordt een analyse gemaakt om de probleemstelling te schetsen en een oplossing te bedenken. Daarna gaat de ontwikkeling van start. Na de ontwikkeling zal de toepassing getest worden en voorzien worden van de nodige documentatie. Hieronder vind je voor elke fase een ruwe schatting van de tijd die nodig is.

Fase	Beschrijving	Duur
1	Analyseren	1 week
2	Ontwikkeling:	3 weken
	- Windows service	(2 weken)
	- Atomfeed in ASP.NET	(1 week)
3	Testen en fouten oplossen	1 week
4	Documenteren	1 week

2.1.5 Primaire doelgroep

De toepassing zal gebruikt worden door al het personeel dat betrokken is bij de onderzoeken en experimenten van ANS. Er zijn geen secundaire doelgroepen.

2.2 Analyse

De ontwikkeling van de applicatie wordt voorafgegaan door een analyse. Hierin worden de eisen van de opdrachtgever in kaart gebracht en de functionaliteiten van de applicatie besproken.

2.2.1 Functionaliteiten

Recente bestanden weergeven

Het systeem moet dagelijks een XML-bestand maken voor een Atomfeed. Dit moet een lijst met de 100 recentste bestanden van het rekenbureau bevatten.

De gebruikers moeten de Atomfeed kunnen bereiken via een link op het intranet.

Bestanden rechtstreeks downloaden

De gebruikers moeten de bestanden direct kunnen openen op de sharefolder.

2.2.2 Atomfeed

Atom is een standaard voor webfeeds, vergelijkbaar met RSS. Webfeeds zijn handig voor gebruikers omdat ze automatisch nieuwe of gewijzigde content aanbieden.

In het algemeen worden de feeds gebruikt om nieuwe blogberichten of krantenartikels samen te vatten. In dit project wordt een lijst van bestanden opgemaakt.

De data voor een Atomfeed wordt opgeslagen in een XML-bestand.

2.2.3 Indexing service

De database van het rekenbureau bevat meer dan 30 000 verschillende bestanden. Om deze gegevens te indexeren is er op Windows Server 2003 een indexing service beschikbaar.

Met deze service kan je een catalogus aanmaken van een bepaalde directory. Voor de bestanden van het rekenbureau is dit '\\admsrv1\calculation\Data'. De catalogus bevat informatie van alle bestanden in deze directory, en wordt automatisch bijgewerkt.

Voor de indexing service zijn IFilters beschikbaar. Deze filters worden gebruikt om bepaalde bestandsformaten te negeren. Indien gewenst worden sommige bestanden dus niet opgenomen in de catalogus.

Het is mogelijk om in deze catalogus te zoeken met sql-query's. Aan de hand van de velden met de creatiedatum kan dus snel achterhaald worden welke de laatste nieuwe bestanden zijn.

Vanaf Windows Server 2008 wordt de indexing service vervangen door Microsoft Search Server (MSS). MSS indexeert ook alle bestanden en is compatibel met de IFilters. Er kunnen nog steeds query's uitgevoerd worden op de geïndexeerde bestanden.

2.2.4 Structuur oplossing

De oplossing zal uit twee delen bestaan: een Windows service en een webapplicatie.

De Windows service gaat op zoek naar de recentste 100 bestanden, en maakt er een lijst van in een XML-bestand. Dit XML-bestand is opgemaakt volgens de structuur van een Atomfeed. Zo kan een feedreader het bestand juist interpreteren.

Er is gekozen voor een Windows service omdat dit een volledig zelfstandig proces is dat op de achtergrond draait. Wanneer de service geïnstalleerd is, zal die zijn taken uitvoeren zonder dat een gebruiker acties hoeft te ondernemen.

Omdat de indexing service (zie hoofdstuk 3.3) niet meer bestaat in recentere versies van Windows Server zijn er twee verschillende services gemaakt. Een daarvan werkt volledig onafhankelijk van de indexing service en kan op elk systeem werken. De andere werkt met de catalogus van de indexing service en kan eventueel aangepast worden om met Microsoft Search Server te werken.

Naast de Windows service is er een webapplicatie. Deze leest de gegevens uit het XML-bestand en zorgt voor een leesbare feed. Aan de webapplicatie is ook een downloadpagina verbonden.

De volledige beschrijving van de services en de webapplicatie vind je in hoofdstuk 2.3.

2.3 Realisatie en resultaat

In dit onderdeel vind je een beschrijving van de realisatie en het resultaat van de Atomfeed. Dit project bestaat uit twee delen. Er is enerzijds een Windows service die een keer per dag zoekt naar de meest recente bestanden. Anderzijds is er een webapplicatie die voor een Atomfeed zorgt. Kort samengevat: de Windows service zorgt voor de gegevens en de webapplicatie zorgt ervoor dat deze gegevens leesbaar en bruikbaar zijn.

2.3.1 Windows service

Een Atomfeed haalt zijn gegevens uit een XML-bestand. Dit bestand wordt aangemaakt door een Windows service.

Het is de bedoeling dat de service telkens rond middernacht (tijdens daluren) op zoek gaat naar de recentste 100 bestanden. Hiervoor is een timer voorzien, die elk uur nagaat of de dag verstreken is. In figuur 2.1 zie je de code die elk uur doorlopen wordt.

```
protected void timer_Elapsed(object sender, System.Timers.ElapsedEventArgs e)
{
   if (_lastRun.Date < DateTime.Now.Date)
   {
        timer.Stop();
        WriteXML();
        _lastRun = DateTime.Now;
        timer.Start();
   }
}
```

Figuur 2.1: codevoorbeeld uit de windows service

Het tijdstip en de datum waarop de service de laatste keer is uitgevoerd wordt bijgehouden in een variabele '_lastRun'. Na elk uur vergelijkt de service de datum van '_lastRun' met de actuele datum. Wanneer de datum van '_lastRun' kleiner is, betekent dit dat er een dag verstreken is sinds de laatste uitvoering.

In dat geval wordt de timer stopgezet en wordt de functie 'WriteXML' opgeroepen. Deze functie maakt het XML-bestand aan op de server. De datum van '_lastRun' wordt bijgewerkt naar de actuele datum en het proces begint opnieuw. Op deze manier wordt het XML-bestand steeds tussen 00.00u en 01.00u aangemaakt.

Van deze Windows service zijn twee verschillende versies ontwikkeld. Het enige verschil tussen beide services is de manier waarop de recentste bestanden gezocht worden in de functie 'WriteXML'.

<u>Via de indexing service:</u>

In de eerste versie wordt een connectie gemaakt met de catalogus van de indexing service. De lijst met bestanden wordt gemaakt aan de hand van een query die op deze catalogus wordt uitgevoerd. Je kan dit zien in figuur 2.2.

```
string connectionString = string.Format("Provider=
                                         \"MSIDXS\";
                                         Data Source=\"Calculation\";"
                                        );

OleDbConnection connection = new OleDbConnection(connectionString);
string query = string.Format(@"SELECT Write,
                                      Path,
                                      FileName,
                                      Directory
                                      FROM scope() "
                                     + @"WHERE size >= 0");
OleDbCommand command = new OleDbCommand(query, connection);
```

Figuur 2.2: lijst met bestanden via catalogus

Dit is de snelste manier om de recentste bestanden te vinden. Het uitvoeren van deze service duurt ongeveer 5 seconden.

Via de directory:

In de tweede versie worden de bestanden rechtstreeks uit de filedirectory opgehaald. De bestandspaden worden opgeslagen in een array. Je kan dit zien in figuur 2.3.

```
string[] filePaths = Directory.GetFiles(path, "*.*", SearchOption.AllDirectories);
```

Figuur 2.3: lijst met bestanden via directory

De namespace 'System.IO' van het .NET-framework bevat verschillende klassen die informatie over bestanden kunnen ophalen aan de hand van het volledige bestandspad. Je vindt een voorbeeld uit de code in figuur 2.4. Op basis van de creatiedatum of de datum van de laatste wijziging wordt dan de lijst opgemaakt.

```
string fileName = System.IO.Path.GetFileName(filepath);
string filePath = System.IO.Path.GetFullPath(filepath);
string extension = System.IO.Path.GetExtension(filepath).ToLower();
DateTime LastModified = System.IO.File.GetLastWriteTime(filepath);
DateTime Created = System.IO.File.GetCreationTime(filepath);
```

Figuur 2.4: ophalen van de bestandsinformatie

Deze manier van zoeken is veel trager en vraagt meer van de server. Voor de bestanden van het rekenbureau duurt het langer dan een minuut. Ook de IFilters worden op deze manier genegeerd, er kunnen dus geen bestanden uitgesloten worden.

Een voordeel is echter wel dat het op elk systeem werkt. Het is onafhankelijk van de indexing service. Deze service kan als een soort back-up dienen als de indexing service niet werkt of niet beschikbaar is.

Het XML-bestand ziet er steeds hetzelfde uit, ongeacht welke versie van de service gebruikt wordt. In figuur 2.5 zie je de inhoud van atomfeed.xml.

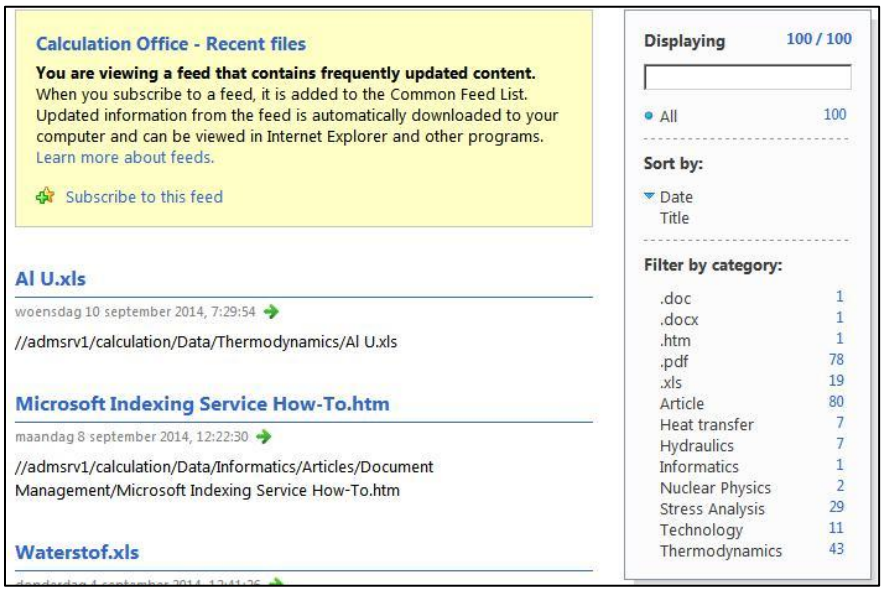
```
<?xml version="1.0" encoding="utf-8" ?>
<feed xmlns="http://www.w3.org/2005/Atom">
<title type="text">Calculation Office - Recent files</title>
<subtitle type="text">SCK, CEN, SCKCEN, calculation office</subtitle>
<id>Calculation Office - Recent files</id>
<updated>2014-09-15T08:14:46.293Z</updated>
<link rel="alternate" href="http://intern.sckcen.be/nl"/>
<entry>
<id>id=1</id>
<title type="text">Al U.xls</title>
<summary type="text">//admsrv1/calculation/Data/Thermodynamics/Al U.xls</summary>
<published>2012-05-24T05:59:18.040Z</published>
<updated>2014-09-10T06:29:54.540Z</updated>
<link rel="alternate" href="//admsrv1/calculation/Data/Thermodynamics/Al U.xls"/>
<category term=".xls"/>
<category term="Thermodynamics"/>
</entry>
<entry>
```

Figuur 2.5: voorbeeld xml-bestand voor atomfeed

2.3.2 Webapplicatie

Het tweede gedeelte van de toepassing is een ASP.NET-pagina. Dit is de pagina waar de gebruikers naar toe zullen surfen. De code achter de pagina ontleedt het XML-bestand dat door de service is aangemaakt en zet het om in een leesbare feed. De populaire hedendaagse browsers hebben standaard een feedreader ingebouwd. Die kan de feed interpreteren en de browser geeft dan een pagina weer zoals in het voorbeeld van figuur 2.6.

De gebruikers kunnen de bestanden rechtstreeks downloaden vanuit deze pagina door op de links te klikken.



Figuur 2.6: voorbeeld van de atomfeed

Een Atomfeed herkent alleen links met de prefix 'http://'. In deze applicatie wordt er rechtstreeks gelinkt naar bestanden op de server, en niet naar een webpagina. Om naar bestanden te linken wordt de prefix 'file:\\' gebruikt. De Atomfeed herkent dit niet als link, waardoor het onmogelijk is voor de gebruiker om het bestand te downloaden.

Dit probleem wordt opgelost door een bijkomende downloadpagina. Wanneer de applicatie geïmplementeerd is, kan er naar de downloadpagina gesurft worden via een normale link met 'http://' als prefix. De locatie van het bestand wordt meegegeven door een *QueryString.* Een *QueryString* is een deel van een URL waarmee een parameter kan worden meegegeven. In figuur 2.7 zie je een voorbeeld van een URL met een QueryString uit de applicatie.

```
http://intern.sckcen.be/Calculation/XMLToAtomFeed
/Download.aspx?file=\\admsrv1\calculation\data\Thermodynamics\Zink.xls
```

Figuur 2.7: url met querystring

De QueryString begint bij het vraagteken achter Download.aspx. De naam van de parameter is 'file' en de waarde is de locatie waar het bestand te vinden is.

De gebruiker wordt doorgestuurd naar de downloadpagina. De code achter deze pagina vangt de QueryString op en maakt het bestand klaar om te downloaden met 'Response.TransmitFile'. Onmiddellijk daarna wordt de gebruiker teruggestuurd naar de defaultpagina van de Atomfeed. Je kan de code hiervan zien in figuur 2.8.

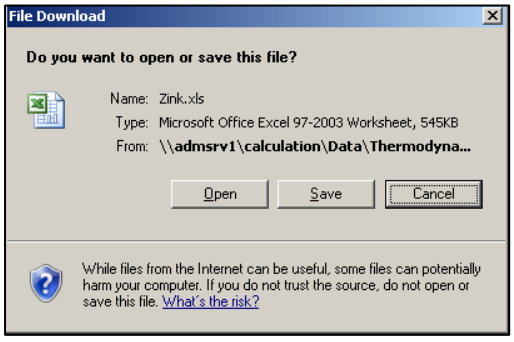
```
protected void Page_Load(object sender, EventArgs e)
{
    string path = Request.QueryString["file"];
    path.Replace("/", @"\");
    path = @"\" + path;

    FileInfo fi = new FileInfo(path);
    string filename = fi.Name;
    Response.ContentType = "application/octet-stream";
    Response.AddHeader("Content-Disposition", "attachment; filename=" + filename);
    Response.TransmitFile(path);
    Response.End();

    Server.Transfer("~/Default.aspx");
}
```

Figuur 2.8: code uit download.aspx

Dit hele proces gebeurt op de achtergrond. Er is geen voor de gebruiker zichtbare downloadpagina, alleen het venster uit figuur 2.9 verschijnt op het scherm. De bestanden kunnen alleen maar gedownload worden als de gebruiker de juiste rechten heeft.



Figuur 2.9: downloadvenster

3 BIDASSE WEB APPLICATION

De tweede opdracht van de stage is de ontwikkeling van een webapplicatie. In dit hoofdstuk vind je een beschrijving van de opdracht, het plan-van-aanpak en de realisatie en het resultaat ervan.

3.1 Opdracht en plan-van-aanpak

In dit onderdeel vind je achtergrondinformatie over de opdracht en het plan-van-aanpak dat opgesteld is om tot een resultaat te komen.

3.1.1 Aanleiding en achtergrond

In de onderzoeksreactor BR2 worden experimenten uitgevoerd. Bij elk experiment worden per minuut verschillende kanalen gemeten. Deze meetkanalen registreren waarden zoals stroomsterkte, spanning en engineering parameters (temperatuur, druk, debiet,...). Elk uur wordt een binair bestand aangemaakt dat de gemeten waarden van het betreffende uur bevat. De bestanden bevinden zich in een mappenstructuur en zijn per dag ingedeeld.

Er bestond al een toepassing 'BIDASSE_2008' die de data van elk experiment kan uitlezen en waarmee de gebruikers grafieken kunnen genereren. Deze toepassing dient echter op elke client-pc geïnstalleerd te worden, en ook updates moeten manueel worden doorgevoerd. Daarom gaf het SCK de opdracht om hiervoor een webapplicatie te ontwikkelen.

3.1.2 Verwacht resultaat

Als resultaat wordt een werkende webapplicatie verwacht waarmee de gebruikers de experimenten in de onderzoeksreactor kunnen monitoren en grafieken genereren.

De ontwikkeling van de applicatie maakt in elk geval deel uit van de stageopdracht. Omdat alle applicaties van het SCK gemigreerd zullen worden naar een nieuwe server, hoort de implementatie niet meer bij de stageopdracht. De nieuwe servers zijn pas in 2015 beschikbaar, wanneer de stage reeds afgelopen is.

Het is de bedoeling dat de applicatie ontwikkeld wordt in .NET. Dit framework wordt bij het SCK voor meerdere applicaties gebruikt, het is dus aangewezen om dit ook voor de stageopdracht te gebruiken. Zo is de code achteraf eenvoudig aan te passen en uit te breiden.

Er wordt ook verwacht dat de applicatie van de nodige documentatie voorzien wordt.

De naam van de toepassing is 'BIDASSE web application'.

3.1.3 Business case

Omdat het een webapplicatie is, hoeven gebruikers niets meer zelf te installeren of te updaten. Het volledige beheer van de applicatie gebeurt centraal via de server.

De webapplicatie kan via een beveiligd netwerk ook beschikbaar gesteld worden voor personeel dat vanaf thuis werkt. De applicatie kan ook op mobiele toestellen gebruikt worden.

3.1.4 Fasering

De opdracht is verdeeld in verschillende fasen. In dit onderdeel vind je de fasen die doorlopen worden en een ruwe planning.

Fase	Beschrijving	Duur
1	Het bestaande systeem leren kennen	1 week
2	Ontwikkeling:	3 weken
	- Monitoringgedeelte	(2 weken)
	- Extra functionaliteiten	(1 week)
3	Testen en fouten oplossen	1 week
4	Documenteren	1 week

3.1.5 Primaire doelgroep

De applicatie zal gebruikt worden door de projectingenieurs die betrokken zijn bij de experimenten in BR2. Er zijn geen secundaire doelgroepen.

3.2 Analyse

De ontwikkeling van de applicatie wordt voorafgegaan door een analyse. Hierin worden de eisen van de opdrachtgever in kaart gebracht en de functionaliteiten van de applicatie besproken.

3.2.1 Functionaliteiten

Binaire data uitlezen

De applicatie moet de gemeten waarden kunnen tonen aan de gebruiker. Deze waarden zijn in binair formaat opgeslagen op de server en moeten dus verwerkt worden naar een leesbaar formaat.

Gemiddelde waarden berekenen

Naast de waarden die in real-time worden weergegeven moet er ook een gemiddelde waarde te zien zijn voor de laatste 15 minuten.

Grafieken genereren

De resultaten van de verschillende meetkanalen van een experiment moeten in een grafiek omgezet kunnen worden. De tijdschaal moet kunnen variëren van een periode van de laatste tien uur tot de laatste drie maanden.

De naam van het kanaal en de eenheid moeten zichtbaar zijn.

3.2.2 Ontwikkelomgeving

De aangewezen ontwikkelomgeving voor het .NET-framework is Visual Studio van Microsoft. Er zijn ook gratis alternatieven zoals SharpDevelop op de markt, maar de mogelijkheden hiervan zijn eerder beperkt.

Op de stageplaats is Visual Studio 2008 beschikbaar, en dus zal dit gebruikt worden voor de ontwikkeling van de applicatie. Versie 3.5 van het .NET-framework is geïnstalleerd.

3.2.3 Programmeertaal

Er zijn verschillende programmeertalen die gebruikt kunnen worden voor .NET-ontwikkeling. Bij het SCK worden C# en Visual Basic het vaakst gebruikt. Het is dus aangeraden dat voor dit project één van deze talen gebruikt wordt. Zo kan de code achteraf nog makkelijk aangepast of uitgebreid worden.

Een keuze maken tussen VB of C# hangt vooral af van je persoonlijke voorkeur. Beide talen gebruiken dezelfde library's en op technisch vlak is geen van beide talen ontegensprekelijk beter dan de andere.

Voor dit project is gekozen voor C#. De online ondersteuning is hiervoor iets uitgebreider. Op de website van Microsoft vind je codevoorbeelden en oplossingen voor zowel VB als C#, maar elders op het internet is er meer informatie te vinden voor C#.

Er bestaan verschillende websites waar je als programmeur terecht kunt. De grootste online gemeenschap voor programmeurs is Stack Overflow (www.stackoverflow.com). Hier kan je een technisch probleem voorleggen aan collega-programmeurs, en zij zullen helpen om een oplossing te zoeken. In de meeste gevallen vind je zelfs onmiddellijk een mogelijke oplossing. De kans is immers groot dat iemand anders al eens met hetzelfde probleem geconfronteerd werd. De oplossingen en voorbeelden die op Stack Overflow verschijnen zijn meestal geschreven in C#.

3.2.4 Structuur dataopslag

Logischerwijs verwacht je dat er een databank is in een systeem met data. De metingen van de experimenten die gebeuren worden echter niet in een database opgeslagen, maar in aparte bestanden op een server.

Dit is bewust zo gedaan door SCK. Het systeem voor de data-acquisitie is opgezet in 1995 en draait dus op oudere pc's. Databanken moeten regelmatig gemigreerd worden naar nieuwere versies, omdat de ondersteuning van oudere versies wordt stopgezet. Hierbij loop je het risico dat de databanken niet meer compatibel zijn met het data-acquisitiesysteem, en dat er veel werk nodig is om de gegevens op de juiste manier in de nieuwe databank te krijgen.

Een extra obstakel is in dit geval de security die komt kijken bij nucleaire experimenten. Het data-acquisitiesysteem draait immers op een volledig afgesloten netwerk binnen het gebouw van BR2 en de communicatie met een databank zou niet altijd vanzelfsprekend zijn.

Een laatste reden om geen database te voorzien is de opslagcapaciteit. Losse binaire bestanden nemen minder plaats in dan een volledig databasesysteem. We leven nu in een tijd waar terabytes de norm zijn, maar toen het systeem ontworpen werd was dit niet zo. De experimenten lopen over verschillende jaren en elke minuut moeten er gegevens worden opgeslagen. Dit zorgt voor een grote hoeveelheid data waarvan de stockageruimte zo klein mogelijk moest worden gehouden.

Kort samengevat: men heeft niet gekozen voor een database om de volgende redenen:

- Onveranderd systeem gedurende meerdere jaren
- Compatibiliteit met data-acquisitiesysteem
- Veiligheidsoverwegingen
- Opslagcapaciteit

Als oplossing heeft men gekozen voor een systeem met verschillende databestanden.

Er zijn enerzijds bestanden in binair formaat, die alleen cijfergegevens bevatten van de metingen. Anderzijds zijn er cha-bestanden die de metadata bevatten die nodig zijn om het binaire bestand te kunnen interpreteren.

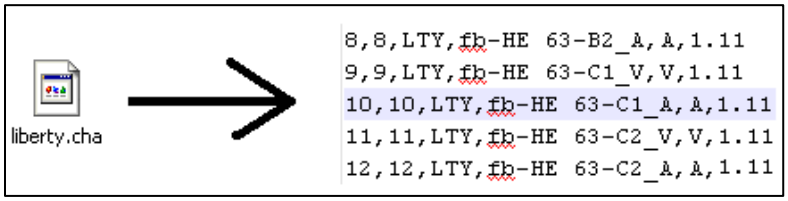
Deze verschillende bestanden en hun werking worden hieronder verder toegelicht.

Cha-bestanden

Per experiment is een cha-bestand aanwezig in een directory op de BIDASSE-server. 'Cha' duidt hier op 'channel', aangezien het bestand de informatie van alle meetkanalen van het experiment bevat.

Dit wordt duidelijk aan de hand van volgend voorbeeld:

Een van de experimenten heeft de naam 'Liberty'. Op de server staat dus een bestand 'liberty.cha'.



Figuur 3.1: inhoud cha-bestand

Een cha-bestand is opgebouwd zoals een CSV-bestand. Je kan een CSV-bestand interpreteren als een tabel. Elke regel stelt een rij voor, en een komma wordt gebruikt als scheidingsteken om de kolommen aan te geven.

Voor de gemarkeerde lijn uit het voorbeeld van figuur 3.1 geldt dus het volgende:

Kanaalnummer	10
Regelnummer in databestand	10
Extensie van databestand	LTY
Naam van het kanaal	fb-HE 63-C1_A
Meeteenheid	Ampère (A)
Getalnotatie	Twee cijfers na de komma (1.11)

Tabel 3.1: interpretatie van de inhoud van een cha-bestand

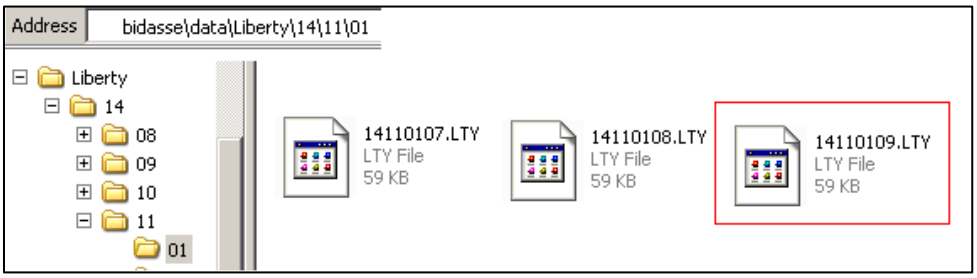
We weten dus dat we voor kanaal 'Fb-HE 63-C1_A' de gemeten waarden kunnen terugvinden op regel 10 van een databestand met extensie '.lty'. De eenheid die bij de waarde hoort, is 'A' en het getal wordt steeds met 2 cijfers na de komma getoond.

In werkelijkheid zijn er veel meer kolommen in het cha-bestand, maar die zijn weggelaten om het voorbeeld eenvoudig te houden.

Binaire databestanden

Voor elk uur van de dag wordt automatisch een nieuw binair databestand aangemaakt. Dit bestand krijgt als naam de datum en het uur waarop het aangemaakt wordt. Het formaat voor de bestandsnaam is jaar-maand-dag-uur.

Bijvoorbeeld: '2014110109.lty' is het bestand met de gemeten waarden van het Liberty-experiment op het negende uur van 1 november 2014.



Figuur 3.2: locatie van binaire bestanden in de mappenstructuur

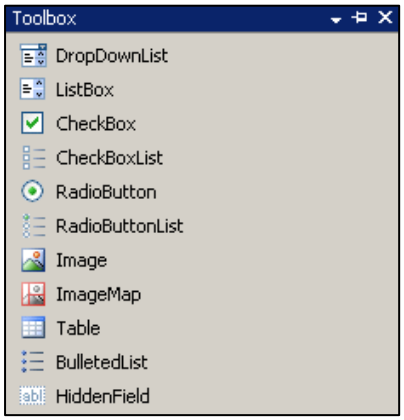
Een binair databestand is als volgt opgebouwd: elke regel bevat de waarden die voor een bepaald kanaal gemeten worden gedurende een uur. Aangezien er elke minuut een meting gebeurt, bevat elke regel dus 60 waarden.

Als we bijvoorbeeld voor het Liberty-experiment de meting willen van kanaal 'Fb-HE 63-C1_A' op 1 november 2014 om 09:30u, weten we aan de hand van het cha-bestand dat we in '14110109.lty' moeten zoeken. Het getal dat we zoeken is het 31^{e} getal (30^{e} minuut) op regel 10.

3.2.5 Chart control

Bij het programmeren in Visual Studio wordt gebruik gemaakt van controls. De controls zijn gemaakt om een applicatie van bepaalde functionaliteiten te voorzien. Via de controls is er interactie mogelijk tussen de gebruiker en de applicatie.

Heel wat controls zijn standaard ingebouwd in Visual Studio, zoals checkboxen of radiobuttons. Je kan aan extra controls komen door ze te downloaden of zelf te ontwikkelen.



Figuur 3.3: enkele standaardcontrols van visual studio 2008

In de BIDASSE web app moeten grafieken worden getoond, en dat kan het beste via een chart control. Deze is echter niet standaard beschikbaar in .NET 3.5. Het is wel mogelijk om een chart control te downloaden via de website van Microsoft. Aan de hand van een handleiding is deze geïnstalleerd in Visual Studio. (Lu, 3 april 2009)

De chart control is eerst getest in een bestaande applicatie van SCK, Material Properties (zie hoofdstuk 7).

3.3 Realisatie en resultaat

In dit hoofdstuk vind je een beschrijving van de realisatie en het resultaat van de BIDASSE web app.

3.3.1 Data uitlezen

In het project wordt de BinaryReader van het .NET-framework gebruikt om de binaire data uit te lezen.

```
//read channels from binary file
using (BinaryReader reader = new BinaryReader(File.Open(file,
        FileMode.Open, FileAccess.Read, FileShare.ReadWrite)))
{
    for (i = 1; i <= channel; i++)
    {
        for (int i2 = 0; i2 < 60; i2++)
        {
            values[i, i2] = reader.ReadSingle();
        }
    }
}
```

Figuur 3.4: code voor het uitlezen van een binair bestand

Zoals je kan zien in figuur 3.4 worden de waarden ingelezen in een tweedimensionale array of matrix. De eerste dimensie bevat het nummer van het kanaal en de tweede dimensie de minuut van het uur.

Even een voorbeeld om dit toe te lichten: values[10,31] geeft dus de waarde die op minuut 30 gemeten werd in kanaal 10.

Aan de hand van de serverdatum worden de meest recente gegevens gelezen. Vervolgens worden ze op het scherm getoond in een tabel zoals in figuur 3.5. Elke minuut wordt de tabel geüpdatet om de gegevens live weer te kunnen geven, het tijdstip van de meting wordt boven de tabel getoond. Wanneer er geen live gegevens beschikbaar zijn, worden de laatst gemeten waarden getoond. De tijdsaanduiding boven de tabel verschijnt dan in het rood.

Graph 08:45 | 2014-12-03

	channelName	current value	average value (15mins)	unit
☐	A122	45,1	34,3	A
☐	A123	0,8	0,8	A
☐	C113	60,0	58,4	%
☐	C122	9,6	9,1	kW
☐	C127	22,2	23,3	%

Figuur 3.5: tabel met waarden van een experiment

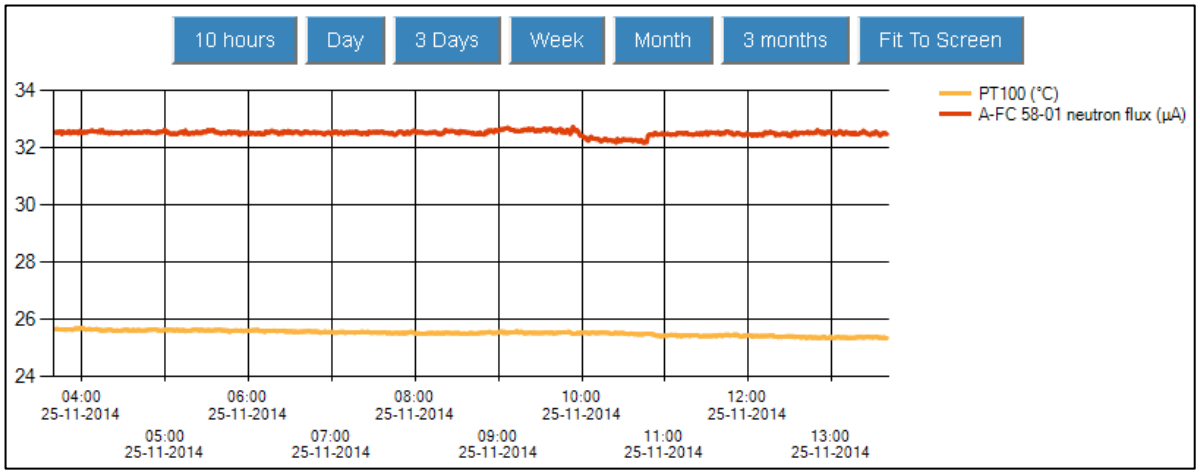
Aan de hand van de selectievakjes kan de gebruiker aanduiden welke kanalen in een grafiek getoond moeten worden.

3.3.2 Data tonen in grafiek

De grafiek wordt standaard weergegeven met de waarden van de laatste 10 uur. Verder worden de namen van de geselecteerde kanalen in de legende getoond, met de meeteenheid erbij.

De gebruiker kan de tijdschaal aanpassen met knoppen boven de grafiek. De gegevens kunnen maximaal 3 maanden terug in de tijd gaan.

De tijdsaanduiding staat op de x-as.



Figuur 3.6: grafiek van 2 meetkanalen (laatste 10 uur)

jQuery

De grafiek is een afbeelding die op de server wordt gemaakt en doorgestuurd naar de browser van de gebruiker. De server kan in principe niet weten welke schermresolutie de gebruiker heeft, of hoe hij zijn browservenster heeft ingesteld.

Deze informatie moet dus aan de client-zijde worden opgevangen, en doorgestuurd naar de server. Hiervoor zijn *Hidden Fields* gebruikt. Dit is een control in Visual Studio met dezelfde eigenschappen als een gewoon tekstveld, maar Hidden Fields blijven onzichtbaar voor de gebruiker.

```
<asp:HiddenField ID="hfWidth" runat="server" ></asp:HiddenField>
<asp:HiddenField ID="hfHeight" runat="server"></asp:HiddenField>
```

Figuur 3.7: hidden fields op de asp-pagina

Om de Hidden Fields op te vullen met de juiste waarden moet er programmacode uitgevoerd worden aan de client-zijde. jQuery biedt hier de oplossing. Wanneer de HTML-code vanuit de server is doorgestuurd, komt het jQuery-script in actie. jQuery is compatibel met de meeste populaire browsers, ook die van mobiele toestellen.

In figuur 3.8 zie je het jQuery-script uit de applicatie.

```
$(document).ready(jqUpdateSize);    // When the page first loads
$(window).resize(jqUpdateSize);     // When the browser changes size
function jqUpdateSize(){
    // Get the dimensions of the viewport
    var width = $(window).width();
    var height = $(window).height();

    $("[id$=hfWidth]").val(width);       // Set the width
    $("[id$=hfHeight]").val(height);     // Set the height

};
```

Figuur 3.8: jQuery script om de venstergrootte te bepalen

Met de functies *$(window).width()* en *$(window).height()* worden de vensterbreedte en vensterhoogte vastgesteld. Deze waarden worden weggeschreven in de *Hidden Fields*. Wanneer de gebruiker het browservenster groter of kleiner maakt, worden de *Hidden Fields* automatisch overschreven met de nieuwe afmetingen.

Op basis van deze gegevens worden de afmetingen van de grafiek ingesteld. Zo blijft hij altijd leesbaar, ongeacht de schermresolutie of venstergrootte van de gebruiker.

Als de venstergrootte aangepast wordt nadat de grafiek gemaakt is, kan de gebruiker de grafiek weer leesbaar maken met de knop 'Fit to screen' (zie figuur 3.6).

3.3.3 Lay-out

De lay-out van de pagina is opgebouwd in een masterpage. Alle subpagina's zijn hierop gebaseerd, zo blijft de structuur voor elke pagina gelijk. De masterpage bevat de header en footer die op elke pagina te zien is. Tussen de header en de footer is een *ContentPlaceHolder* geplaatst. Hierin komt de inhoud van de verschillende pagina's terecht. Het startmenu, de tabel met meetresultaten en de grafiek worden in de *ContentPlaceHolder* geplaatst. De structuur van de masterpage zie je in figuur 3.9.

```
--Header--

<!-- Content -->
<div class="wrapper col3">
  <div id="container">
    <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
    </asp:ContentPlaceHolder>
  </div>
</div>

--Footer--
```

Figuur 3.9: structuur masterpage

Om te navigeren in de applicatie is er in de header een 'breadcrumb' voorzien. Dit is een manier om te tonen in welke pagina je je bevindt en om snel terug te keren naar een van de vorige pagina's. In het voorbeeld van figuur 3.10 bevindt de gebruiker zich op de grafiekpagina. Via de links 'menu' en 'Libido' kan hij snel terugkeren naar het hoofdmenu of de meetresultaten van het experiment.

Figuur 3.10: breadcrumb

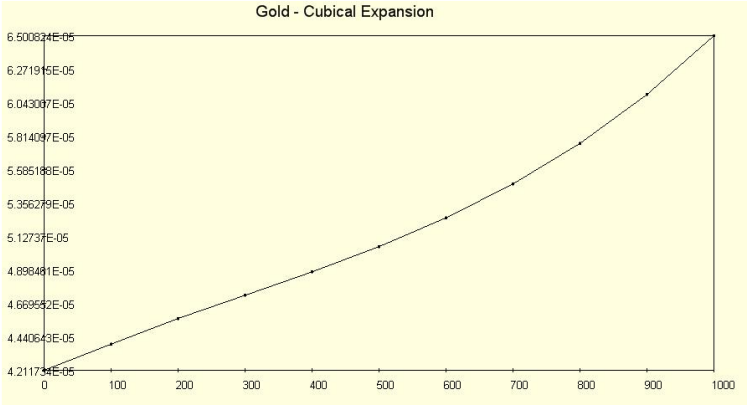
Om de lay-out in zijn geheel te zien kan je de screenshots in de bijlage bekijken.

4 MATERIAL PROPERTIES

De stage bestond voornamelijk uit twee projecten, de Atomfeed en de webapplicatie voor BIDASSE. Buiten deze projecten is er ook nog gewerkt aan andere, kleine opdrachten. Het upgraden van 'Material Properties' is een van deze opdrachten.

Het is een webapplicatie waarmee gebruikers via het intranet bepaalde eigenschappen van materialen kunnen opvragen. De applicatie is in Visual Basic geschreven is. Behalve de grafieken kunnen gebruikers ook Excelbestanden laten genereren van de waarden.

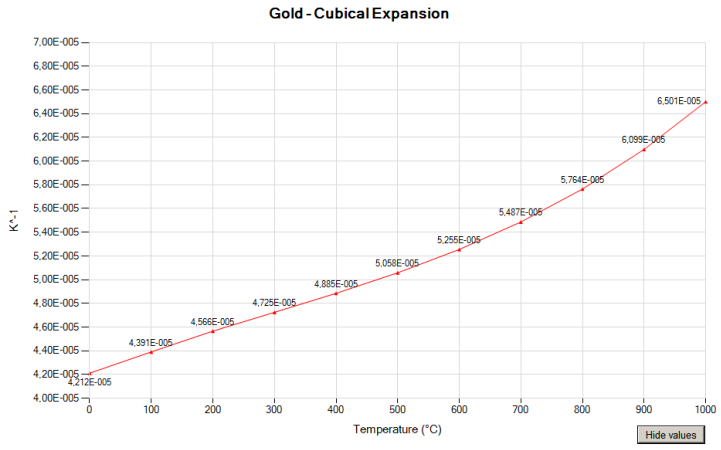
De eigenschappen van de materialen kunnen ook in een grafiek getoond worden. Deze grafiek werd nog met een oude chart control gemaakt, die weinig opmaakmogelijkheden biedt. In figuur 4.1 zie je de oude versie van de chart control.



Figuur 4.1: oude versie chart control

Om de chart control voor .NET 3.5 (zie hoofdstuk 3.2.5) uit te testen is de oude chart van Material Properties vervangen door een nieuwe.

Voor de gebruikers is er weinig veranderd, alleen de grafiek ziet er anders uit. De assen zijn voorzien van een titel en de waarden op de y-as zijn geformatteerd om beter leesbaar te zijn. Er is ook een knop bijgekomen waarmee de gebruiker kan kiezen om de waarden op de grafiek te tonen of te verbergen (zie figuur 4.2).



Figuur 4.2: nieuwe versie chart control

BESLUIT

De ontwikkeling van de verschillende applicaties is succesvol afgerond. De basisfunctionaliteiten werken en beide systemen kunnen geïmplementeerd worden wanneer de nieuwe servers van het SCK beschikbaar zijn.

Om de bestanden op de server van het rekenbureau in de gaten te houden is er nu een feed om de laatste 100 bestanden te bekijken. De bestanden kunnen van hieruit gedownload worden via een omleiding naar een downloadpagina.

Het aanmaken van de feed gebeurt volledig automatisch door een Windows service.

Voor het monitoringsysteem BIDASSE_2008 is er een webapplicatie als alternatief gekomen. De applicatie kan de experimenten van BR2 in real-time monitoren en er kunnen grafieken gecreëerd worden.

Deze webapplicatie is beschikbaar via het intranet, en kan in de toekomst toegankelijk gemaakt worden voor personeel dat thuis werkt. Het is ook mogelijk om de applicatie op mobiele toestellen te gebruiken.

LITERATUURLIJST

Gouat, P. (sd). *Calculation Office - People - Who.* Opgeroepen op 20 september 2014, van Intranet SCK-CEN.

Lu, C. (3 april 2009). *Step by Step installing MS Chart Controls on Visual Studio 2008.* Opgeroepen op 14 oktober 2014, van Can Lu Blogspot: http://canlu.blogspot.be/2009/04/step-by-step-installing-ms-chart.html

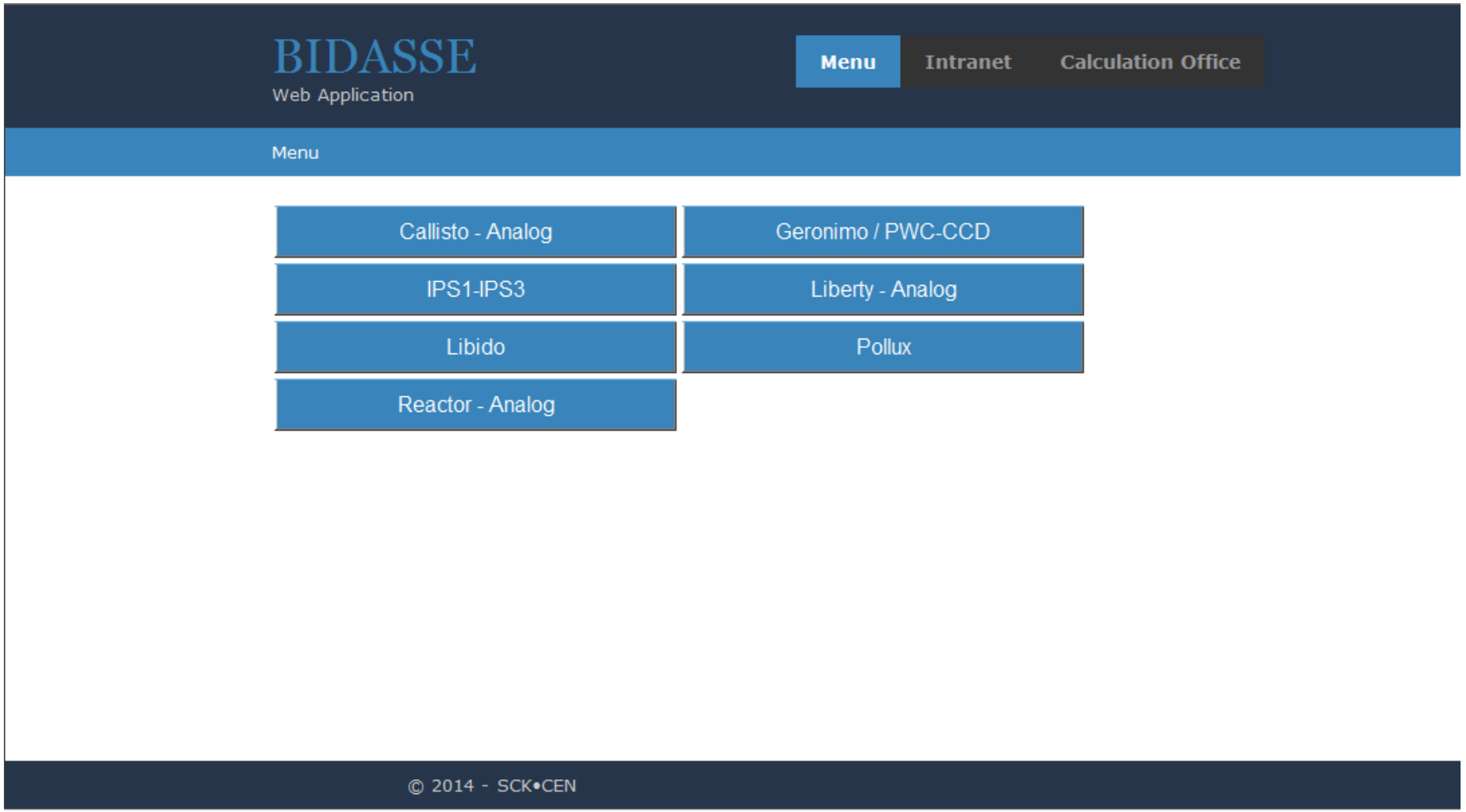
SCK-CEN. (sd). *ANS.* Opgeroepen op 20 september 2014, van Intranet SCK-CEN.

SCK-CEN. (sd). *Even voorstellen.* Opgeroepen op 20 september 2014, van SCK-CEN: https://www.sckcen.be/nl/About/Introduction

SCK-CEN. (sd). *Structuur en leiding.* Opgeroepen op 20 september 2014, van SCK-CEN: https://www.sckcen.be/nl/About/Structure

BIJLAGE: SCREENSHOTS BIDASSE WEBAPPLICATIE

Startmenu



Meetresultaten

BIDASSE
Web Application

Menu | Intranet | Calculation Office

Menu > Callisto

Graph

09:35 | 2014-12-03

	channelName	current value	average value (15mins)	unit
☐	A101	36,6	36,6	A
☐	A102	36,2	36,3	A
☐	A103	1,1	1,1	A
☐	A116A	109,5	109,6	A
☐	A116B	111,6	111,7	A
☐	A116C	63,8	63,9	A
☐	A122	50,9	45,2	A
☐	A123	0,7	0,7	A
☐	C113	60,6	60,6	%
☐	C122	8,9	9,1	kW
☐	C127	20,2	22,6	%
☐	C144	67,1	67,1	%

Grafiekpagina