Mika Laitinen

# Microsoft SharePoint 2010

## Ohjelmistokehittäjän näkökulma

Metropolia Ammattikorkeakoulu
Ohjelmistotekniikka
Tietotekniikan koulutusohjelma
Insinöörityö
7.10.2011

Metropolia

**Tiivistelmä**

| | |
|---|---|
| Tekijä(t)<br>Otsikko<br><br>Sivumäärä<br>Aika | Mika Laitinen<br>Microsoft SharePoint 2010<br>Ohjelmistokehittäjän näkökulma<br>39 sivua<br>7.10.2011 |
| Tutkinto | Ohjelmistotekniikan insinööri |
| Koulutusohjelma | Tietotekniikka |
| Suuntautumisvaihtoehto | Ohjelmistotekniikka |
| Ohjaaja(t) | Lehtori Simo Silander<br>Yliopettaja Kari Järvi |


Työn tekijällä on usean vuoden kokemus ohjelmistokehityksestä, ja hän on työskennellyt alan yrityksessä SharePoint-ohjelmistokehittäjänä vajaan vuoden verran. Työn idea on syntynyt siitä, että SharePoint-ohjelmistokehitys poikkeaa muusta ohjelmistokehityksestä suhteellisen paljon ja siihen siirtyminen vaatii aikaa.

Työn tavoitteena on tehdä aloituspaketti ohjelmistokehittäjälle, joka on aikeissa siirtyä SharePoint-ohjelmistokehittäjäksi. Työstä on tarkoitus saada hyvä kuva, mitä SharePoint-ohjelmistokehittäminen on sekä tietoa ja työkaluja käytännön tekemiseen.

Opinnäytetyössä käytiin läpi SharePointin perusominaisuudet, toiminnallisuudet, ohjelmistokehitykseen liittyvät asiat ja päivittäisessä SharePoint-ohjelmistokehityksessä tarvittavat työkalut. Lopuksi käydään läpi esimerkin avulla, miten SharePointissa voidaan toteuttaa sama asia eri menetelmillä.

Kehitysympäristönä käytettiin 64-bittistä kannettavaa tietokonetta, johon oli asennettuna VMware-virtuaalikoneympäristö. Suorituskyvyn takaamiseksi virtuaalikoneympäristö oli sijoitettu SSD-levylle. Esimerkkejä varten luotiin oma ”testsite.sp.dev:80”-web-sovellus sekä asianmukainen sivustokokoelma. Ohjelmistokehitystyökaluna käytössä oli Microsoft Visual Studio 2010 Premium.



| | |
|---|---|
| Avainsanat | Microsoft, SharePoint, SharePoint 2010, ohjelmistokehitys, intranet, extranet, www |

Metropolia

**Abstract**

| | |
|---|---|
| Author(s)<br>Title<br><br>Number of Pages<br>Date | Mika Laitinen<br>Microsoft SharePoint 2010<br>Software Developer's Perspective<br>39 pages<br>7 Oct 2011 |
| Degree | Bachelor of Science |
| Degree Programme | Information Technology |
| Specialisation option | Software Engineering |
| Instructor(s) | Simo Silander, Senior Lecturer<br>Kari Järvi, Principal Lecturer |


The purpose of this thesis is to provide a good overall view of what SharePoint software development is and to give useful information and tools for actual work. The main aim is to clarify the differences between SharePoint software development and other kind of software development and thus make SharePoint easier to adopt for those software developers who wish to become SharePoint software developers.

This study describes SharePoint basic features and functionalities. In addition it deals with software development and tools that are needed on a daily basis in SharePoint software development. The study also provides an example of how to get the same result with a different approach in SharePoint.

The development environment was based on a 64-bit laptop which had VMware virtualization environment installed. To ensure efficient performance the virtualization environment was placed on SSD. A web application "testsite.sp.dev:80" and site collection was created in order to create examples. Microsoft Visual Studio 2010 Premium was used as a SharePoint software development tool.



| | |
|---|---|
| Keywords | Microsoft, SharePoint, SharePoint 2010, software development, intranet, extranet, www |

**Sisällys**

**Lyhenteitä ja käsitteitä**

| | |
|---|---|
| API | *Application Programming Interface.* Ohjelmointirajapinta, jonka avulla eri ohjelmat voivat keskustella keskenään. |
| ASP.NET | Microsoftin kehittämä web-sovelluskehys, jonka avulla voidaan kehittää web-sivustoja, web-sovelluksia ja web-palveluja. |
| BI | *Business Intelligence.* Tietokonepohjaiset tekniikat, joita käytetään tiedon tunnistamiseen, tiedon poimintaan ja analysoimaan liiketoiminnan kannalta kriittistä tietoa. |
| CA | *Central Administration.* Keskitetty hallinta, josta tehdään käytännössä melkein kaikki konfiguroinnit. |
| CAML | *Collaborative Application Markup Language.* XML-pohjainen kuvauskieli, jota käytetään SharePointissa kenttien ja näkymien määrittämiseen. |
| DLL | *Dynamic Link Library.* Microsoftin kehittämä jaettujen kirjastojen konsepti. |
| GUID | *Globally Unique Identifier.* Yksikäsitteinen 128-bittinen luku, jota Windows-käyttöjärjestelmä ja sovellukset käyttävät identifioimaan tiettyjä komponentteja, sovelluksia, tiedostoja, käyttäjiä, jne. |
| HTTP | *Hypertext Transfer Protocol.* Verkkoprotokolla, jota selaimet ja WWW-palvelimet käyttävät tiedonsiirtoon. |
| HTTPS | *Hypertext Transfer Protocol Secure.* HTTP- ja SSL/TLS-protokollan yhdistelmä, jota käytetään salatun tiedon tiedonsiirtoon. |
| IIS | *Internet Information Services.* Microsoftin kehittämä palvelinohjelmistokokonaisuus, joka on tarkoitettu käytettäväksi Windows-pohjaisissa palvelimissa. |
| LINQ | *Language-Integrated Query.* Microsoft .NET -sovelluskehyskomponentti, jonka avulla voidaan tehdä SQL-kyselyjä natiivisti .NET-kielillä. |
| SOAP | *Simple Object Access Protocol.* XML-pohjainen protokolla, joka mahdollistaa proseduurien etäkutsun (RPC). Käytetään ohjelmistokehityksessä ohjelmien väliseen kommunikointiin internetin välityksellä. |
| SSO | *Single sign-on.* Pääsynvalvonnan ominaisuus, jonka avulla käyttäjän on mahdollista saada käyttöoikeus useampaan järjestelmään yhdellä sisäänkirjautumisella. |

| | |
|---|---|
| TFS | *Team Foundation Server.* Microsoftin kehittämä työkalu lähdekoodien kontrolloimiseen, tiedon keräämiseen, raportointiin ja projektin seuraamiseen. |
| .NET | *.NET Sovelluskehys.* Microsoftin kehittämä ohjelmistokomponenttikirjasto, joka tukee useita ohjelmointikieliä ja yhteentoimivuutta niiden välillä. |
| XAML | *Extensible Application Markup Language.* Microsoftin kehittämä XML-pohjainen kuvauskieli, jota käytetään alustamaan strukturoituja arvoja ja objekteja. |
| XML | *Extensible Markup Language.* Kuvauskieli ja standardi, jolla tiedon sekaan voidaan kuvata myös tiedon merkitys. |

# 1 Johdanto

SharePoint 2010 on laaja tuote, joka sopii hyvin monenlaisten ja kokoisten yritysten sekä yhteisöjen intranet-, extranet- ja www-sivustojen alustaksi. SharePoint tarjoaa jo sinällään paljon toiminnallisuutta. Tuotteen syvällisin idea onkin se, että sillä voidaan rakentaa sivustoja ilman tarvetta ohjelmistokehitysosaamiselle. Hyvin usein asiakkaiden vaatimukset SharePoint-projekteissa ovat kuitenkin niin BI-lähtöisiä, että sivustojen yksityiskohtainen räätälöinti on välttämätöntä.

Työn tavoitteena on antaa tietoa SharePoint-ohjelmistokehityksestä niille aloittaville ohjelmistokehittäjille, jotka ovat aikeissa siirtyä SharePoint-ohjelmistokehityksen pariin. Työssä pyritään tuomaan esille, mitä päivittäinen SharePoint-ohjelmistokehitys on ja mitä se vaatii.

Työ on toteuttu pitkälti tekijän omien kokemusten perusteella SharePoint-ohjelmistokehityksen tiimoilta.

Aluksi työssä käydään läpi SharePoint 2010:n perusominaisuuksia ja toiminnallisuuksia. Tämän jälkeen tarkastellaan tuotetta ohjelmistokehityksen näkökulmasta sekä käydään läpi tärkeimpiä työkaluja ohjelmistokehitystä varten. Lopuksi käydään läpi esimerkkien avulla, miten SharePointissa saadaan sama asia toteutettua eri lähestymistavoilla.

# 2 SharePoint 2010

Tässä luvussa käydään läpi SharePoint 2010 -tuotteen perusominaisuuksia, kuten minkä päälle tuote rakentuu, laitteistovaatimukset, selainrajoitteet, hakemistorakenne, tärkeimpien tiedostojen merkitykset ja perusominaisuudet. Jotta voidaan puhua SharePoint-toiminnallisuuksista, on erittäin tärkeää tuntea tuotteen perusominaisuudet sitä ennen.

## 2.1 SharePoint pähkinänkuoressa

SharePoint on Microsoftin kehittämä web-sovellusalusta pienille ja suurille organisaatioille. Sen pääasiallinen käyttötarkoitus on tyypillisesti toimia yritysten web-sisällön- ja dokumentinhallintajärjestelmänä.

SharePoint on hyvin monikäyttöinen, ja sen avulla voidaankin hallita ja luoda intranet-portaaleja, ekstranettejä, verkkosivustoja, dokumentin- ja tiedostonhallintajärjestelmiä, työtiloja, sosiaalisen verkostoinnin työkaluja, hakukoneita, liiketoimintaälyä sisältäviä työkaluja, kolmannen osapuolen kehittämiä sovelluksia tai sitä voidaan käyttää vaikka web-sovelluksien kehitysalustana.

SharePointin suunnittelussa on otettu huomioon skaalautuvuus, ja yhdellä ”palvelinfarmilla” onkin mahdollista tyydyttää useamman organisaation tarpeet. SharePoint-palveluja on mahdollista saada myös pilvipalveluna osana BPOS- ja Office365-palveluja.

SharePointia pitää ajatella alustana ja sovelluskehyksenä. Monia asioita voidaan tehdä pelkästään konfiguroimalla palvelinta ja aktivoimalla haluttuja ominaisuuksia. Suurin SharePointin ominaisuus kuitenkin on se, että ei tarvitse välttämättä olla mikään SharePoint-ohjelmistokehittäjä voidakseen rakentaa ja ylläpitää komplekseja verkkosivustoja.

SharePointissa on useita menetelmiä web-alueiden kustomointiin ja konfigurointiin. Perussivun editoinnin, tiedoston tallentamisen ja kustomoidun ulkoasun lisäksi yksi vallitsevista mahdollisuuksista on kolmannen osapuolen kustomointien asentaminen. Niitä kutsutaan web-osiksi ja niistä lisää luvussa 3.1.

Jokainen SharePoint-sivusto sisältää tyypillisesti kirjastoja, listoja, kalentereita, tehtäviä, ilmoituksia jne. Jokainen näistä objekteista on pitkälti räätälöity ilmentymä peruslistasta. Jokaisella listalla on erillinen ryhmä sarakkeita ja näkymiä käyttöliittymän luontia varten.

Tapahtumia käytetään käynnistämään toimenpiteitä silloin, kun tietty ehto täyttyy. Tiedon organisointi perustuu sisältölajikonseptiin, joka antaa mahdollisuuden tallentaa

erityyppisiä dokumentteja samaan kirjastoon kuitenkin pitäen rakenteen samana. Se on osaksi kompromissi tyypillisen jäsentämättömän maailman ja korkeasti jäsennetyn varastoinnin välillä tarjoten ikään kuin relaationaalisen tietokannan.

## 2.2 Microsoft SharePoint Foundation 2010

SharePoint Foundation on koko SharePoint-tuoteperheen perusta. Se sisältää kaiken päätoiminnallisuuden ja arkkitehtuurin, jota myös kaupalliset SharePoint-versiot hyödyntävät. SharePoint Foundation on ilmaistuote, mutta se vaatii toimiakseen Windows 2008 Server R2:n (joka taas ei ole ilmainen). SharePoint Foundation sisältää vain perusominaisuudet, ja sen tietokanta voi käyttää vain rajoitettua SQL Express -ilmaisversiota.

SharePoint Foundation 2010 -tuotetta edeltävän version nimi on Windows SharePoint Services 3.0.

## 2.3 Microsoft SharePoint Server 2010

Toisin kuin SharePoint Foundation, SharePoint Server -tuote on täysin kaupallinen ja siitä on tarjolla useampia eri lisenssimalleja, jotka määräytyvät käyttäjämäärän, ominaisuuksien ja sen mukaan, ovatko käyttäjät organisaation sisällä vai sen ulkopuolella.

SharePoint Server 2010 on uusi nimi aikaisemmalle versiolle, joka on nimeltään Microsoft Office SharePoint Server 2007 (MOSS).

## 2.4 Lisensointi

Mikäli SharePointia käyttää vain organisaation sisäinen henkilökunta, niin silloin tarvitaan ”SharePoint Server 2010 CAL” -lisenssi. Sen sijaan mikäli SharePoint-toteutus on esimerkiksi julkinen sivusto, jota käyttävät anonyymit käyttäjät, niin silloin tarvitaan ”SharePoint 2010 for Internet Sites” -lisenssi.

Tämän lisäksi organisaation sisäiseen käyttöön tarkoitettu lisensiointi jakautuu kahteen eri malliin: käyttäjä- ja laitekohtaiseen lisenssiin. Käyttäjäkohtaisissa lisensseissä on käyttäjämäärällä tietty raja, ja jokainen nimetty käyttäjä autentikoidaan laitteesta riippumatta. Laitekohtaisessa lisensseissä käyttäjämääriä ei ole rajoitettu, mutta

autentikoinnin pitää tapahtua nimetyiltä laitteilta. Käyttäjäkohtaiset lisenssit ovat huomattavasti yleisemmin käytettyjä ja laitekohtaisia lisenssejä käytetäänkin vain kohteissa, joissa on kiinteät työasemat, kuten kirjastoissa ja kouluissa.

SharePoint Server CAL:stä (Client Access Licence) on kaksi eri mallia: Standard ja Enterprise. Standard-version toiminnallisuuden lisäksi Enterprise-versio sisältää edistyneitä lisäominaisuuksia kuten web-sisällönhallintatoimintoja (sisältäen julkaisemisen), single sign-on (SSO) -integroinnin toisten portaalien kanssa sekä rekisterin hallinnan tukien arkaluonteisten dokumenttien säilönnän turvallisuutta ja auditointia ajatellen.

CAL-lisenssien kanssa on huomioitava se, että jotta saadaan käyttöön Enterprise-ominaisuudet, pitää olla olemassa myös Standard CAL. Jos organisaatiossa suurin osa käyttäjistä käyttää Standard-lisenssin sisältämiä ominaisuuksia ja pieni osa Enterprise-lisenssin sisältämiä ominaisuuksia, voidaan Enterprise-lisenssi hankkia erikseen tälle pienelle joukolle ilman, että tarvitsee maksaa Enterprise-ominaisuuksista koko organisaatiolle. On tosin organisaation vastuulla kontrolloida, että Enterprise-ominaisuuksia eivät käytä henkilöt, jotka käyttävät Standard-lisenssiä. SharePoint 2010 sisältää käyttäjänseurannan, jolla nähdään, kuka käyttäjä käyttää mitäkin sivustojen alueita ja ominaisuuksia.

Organisaation ulkopuoliseen käyttöön tarkoitettu lisenssi jakautuu myös kahteen eri malliin samalla tavoin kuin sisäiseen käyttöön tarkoitetut lisenssit: ”Microsoft SharePoint 2010 for Internet Sites Standard” ja ”Microsoft SharePoint 2010 for Internet Sites Enterprise”. Lisenssit ovat lähes samankaltaisia sillä erotuksella, että organisaation ulkopuoliseen käyttöön tarkoitetut lisenssit rajoittavat ylimmän tason domainien käytön yhteen. Lisäksi niihin ei tarvitse olla erillisiä CAL:leja, koska ne on tarkoitettu pelkästään ulkopuolisten henkilöiden käyttöä varten.

## 2.5 Vaatimukset

### 2.5.1 Laitteistovaatimukset

SharePoint 2010 vaatii toimiakseen 64-bittisen ympäristön, joka takaa sovelluksille stabiilisuuden, korkean suorituskyvyn ja skaalautuvuuden.

Web-palvelimet, sovelluspalvelimet ja yhden palvelimen asennuksien laitteistovaatimukset:

- Prosessori: 64-bittinen, 4 ydintä.
- RAM: 8 GB muistia (4 GB kehitysympäristö).
- Kovalevy: 80 GB järjestelmälevylle.

Tietokantapalvelimien laitteistovaatimukset:

- Prosessori: 64-bittinen, 4 ydintä (pienet ympäristöt), 8 ydintä (keskikokoiset ympäristöt).
- RAM: 8 GB (pienet ympäristöt), 16 GB (keskikokoiset ympäristöt).
- Kovalevy: 80 GB järjestelmälevylle.

### 2.5.2 Ohjelmistovaatimukset

Tietokantapalvelinfarmissa, jokin seuraavista:

- 64-bittinen Microsoft SQL Server 2008 R2.
- 64-bittinen Microsoft SQL Server 2008 with Service Pack 1 and Cumulative Update 2.
- 64-bittinen Microsoft SQL Server 2005 with Service Pack 3.

Yhden palvelimen ympäristö sisäänrakennetulla tietokannalla:

- 64-bittinen Windows Server 2008 Standard, Enterprise, Data Center tai Web Server with SP2.
- 64-bittinen Windows 2008 R2 Standard, Enterprise, Data Center tai Web Server.

Asiakaskoneen vaatimuksena SharePointin käyttöön on tuettu selain. Microsoft on jakanut selaimet tasoihin 1 ja 2 niin, että tason 1 selaimet ovat täysin tuettuja ja tason 2 selaimilla kaikki SharePointin toiminnot eivät välttämättä toimi oikein. [1, s. 1; 2, s. 1.]

Taulukko 1. Tason 1 selaimet

| Käyttöjärjestelmä | Selain |
|---|---|
| Windows XP<br>Windows Vista<br>Windows Server 2003<br>Windows Server 2008 | Internet Explorer 7 (32-bit)<br>Internet Explorer 8 (32-bit)<br>Internet Explorer 9 (32-bit)<br>Mozilla Firefox 3.5 |
| Windows 7<br>Windows Server 2008 R2 | Internet Explorer 8 (32-bit)<br>Mozilla Firefox 3.5 |

Taulukko 2. Tason 2 selaimet

| Käyttöjärjestelmä | Selain |
|---|---|
| Apple Mac OS X Snow Leopard | Apple Safari 4.x<br>Mozilla Firefox 3.5 |
| Windows XP<br>Windows Vista<br>Windows Server 2003<br>Windows Server 2008 | Internet Explorer 7 (64-bit)<br>Internet Explorer 8 (64-bit)<br>Internet Explorer 9 (64-bit) |
| Windows 7<br>Windows Server 2008 R2 | Internet Explorer 8 (64-bit)<br>Internet Explorer 9 (64-bit) |
| UNIX/Linux 8.1 | Mozilla Firefox 3.5 |

## 2.6 Hakemistorakenne

SharePoint 2010 -järjestelmätiedostot sijaitsevat hakemistossa ”Program Files\Common Files\Microsoft Shared\Web Server Extensions\14”, joka on siis SharePoint 2010 -asennushakemisto.

Tätä järjestelmähakemistoa kutsutaan myös nimellä ”14 hive”. Se juontuu aikaisemmasta versiosta SharePoint 2007, jossa kansio oli nimeltään ”12 hive”. Se miksei seuraava numero ollut 13, johtuu siitä, että Microsoft pitää sitä huonona enteenä, koska numeroa 13 pidetään epäonnen numerona.

14-kansion alla olevia hakemistoja:

```
%CommonProgramFiles%\Microsoft Shared\Web Server Extensions\ISAPI
```

Tässä hakemistossa sijaitsevat SharePointin tarvitsemat .DLL-kirjastot.

```
%CommonProgramFiles%\Microsoft Shared\Web Server Extensions\14\ADMISAPI
```

Tässä hakemistossa sijaitsevat keskitetyn hallinnan SOAP-palvelut.

```
%CommonProgramFiles%\Microsoft Shared\Web Server Extensions\14\CONFIG
```

Tämä hakemisto sisältää tiedostoja, joita käytetään IIS web -sivustojen laajentamiseen.

```
%CommonProgramFiles%\Microsoft Shared\Web Server Extensions\14\LOGS
```

Tästä hakemistosta löytyvät kaikki asetuslokit sekä ajonaikaiset jäljityslokit.

```
%CommonProgramFiles%\Microsoft Shared\Web Server Extensions\UserCode
```

Tämä hakemisto sisältää tiedostoja, joita käytetään hiekkalaatikkosovellusten tukemiseen.

```
%CommonProgramFiles%\Microsoft Shared\Web Server Extensions\WebClients
```

Tässä hakemistossa on tiedostoja Client Object Modeliin liittyen.

```
%CommonProgramFiles%\Microsoft Shared\Web Server Extensions\WebServices
```

Tämä hakemisto on SharePointin back-end web-palvelujen juurihakemisto, josta esim. Excel- ja Search-palveluja isännöidään.

## 2.7 Keskitetty hallinta

Keskitettyä hallintaa (Central Administration) käyttäen on helpointa muuttaa kaikkia SharePointin asetuksia, koska siinä on graafinen käyttöliittymä. Se ei kuitenkaan ole ainoa tapa, vaan kaikkia asetuksia voidaan toki muuttaa myös käyttäen STSADM / PowerShell -komentokehotetyökaluja (joista lisää luvussa 4.8).

### 2.8 Silverlight

Microsoft Silverlight on sovelluskehys, joka tukee rikkaiden internetsovelluksien kehitystä. Tämä alusta- ja selainriippumaton käyttöliittymäteknologia käyttää omaa ajonaikaista ympäristöään, joka on saatavilla selaimeen lisäosana sekä itsenäisenä ajonaikaisena ympäristönä. Silverlight-sovelluksissa käyttöliittymä määritetään käyttäen XAML:ia (Extensible Application Markup Language) ja ohjelmistokehitys tehdään käyttäen .NET-sovelluskehystä.

Silverlight-sovellukset ovat itsenäisiä. Tästä johtuen SharePointin kanssa voidaan käyttää mitä tahansa versiota Silverlightista. Silverlight-sovelluksien tiedostopääte on .XAP, ja se sisältää XAML-koodia sekä käännetyn lähdekoodin. Ne voidaan sijoittaa SharePointissa useaan paikkaan, mutta dokumenttikirjasto on joustavin vaihtoehto, koska silloin .XAP-tiedostoja päästään hallinnoimaan käyttöliittymän kautta. Suositeltu vaihtoehto on sijoittaa ne kuitenkin globaaliin ClientBin-hakemistoon, koska se antaa .XAP-tiedostoilla erillisen sijainnin. ClientBin-hakemisto sijaitsee 14 hiven ”TEMPLATE/LAYOUTS” -kansion alla.

## 3 SharePoint 2010 -toiminnallisuus

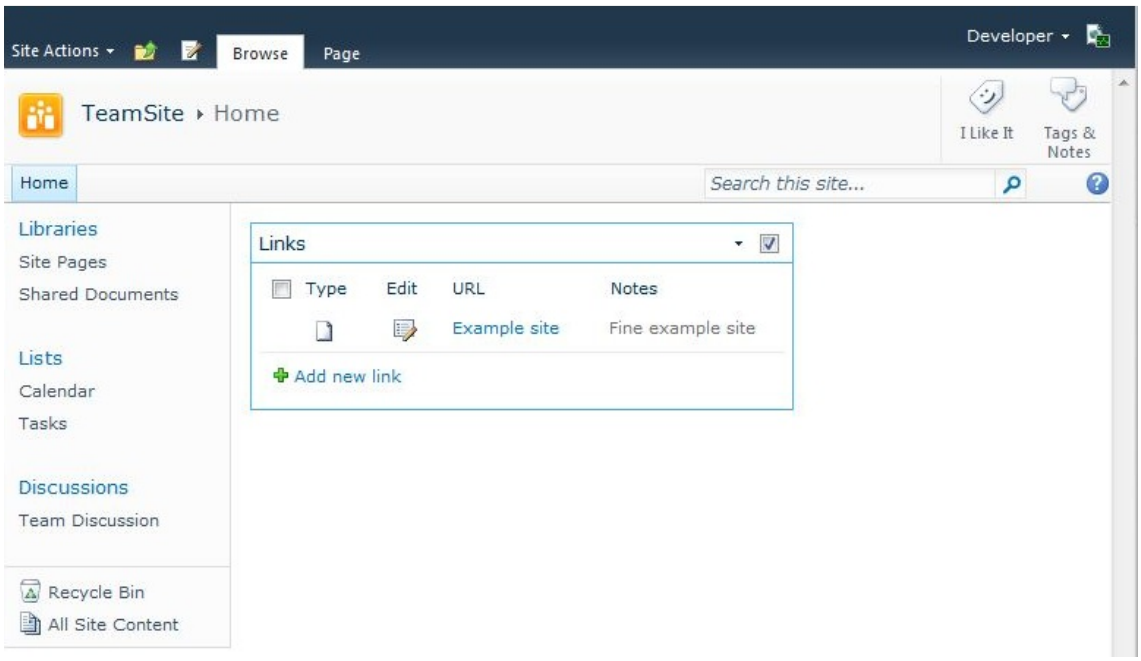
Tässä luvussa käydään läpi SharePoint-tuotteen toiminnallisuuksia. SharePoint sisältää paljon toiminnallisuutta, jota voidaan ja pitää käyttää hyödyksi sellaisenaan tai räätälöinnin pohjana. Toiminnallisuuden tunteminen on ohjelmistokehittäjälle välttämätöntä, jotta osataan käyttää olemassa olevia rakennuspalikoita apuna ohjelmistokehityksessä.

### 3.1 Web-osat

SharePoint 2010:ssä käytetyt Web-osat periytyvät ASP.NET 3.5:stä. Web-osat ovat modulaarisia ja uudelleenkäytettäviä sovelluspalvelinkontrolleja, joita voidaan lisätä loppukäyttäjien selaimista ajonaikaisen ympäristön avulla. Web-osat antavat loppukäyttäjien kontrolloida sisältöä, ulkoasua ja käyttäytymistä sivulla. Web-osia voidaan pudotella määriteltyihin web-osa-alueisiin, jotka toimivat nimenomaan alueina sivulla, mihin web-osia voi järjestellä ja käyttää.

Web-osa koostuu palvelinpään ajovalmiista kontrollista ja web-osan kuvaustiedostosta. Ajovalmis kontrolli pitää sisällään Web-osan toiminnallisuuden ja deskriptori on XML-tiedosto, joka mahdollistaa Web-osan asetusten viennin ja uudelleenkäytön. [3, s. 295.]

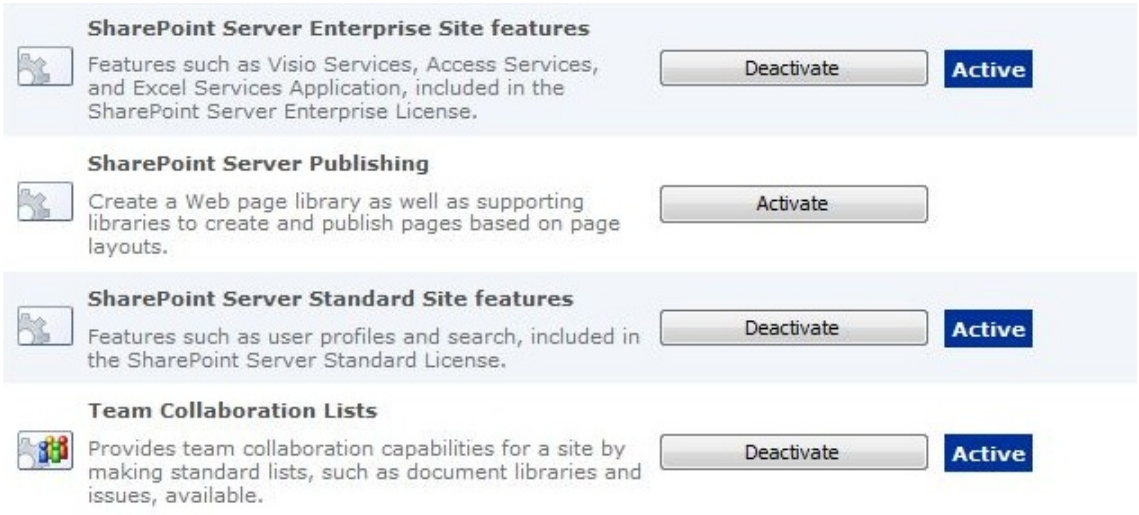
Seuraavana on hyvä esimerkki SharePoint-sivuston näkymästä team site -mallipohjalla (Kuva 1. Web-osa team site -mallipohjalla). Keskellä sivua on linkkien hallintaan käytettävä web-osa.




Kuva 1. Web-osa team site -mallipohjalla

## 3.2 Piirteet

Piirteet ovat erillisiä komponentteja, joissa voi olla monenlaista sisältöä, kuten räätälöityä koodia, listamäärityksiä, tapahtuman käsittelijöitä tai web-osia. Ne voidaan asentaa ja aktivoida erikseen olemassa olevalla SharePoint-sivustolla tai ne voidaan sisällyttää SharePoint-sovellukseen (.WSP-tiedostoon). Piirteet voidaan aktivoida käyttöliittymän kautta sivuston ominaisuuksista (Kuva 2. Sivuston) sekä STSADM.EXE- tai PowerShell -komentoja käyttäen.




Kuva 2. Sivuston ominaisuudet

### 3.2.1 Piirteen asentaminen ja aktivointi STSADM.EXE-työkalulla

1) Kopioidaan piirre hakemistoineen 14 hiven alle hakemistoon ”TEMPLATE\FEATURES”.
2) Avataan komentokehotetyökalu järjestelmän ylläpitäjän oikeuksilla.
3) Suoritetaan komento ”STSADM.EXE -o installfeature -name FeatureHakemis-toNimi”.
4) Aktivoidaan piirre suorittamalla komento ”STSADM.EXE -o activatefeature -name PiirreHakemistoNimi -url http://URLOsoite”.

### 3.2.2 Piirteen asentaminen ja aktivointi PowerShell-komennoilla

Piirteen asentamiseen käytetään PowerShell-komentoa ”Install-SPFeature -Path Feature” ja se aktivoidaan komennolla ”Enable-SPFeature -Identity Feature”.

## 3.3 Hiekkalaatikkosovellukset

Suurin osa SharePointin koodista ajetaan W3WP.EXE-prosessissa. Kuitenkin jotta estetään hiekkalaatikkosovelluksien koodia aiheuttamasta ongelmia W3WP.EXE-prosessille ja siten vaikuttamasta farmin muihin sovelluksiin, hiekkalaatikkosovellukset ajetaan omissa prosesseissaan. Näitä prosesseja kutsutaan myös nimellä ”user code process”, ja ne ovat seuraavat prosessit:

SPUCHostService.exe
SPUCWorkerProcessProxy.exe
SPUCWorkerProcess.exe

### 3.3.1 Hiekkalaatikkosovelluksien suoritusprosessi

Kun hiekkalaatikkosovelluksia ajetaan, ensimmäinen askel prosessissa lähtee siitä, kun SharePointin web-palvelimella sijaitseva suorituksenhallintakomponentti selvittää, onko ajettava koodi hiekkalaatikkosovellustyyppiä. Mikäli suoritettava koodi ei ole hiekkalaatikkosovellustyyppiä, niin koodi ajetaan normaalisti W3WP.EXE-prosessissa.

Jos koodi on osa hiekkalaatikkosovellusta, kontrolli välitetään SPUCHostService.exe-prosessille. Tämän käyttäjäkoodin isännöintipalvelu välittää hiekkalaatikkotyöskentelijäprosessille (SPUCWorkerProcess.exe) tiedot, missä sen haluamat suoritettavat operaatiot validoidaan. Jos koodin validointi onnistuu, varsinainen komentojen ajo suoritetaan yksityiskohtaisen, ei-hiekkalaatikkosovelluksen objektimallin mukaan. [4, s. 11-6.]

### 3.3.2 Hiekkalaatikkosovellukset ja Visual Studio

Uutta SharePoint-projektia luotaessa Visual Studio kysyy, halutaanko tehdä farmisovellus vai hiekkalaatikkosovellus. Sovellustyyppiä voidaan vaihtaa myös projektin luomisen jälkeen asettamalla ”Sandboxed”-ominaisuus joko todeksi tai epätodeksi.

Yrittäessä levittää hiekkalaatikkosovellusta Visual Studio validoi WSP-paketin sisällön eikä levitä sitä, mikäli se sisältää kiellettyjä artifakteja. Mikäli validointitarkistus onnistuu, Visual Studio levittää hiekkalaatikkosovelluksen SharePoint-sivuston sovellusgalleriaan. Lisäksi Visual Studio aktivoi sovelluksen.

Kun hiekkalaatikkosovelluksesta on korjattu virheet, ja se on testattu, Visual Studiolla voidaan paketoida sovellus WSP-tiedostoksi, aivan niin kuin farmisovelluksien kanssa tehdään. Tärkein ero on kuitenkin se, että sivustojen ylläpitäjät voivat yksinkertaisesti ladata WSP-paketin sovellusgalleriaan ja aktivoida sovelluksen ilman, että farmin ylläpitäjän pitää viedä sovelluspaketti erikseen.

## 3.4 Mallipohjat

SharePoint-sivusto on tyypillisesti kokoelma sivuja, listoja ja kirjastoja. Sivusto voi sisältää alisivustoja, ja alisivustot voivat sisältää alisivustoja. Tyypillisesti kaikki sivustot, listat, kirjastot ja mitkä tahansa kustomoitavat objektit pitää luoda tyhjästä. Mallipohjien avulla voidaan kuitenkin luoda ensin yksi mallipohja ja sen jälkeen tarvittavat objektit tästä mallipohjasta.

SharePoint-sovelluskehys sisältää erilaisia mallipohjia, kuten sarake-, lista- ja sivustomallipohjat.

### 3.4.1 Sarakemallipohjat

Sarakemallipohjia käytetään usein, kun samaa saraketta tarvitaan useissa eri listoissa tai sivustoissa. Sarakemallipohjien käyttöönottoa varten tehdään piirre (niistä tarkemmin luvussa 3.2 Piirteet).

### 3.4.2 Listamallipohjat

Listamallipohjien käyttö on järkevää monessa tapauksessa, esimerkiksi kun tarvitaan useita eri instansseja samantyyppisistä listoista samoilla sarakkeilla ja näkymillä. Tällaisissa tapauksissa kannattaa tehdä yksi listamallipohja ja käyttää sitä uudelleen muilla sivustoilla. Listamallipohjia on kahden tyyppisiä, on listamäärityksiä ja listamallipohjia.

Listamallipohjia voidaan luoda SharePointin käyttöliittymästä yksinkertaisesti menemällä listan asetuksiin ja tallentamalla lista mallipohjaksi. Tämä lähestymistapa ei ole kuitenkaan paras mahdollinen, koska kun SharePoint luo listamallipohjan, se voi sisältää viitteitä muihin listoihin ja kenttiin. Niistä voi koitua ongelmia yritettäessä käyttää mallipohjaa kohdejärjestelmässä. Toinen huono puoli listamallipohjassa on rajoitettu laajennusmahdollisuus; siinä voidaan käyttää ainoastaan kenttiä, joita on luotu käyttöliittymästä käsin tai SharePoint Designeria käyttäen.

Listamäärityksien tekemiseen löytyy nimellä ”List Definition” valmis projektimallipohja Visual Studio 2010:stä.

### 3.4.3 Sivustomallipohjat

Sivustomallipohjien avulla voidaan luoda kokonaisia SharePoint-sovelluksia, joita voidaan alustaa kerta toisensa jälkeen. Sivustomallipohjia on myös kahta eri tyyppiä, sivustomallipohjia ja sivustomäärityksiä.

Sivustomallipohjia voidaan luoda joko SharePoint-sovelluksen käyttöliittymän kautta tai käyttämällä SharePoint Designeria. Koko sivuston voi tallettaa mallipohjaksi (.WSP tiedostopäätteellä) ja sitä voidaan jälkikäteen käyttää uudelleen viemällä tiedosto ensin mallipohjagalleriaan. Tällä tavoin tehdyt mallipohjat käyttävät SharePointin perusmallipohjia. Niiden kanssa on oltava tarkkana, että ne täyttävät tiettyjä perusehtoja. Jos esimerkiksi mallipohja on tehty eri kieliversiolla, kuin missä mallipohjaa yritetään käyttää, niin mallipohja ei edes ilmesty valittavaksi yritettäessä luoda uutta sivua. Ongelmien välttämiseksi pitää aina selvittää lähdesivuston kielisyys ja mahdolliset räätälöinnit, jotka voisivat johtaa mallipohjan toimimattomuuteen kohdesivustolla.

Sivustomääritykset sen sijaan ovat XML-tiedostoja, ja ne sisältävät aina kaikki tarpeelliset komponentit – eikä ongelmia pitäisi syntyä (tätä ei pidä ymmärtää väärin, SharePoint-ohjelmistokehityksessä pitää aina varautua yllätyksiin). Sivustomääritykset paketoituivat yhdeksi sovellustiedostoksi (.WSP tiedostopäätteellä), ja ne on näin ollen myös helppo siirtää kohdejärjestelmään.

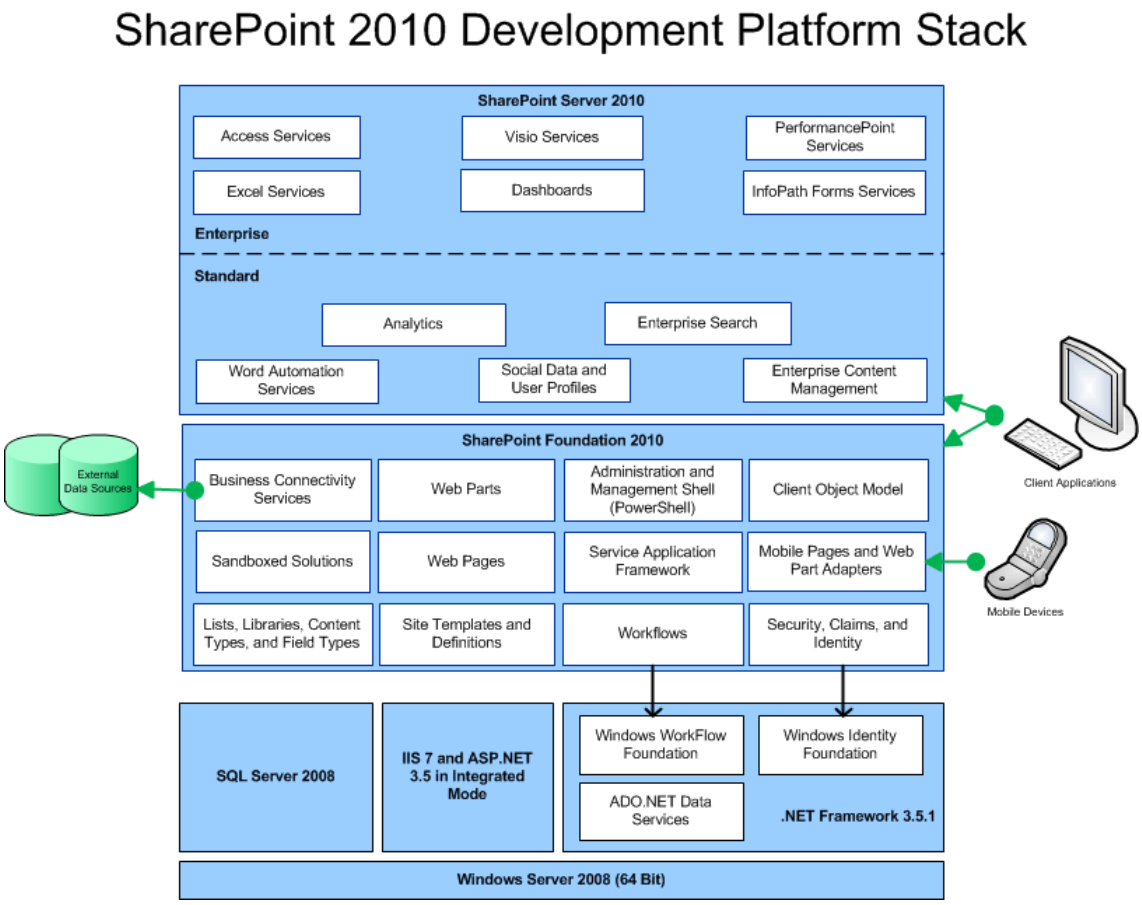
Ohjelmistokehittäjä käyttää mallipohjana tietysti aina sivustomäärityksiä, koska ne ovat luotettavampia ja paremmin laajennettavissa. Sivustomäärityksien tekemistä varten löytyy Visual Studio 2010:estä valmis projektityyppi nimeltä ”Site Definition”. [3, s. 387.]

# 4 SharePoint 2010 ja ohjelmistokehitys

Tässä luvussa käydään läpi tärkeimpiä SharePoint 2010 -ohjelmistokehitykseen liittyviä asioita, kuten kehitystyökaluja, provisiointia ja testausta.

Ennen ohjelmistokehityksen aloitusta SharePoint 2010 -alustalla olisi suotavaa tutustua sen sisältämään vakiotoiminnallisuuteen, jotta ei hukkaa aikaa kehittäessä

toiminnallisuutta, joka on jo olemassa. Lisäksi on hyödyllistä ymmärtää SharePoint 2010 teknologia-alustat (Kuva 3. SharePoint 2010 teknologia-alustat), jotta voidaan hyödyntää aikaisemmin saatu kokemus ja tietotaito mahdollisimman tehokkaasti SharePoint 2010 -ohjelmistokehitysprojekteissa.




Kuva 3. SharePoint 2010 teknologia-alustat [5, s. 2.]

Kuten luvussa 2 kävi ilmi, SharePoint 2010 toimii vain 64-bittisellä alustalla. Ohjelmistokehittäjälle tämä tarkoittaa sitä, että käännökset x64-alustalle pitää tehdä käyttäen 64-bittistä versiota .NET Frameworkista. Vaikka Visual Studio olisi asennettu samalla koneelle kuin SharePoint 2010, todennäköisesti joitain asetuksia pitää muuttaa oikean arkkitehtuurin tukea varten.

Huomionarvoista on myös se, että sovelluskehyksen kohde pitää olla versio 3.5, vaikkakin uudempia versioita olisi saatavilla. Tämä johtuu siitä, että SharePointin lähdekoodit on käännetty sovelluskehyksen versiolle 3.5 eikä se näin ollen ole yhteensopiva version 4.0 kanssa. Se tarkoittaa luonnollisesti myös sitä, että ominaisuudet rajoittuvat siihen, mitä versiosta 3.5 löytyy. Lisäksi se tarkoittaa, että

käytössä on runtimen versio 2.0, koska sekä sovelluskehyksen versioissa 3.0 ja 3.5 runtimen versio on pysynyt samana. Seuraava runtime-versio ASP.NET:ille on 4.0 eikä sitä siis ole mahdollista käyttää SharePoint 2010 -ohjelmistokehityksessä. [5, s. 3.]

## 4.1 Client Object Model

Client Object Modelia voidaan käyttää hyväksi .NET/SilverLight/JavaScript-ohjelmien toteutuksessa. Sen avulla voidaan viitata suoraan SharePointin rakenteeseen ilman, että tarvitsee tehdä kutsuja erinäisten palvelujen kautta. Sitä voidaan käyttää samaan tapaan kuin Server Object Modelia, joskin se on hieman suppeampi versio, jotta kirjastojen koko ja latausajat pysyisivät järkevinä Silverlight/JavaScript-toteutuksissa. Seuraavana esimerkki (Koodilistaus 1) .NET Console Application -sovelluksesta, joka tulostaa ruudulle web-sivun otsikon Client Object Modelia käyttäen.

```
class Ohjelma
{
    static void Main(string[] args)
    {
        ClientContext ctx = new ClientContext(http://esimerkkisaitti.fi);
        Web web = ctx.Web;
        ctx.Load(web);
        ctx.ExecuteQuery();
        Console.WriteLine(web.Title);
    }
}
```

Koodilistaus 1. Client Object Model

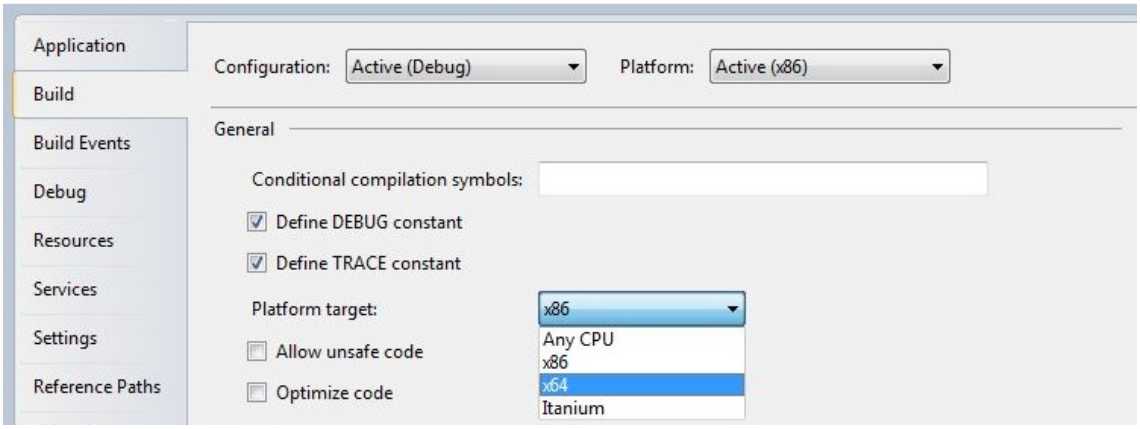
Client Object Model eroaa siinä määrin Server Object Modelista, että alustaessa objekteja ne ovat tyhjiä ja tieto niihin pitää ladatan erikseen ennen kuin ominaisuuksiin voidaan viitata. (Koodilistaus 1, lihavoidut rivit)

## 4.2 Console Applicationin konfigurointi 64-bittiseksi

Luodessa uutta projektia Visual Studio asettaa käännöksen oletusarvoksi 32-bittisen käännöksen ja se pitää muuttaa käsin 64-bittiseksi seuraavalla tavalla:

1) Tehdään uusi projekti käyttäen Console Application -mallia.

2) Valitaan .NET Framework 3.5.
3) Avataan Properties-ikkuna projektista.
4) Valitaan Build-välilehti.
5) Klikataan ”Platform target” -alasvetovalikkoa ja valitaan x64. (Kuva 4. Visual Studion Properties -ikkunan Build-välilehti).
6) Käännetään projekti.




Kuva 4. Visual Studion Properties -ikkunan Build-välilehti

Mikäli (Kuva 4. Visual Studion Properties -ikkunan Build-välilehti) alustaa ”x64” ei löydy alasvetovalikosta, se pitää luoda:

1) Avataan “Build” -valikosta “Configuration Manager”.
2) Halutun projektin kohdalta valitaan Platform sarakkeesta <New...>.
3) Valitaan ”New platform” kohtaan x64.
4) Valitaan ”Copy settings from” alasvetovalikosta x86.

## 4.3 Työkalut

Yleisimmin käytetyt työkalut SharePoint-ohjelmistokehityksessä ovat Microsoft Visual Studio 2010, Microsoft SharePoint 2010 SDK ja Team Foundation Server. Myös Microsoft SharePoint Designer 2010 on uusimman version myötä erittäin tehokas työkalu.

Visual Studio on Microsoftin kehittämä integroitu ohjelmointiympäristö (Integrated Development Environment, IDE), jolla voidaan kehittää niin konsolisovelluksia kuin graafisia käyttöliittymäsovelluksiakin.

SDK (Software Development Kit) on sarja kehitystyökaluja, jotka ovat apuna sovelluskehityksessä. SDK sisältää muun muassa dokumentoinnin, koodiesimerkkejä ja parhaita käytäntöjä.

Team Foundation Server eli lyhyemmin TFS on Microsoftin kehittämä tuote jota käytetään lähdekoodien hallintaan, raportointiin ja projektin seuraamiseen. TFS on käytännössä välttämätön tuote projekteissa, joissa kehittäjiä on useampia.

### 4.3.1 SharePoint Designer

SharePoint Designer 2010 on tehokas työkalu nopeaan SharePoint-sovelluksien kehittämiseen. Kehittäjien keskuudessa se ei ole saavuttanut suosiota johtuen sen tavasta tehdä muutokset suoraan kantaan - huomaamattomasti. SharePoint Designerin kanssa pitää olla aina varovainen, sillä se käsissään osaamaton käyttäjä saa helposti aikaan laajamittaista tuhoa valmiissa toteutuksissa.

Uudempi SharePoint Designer 2010 -versio on kuitenkin huomattavasti aiempaa versiota kehittyneempi ja sitä voivat jopa kehittäjätkin käyttää hyväkseen SharePoint-sovelluskehityksessä.

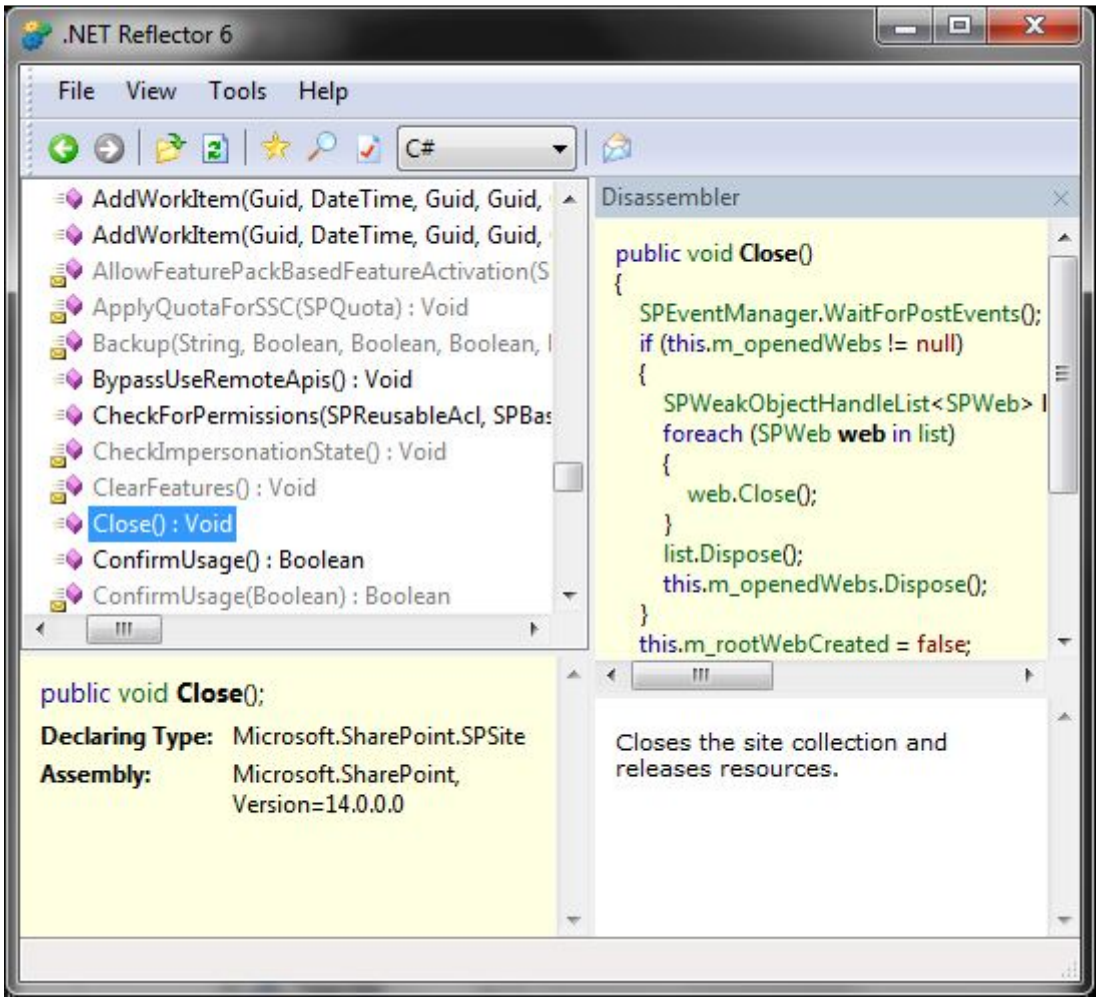
SharePoint Designer 2010:llä voidaan tehdä esimerkiksi vaikkapa työnkulkuja ja viedä ne Visual Studioon jatkokehitystä varten. Sillä voidaan kehittää myös räätälöityä ulkoasua sisältäviä sivuja XSLT, CSS ja muita teknologioita apuna käyttäen.

### 4.3.2 WSPBuilder

Visual Studio 2010:een on olemassa useita kehitystyötä helpottavia lisäosia. Yksi niistä on WSPBuilder, jonka avulla voidaan esim. kääntää, levittää tai poistaa .WSP-paketit. Sen avulla voidaan myös vaikkapa kiinnittyä IIS-työntekijäprosesseihin testausta varten. [6, s. 1.]

### 4.3.3 .NET Reflector

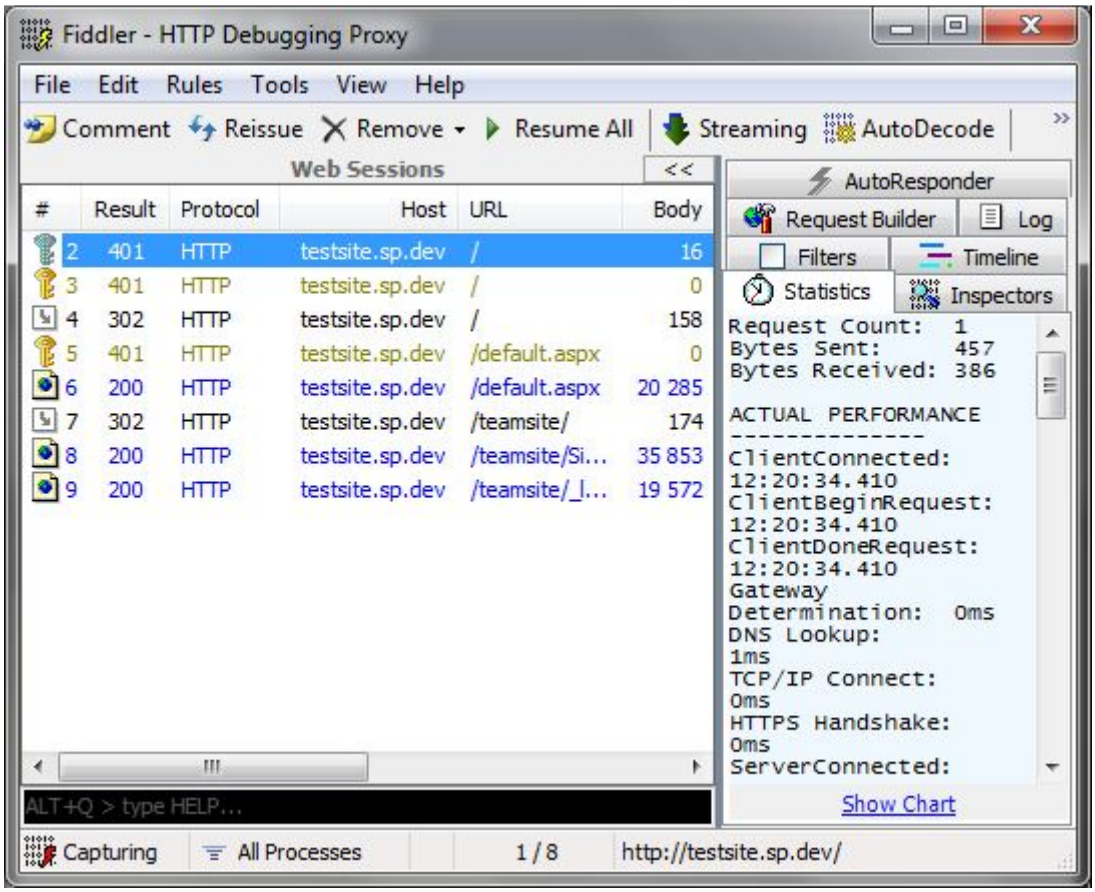
.NET Reflector on yksi tärkeimmistä työkaluista SharePoint-ohjelmistokehityksessä. Sen avulla päästään tutkimaan jo käännettyjä .NET-luokkakirjastoja ja näin ollen saadaan selville, miten mikäkin toiminnallisuus on toteutettu. Välillä se on välttämättömyys esimerkiksi sellaisessa tilanteessa, että halutaan laajentaa jo olemassa olevaa luokkaa, mutta luokka onkin sinetöity (sealed). Tällaisessa tapauksessa yksi tapa tehdä laajennus on ensin selvittää, miten luokka on toteutettu, sitten kopioida toteutus ja lopuksi toteuttaa laajennus. .NET Reflector löytyy kuvasta 5. [7, s. 1.]




Kuva 5. .NET Reflector

### 4.3.4 Fiddler

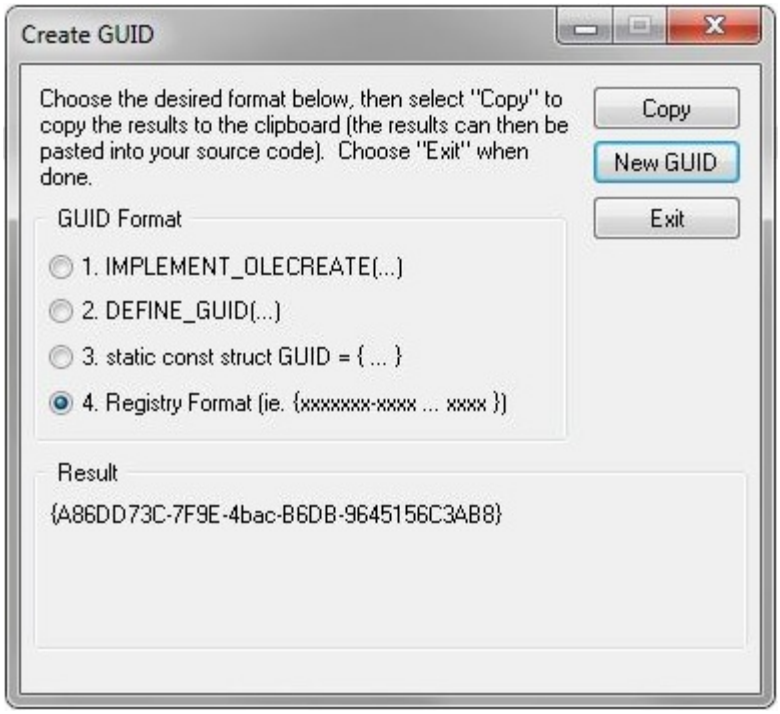
Fiddlerin avulla voidaan tutkia tietoa, mitä linjoja pitkin liikkuu. Sillä nähdään, kun tietoa, otsakkeita ja evästeitä lähetetään palvelimelle ja sieltä takaisin selaimeen. Fiddler on testaustyökalu webiin, jonka voidaan ajatella sijaitsevan selaimen ja verkkopistokkeen välissä. Se kirjoittaa lokia kaikesta sisääntulevasta HTTP- ja HTTPS-liikenteestä. Fiddler on itsenäinen ohjelma ja on yhteensopiva useimpien selainten kanssa. Fiddler löytyy kuvasta 6. [8, s. 1.]




Kuva 6. Fiddler

### 4.3.5 GUID Generator

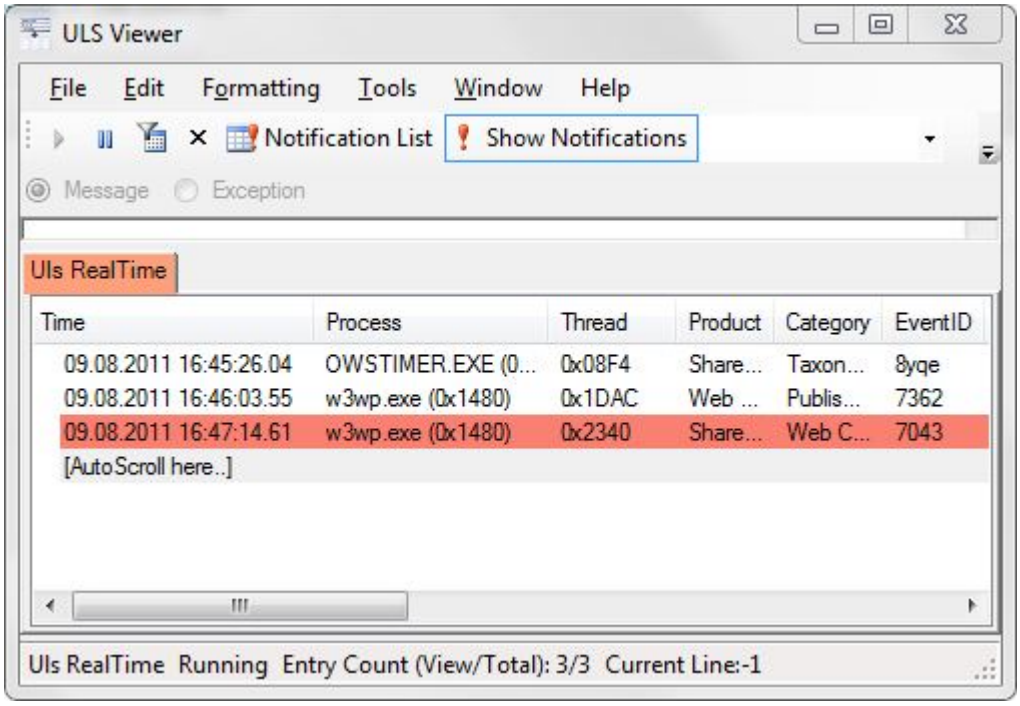
SharePoint käyttää GUID:eja identifoimaan piirteitä, sovelluksia jne. Windows SDK sisältää todella käyttökelpoisen työkalun luomaan valideja GUID:eja. Se on nimeltään Guid Generator (Kuva 7. Guid Generator). Guid Generator -työkalua käyttäessä pitää muistaa aina ensin klikata ”New GUID”, jotta luodaan uusi GUID, ettei käytetä vahingossa vanhaa jo luotua ja mahdollisesti käytössä olevaa GUID:ia.




Kuva 7. Guid Generator

### 4.3.6 ULS Viewer

ULS Viewer on Microsoftin kehittämä erittäin käyttökelpoinen työkalu lokien tutkiskeluun. Sillä voidaan kätevästi suodattaa ja lajitella lokin tapahtumia. Paras ominaisuus on kuitenkin mahdollisuus tarkastella lokeja reaaliaikaisesti ja korostaa virhetapahtumat eri värillä. Näin ollen itse virheet on helppo löytää lokista. ULS Viewer löytyy kuvasta 8. [9, s. 1.]




Kuva 8. ULS Viewer

## 4.4 Sovelluspaketit

Sovelluspaketti on ikään kuin jakelupaketti, jolla voidaan jakaa räätälöity kehitystyö SharePoint 2010 farmipalvelimille. Sovelluspaketteihin voidaan pakata räätälöityjä ominaisuuksia, sivustomäärityksiä, mallipohjia, sivun ulkoasuja, web-osia, CSS-tyylipohjia ja muita elementtejä.

Sovelluspaketti on CAB-tiedosto .WSP-tiedostopäätteellä ja manifest-tiedostolla. Sovelluspakettien luontiin kannattaa käyttää Visual Studio 2010 -työkaluja, mutta sovelluspaketteja voidaan luoda myös manuaalisesti käyttämällä esim. MAKECAB.EXE-työkalua.

Sovelluspakettiin voidaan pakata esimerkiksi .NET-sovelluskehyksen käännöksiä, resurssitiedostoja, piirteitä, räätälöityjä listoja, kirjastoja, kenttiä, sisältölajeja, mallipohjia, sivustomäärityksiä, web-sivuja ja kuvia. [10, s. 1.]

### 4.5 Provisiointi

Provisioinnilla tarkoitetaan sivustokokoelman uudelleenluomista asennus skriptin avulla. Käytännössä skriptit poistavat sivustokokoelman, vetävät paketit pois käytöstä, poistavat paketit, asentavat uudet paketit ja ottavat uudet paketit käyttöön. Tämän lisäksi skripti poistaa ensin sivustokokoelman ja luo sen uudelleen. Kaikki tämä voidaan toki tehdä myös manuaalisesti käyttöliittymän kautta tai käyttäen komentokehotetyökaluja, mutta skriptin rakentaminen on monesti hyödyllisempää ajan säästämiseksi.

Provisiointi on erityisesti ohjelmistokehittäjälle hyödyllinen, aikaa säästävä toimenpide. Ohjelmistokehittäjä voi helposti milloin tahansa luoda sivustot uusilla asennuspaketeilla ja näin ollen testata projektiin tehtyjä muutoksia heti. Mahdollisten virheiden korjaukset voidaan ja on parempi tehdä heti, kun tiedetään, mikä virheen aiheuttaa. Mikäli projektiin tehdään paljon toiminnallisuutta testaamatta aika ajoin, virheen paikantaminen voi usein olla hyvinkin vaikeaa. Tämä johtuu SharePointin tavasta antaa välillä hyvinkin ylimalkaisia virheilmoituksia. Työskennellessä varsinkin XML-tiedostojen kanssa monesti vain tiedostonimi on ainoa asia, joka virheilmoituksesta saadaan selville.

### 4.6 Testaus

Testaus on tärkeä osa SharePoint-ohjelmistokehitystä, niin kuin tietysti mitä tahansa muutakin ohjelmistokehitystä. Testaus SharePoint-ympäristössä on kuitenkin astetta monimutkaisempaa kuin normaalin .NET-sovelluksen testaus. Ohjelmistokehittäjän onkin tunnettava testaukseen liittyvät asiat ja käytännöt SharePoint-ympäristössä, jotta osaa käyttää testaustyökaluja oikein ja tehokkaasti.

IIS on tärkeässä roolissa SharePointin virheenkorjauksessa, koska kaikki pyynnöt ja vastaukset kulkevat sen kautta. On erilaisia tapoja testata IIS:iä, jotta voidaan nähdä, mitä tietoa kulkee selaimelta palvelimelle ja takaisin. Testauksen lisäksi myös suorituskykyasiat pitää ottaa huomioon. Niitä varten olisi hyvä olla olemassa jokin

kuormitustyökalu, jotta voidaan varmistua SharePoint-sovelluksen toimivuudesta myös kuormituksen alla.

Mikäli SharePoint-palvelin ei vastaa odotetulla tavalla, eikä testaustyökalu näytä mitään hyödyllistä ilmoitusta, se johtuu SharePointin oletusominaisuudesta piilottaa kaikki virheilmoitukset sekä korvata poikkeusilmoitukset ja lokit lähes hyödyttömällä tiedolla. Se on suunniteltu loppukäyttäjiä varten, koska heille ei ole tarvetta paljastaa oikeita virheilmoituksia. Ominaisuuden voi kytkeä pois päältä asettamalla seuraavat parametrit kyseisen web-sovelluksen web.config-tiedostoon:

```
<configuration>
  <SharePoint>
    <SafeMode CallStack="true" ... />
    ...
  </SharePoint>
  <system.web>
    <customErrors mode="off" />
    ...
  </system.web>
</configuration>
```

Siitä huolimatta, että kaikki virheet eivät aiheuta poikkeusta eivätkä kaikki viestit ole kovinkaan hyödyllisiä, on muitakin hyödyllisiä keinoja löytää ongelmanlähde:

- SharePointin tapahtumaloki
- SharePointin omat lokitiedostot
- testaustyökalun kiinnittäminen olemassa olevaan prosessiin ja poikkeuksien vahtiminen
- IIS-lokit
- lisäämällä koodin sekaan jäljityspiste ja niiden vahtiminen
- Etätestaus, mikäli kohdekoneella ei ole testaustyökalua asennettuna.

SharePoint kirjoittaa paljon tietoa lokeihin, mikäli se on sallittu SharePointin keskitetystä hallinnasta. On suositeltavaa aktivoida lokeihin kirjoitus kehittämisen ja testaamisen ajaksi. SharePointin lokitiedostot löytyvät seuraavasta hakemistosta:

```
%CommonProgramFiles%\Microsoft Shared\Web Server Extensions\14\LOGS
```

IIS-lokit sisältävät paljon tietoa jokaisesta pyynnöstä ja vastauksesta. Jos on epäiltävissä, että oma koodi aiheuttaa outoa käyttäytymistä ja sitä on mahdotonta selvittää testaustyökalulla, on jäljityspisteen käyttäminen hyvä vaihtoehto. Sitä varten pitää kirjoittaa tietoa koodista trace-provideriin ja asettaa jäljitys päälle kyseisen web-sovelluksen web.config-tiedostosta seuraavilla parametreilla:

```
<configuration>
  <system.web>
    <trace enabled=”true” requestLimit=”40” localOnly=”false” />
  </system.web>
</configuration>
```

IIS-lokit sijaitsevat hakemistossa:

```
%SystemRoot%\System32\LogFiles
```

### 4.6.1 Hiekkalaatikkosovelluksien testaus

Kun Visual Studiossa käytetään testaustoiminnallisuutta muilla kuin hiekkalaatikkosovelluksilla, Visual Studio vie sovelluspaketin farmin sovellusvarastoon ja kiinnittyy automaattisesti W3WP.EXE-prosessiin. Tämä lähestymistapa ei toimi kuitenkaan hiekkalaatikkosovelluksien kanssa, koska niitä ei voi levittää farmin sovellusvarastoon eikä W3WP.EXE-prosessi aja niitä.

Sen sijaan käyttämällä testaustoiminnallisuutta hiekkalaatikkosovelluksen kanssa, Visual Studio vie sovelluspaketin sovellusgalleriaan ja kiinnittyy automaattisesti SPUCWorkerProcess.EXE-prosessiin. Tämä tarkoittaa sitä, että testauksen suorituksen pysäytyskohdat, seurannat, koodin askeltaminen ja kaikki muut testausominaisuudet toimivat normaalisti.

Jos halutaan käyttää testaustoiminnallisuutta sovellukseen, joka on jo viety sovellusgalleriaan, voidaan joko vetää sovellus takaisin ja levittää sovellus uudelleen tai vaihtoehtoisesti voidaan kiinnittyä SPUCWorkerProcess.exe-prosessiin manuaalisesti.

## 4.7 Kehittäjän kojelauta

Monet edelläkäydyistä työkaluista ovat tehokkaita, mutta eivät aina soveliaita tuotantoympäristöön. Kehittäjän kojelauta (Developer Dashboard) on tehty SharePoint 2010:een tuottamaan vähintäänkin perustietoa sisäisistä prosesseista. Oletuksena kehittäjän kojelauta on kytketty pois päältä, sen voi kytkeä päälle käyttäen joko STSADM- tai PowerShell-työkalua. Kehittäjän kojelaudan saa kytkettyä päälle myös koodilla.

### 4.7.1 Kehittäjän kojelaudan aktivointi STSADM-työkalulla

```
Stsadm -o setproperty -pn developer-dashboard -pv ondemand
```

Komentovalitsin ”-pv” ottaa vastaan seuraavia asetuksia: {on/off/ondemand}
”on”- ja ”off”-asetuksilla voidaan yksinkertaisesti joko asettaa kehittäjän kojelauta käyttöön tai pois käytöstä. Asetus ”ondemand” sen sijaan on monesti suositeltava valinta, koska se tekee kojelauta-ikonin sivuston oikeaan yläkulmaan, josta voidaan tilanteen mukaan joko kytkeä kojelauta päälle tai pois.

### 4.7.2 Kehittäjän kojelaudan aktivointi PowerShell-työkalulla

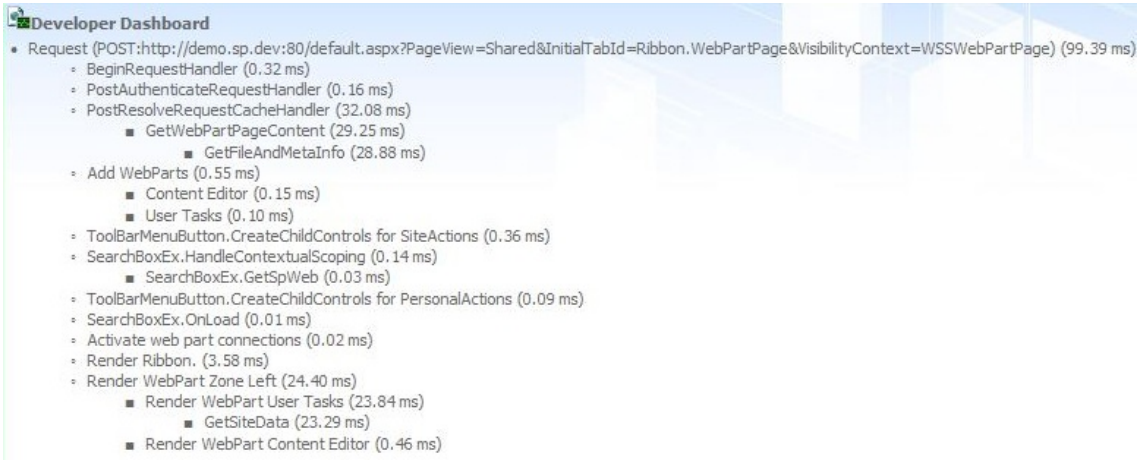
Kehittäjän kojelaudan voi aktivoida PowerShellin kautta seuraavalla skriptillä:

```
$service=[Microsoft.SharePoint.Administration.SPWebService]::ContentService
$addsetting=$service.DeveloperDashboardSettings
$addsetting.DisplayLevel=
[Microsoft.SharePoint.Administration.SPDeveloperDashboardLevel]::OnDemand
$addsetting.Update()
```

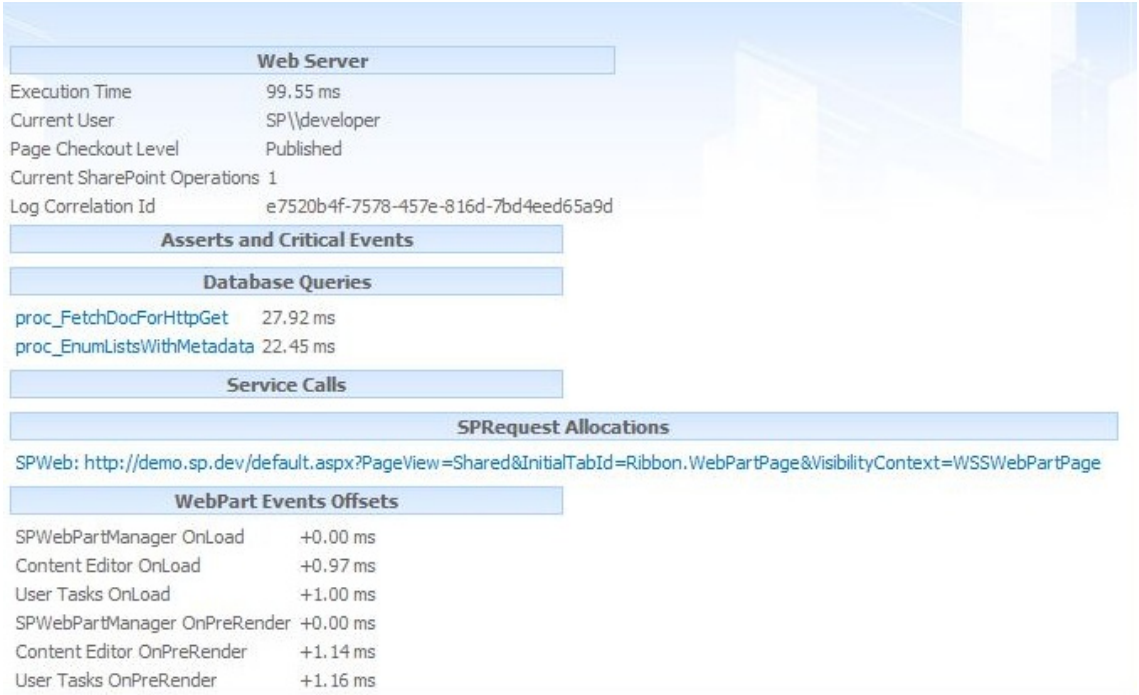
### 4.7.3 Kehittäjän kojelaudan aktivointi koodilla

```
SPWebService svc = SPWebService.ContentService;
Svc.DeveloperDashboardSettings.DisplayLevel = SPDeveloperDashboardLevel.OnDemand;
Svc.DeveloperDashboardSettings.Update();
```

Työkalulla nähdään erittelyn operaatioiden pyyntö/vastaus-syklistä ajankohtineen, tietokantakyselyiden vastausajoista ja jokaisen sivulla olevan web-osan latausajoista (Kuva 8. Kehittäjän kojelauta osa 1/2 ja Kuva 9. Kehittäjän kojelauta osa 2/2).

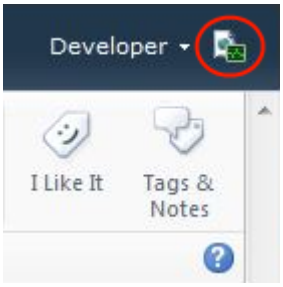



Kuva 8. Kehittäjän kojelauta osa 1/2




Kuva 9. Kehittäjän kojelauta osa 2/2

Mikäli kehittäjän kojelauta on aktivoitu ”OnDemand”-tilaan, niin sitä päästään käyttämään klikkaamalla oikeassa yläkulmassa olevaa kojelautaikonia (Kuva 10. Kehittäjän kojelauta-ikoni).




Kuva 10. Kehittäjän kojelauta-ikoni

Kojelauta analysoi pelkästään nykyisen sivun. Taustalla oleva ASP.NET-moottori pystyy näyttämään paljon enemmän tietoa, mitä itse kojelauta tarjoaa. Se löytyy tiedostosta Trace.axd ja siihen voidaan viitata suoraan URL-osoiteriviltä.

ASP.NET trace näyttää käytännössä kaiken tiedon ajonaikaisesta pyyntö/vastaus-syklistä, kuten headerit (otsakkeet), keksit ja lomaketiedot. Tämä tekee kojelaudasta täydellisen työkalun virheiden etsimiseen, mutta vieläkin hyödyllisemmän suorituskykypuutteiden löytämiseen.

Kojelauta on hyvä apuväline, jolla saadaan paljon hyödyllistä tietoa liittyen ajonaikaiseen ympäristöön. Kuitenkin mahdollisen toimintahäiriön tai huonon suorituskyvyn aiheuttaja saattaakin olla kiinni omasta koodista, joten on hyvien käytäntöjen mukaista testaustyökalujen lisäksi lisätä diagnostista tulostusta koodiin, joka lisää sisältöä kojelaudalle. SPMonitoredScope-luokkaa voidaan käyttää apuna koodissa luomaan lokia ja sisältöä kojelaudalle, joskaan se ei ole käytettävissä hiekkalaatikkosovelluksien kanssa.

### 4.8 Komentokehotetyökalut

SharePointin mukana tulee useita eri komentokehotetyökaluja helpottamaan hallintaa ja automatisointia. Osa työkaluista on erityisesti suunniteltu helpottamaan ohjelmistokehittäjien arkea.

#### 4.8.1 PSCONFIG.EXE

PSCONFIG.EXE-työkalulla voidaan konfiguroida SharePoint-asennusta tai korjata epäonnistunut asennus. Se on sama kuin graafinen ”SharePoint Products and Technologies Configuration Wizard”, mutta komentokehotepohjaista työkalua voidaan ajaa eräajo-tiedostoissa ja näin ollen tehdä automatisointia.

#### 4.8.2 STSADM.EXE

STSADM.EXE on kaikista hallintatyökaluista tärkein. Melkein kaikki hallintaan liittyvät tehtävät voidaan tehdä tällä komentokehotepohjaisella työkalulla. Työkalua on myös mahdollista laajentaa lisäämällä omia komentoja, jotka helpottavat järjestelmän ylläpitäjiä, kun kaikki hallinta voidaan tehdä yhden ja saman työkalun kautta.

SharePoint 2007:ssa tämä oli ainoa komentokehotepohjainen työkalu, mutta SharePoint 2010:n mukana tulee tämän työkalun lisäksi tuki myös PowerShell-työkalun käytölle.

Huomionarvoista on myös se, että monia asioita ei voida tehdä ollenkaan SharePointin keskitetyn hallinnan tai sivuston käyttöliittymän kautta, vaan ainoa tapa suorittaa operaatiot on käyttää STSADM.EXE-työkalua.

STSADM.EXE on tehokas työkalu, ja jokaisen SharePoint 2010 -järjestelmän ylläpitäjän on hallittava sen käyttö. On kuitenkin suositeltavaa käyttää PowerShell-työkalua, koska ennemmin tai myöhemmin Microsoft siirtää kaikki hallintatyökalut PowerShellin komennoiksi ja STSADM.EXE-työkalu jää historiaan.

### 4.8.3 PowerShell

PowerShell on Microsoftin kehittämä työkalu tehtävien automatisointia varten. Se koostuu komentokehotepohjaisesta käyttöympäristöstä ja .NET-sovelluskehys-skriptikielestä.

PowerShellin tehtävät ajetaan cmdlet-komentoina, jotka ovat .NET-luokkien tiettyjä komentoja. Cmdlet-komentoja voidaan yhdistää skripteiksi tai omiksi suoritettaviksi ohjelmikseen.

PowerShelliä voidaan lokaalin käytön lisäksi käyttää myös etäpohjaisissa Windows-ratkaisuissa.

Jotta PowerShellillä voidaan hallita SharePointia, pitää sitä varten ladata oma lisäosa seuraavalla komennolla:

```
Add-PSSnapin Microsoft.SharePoint.PowerShell
```

### 4.8.4 SPMETAL.EXE

SPMETAL.EXE-komentokehotetyökalun avulla voidaan sallia LINQ to SharePointin käyttö olemassa oleviin listoihin ja kirjastoihin. Se tapahtuu luomalla työkalulla proxy-luokka. Työkalu voidaan asettaa ajamaan myös prebuild-komentona Visual Studiosta.

## 4.9 Graafiset työkalut

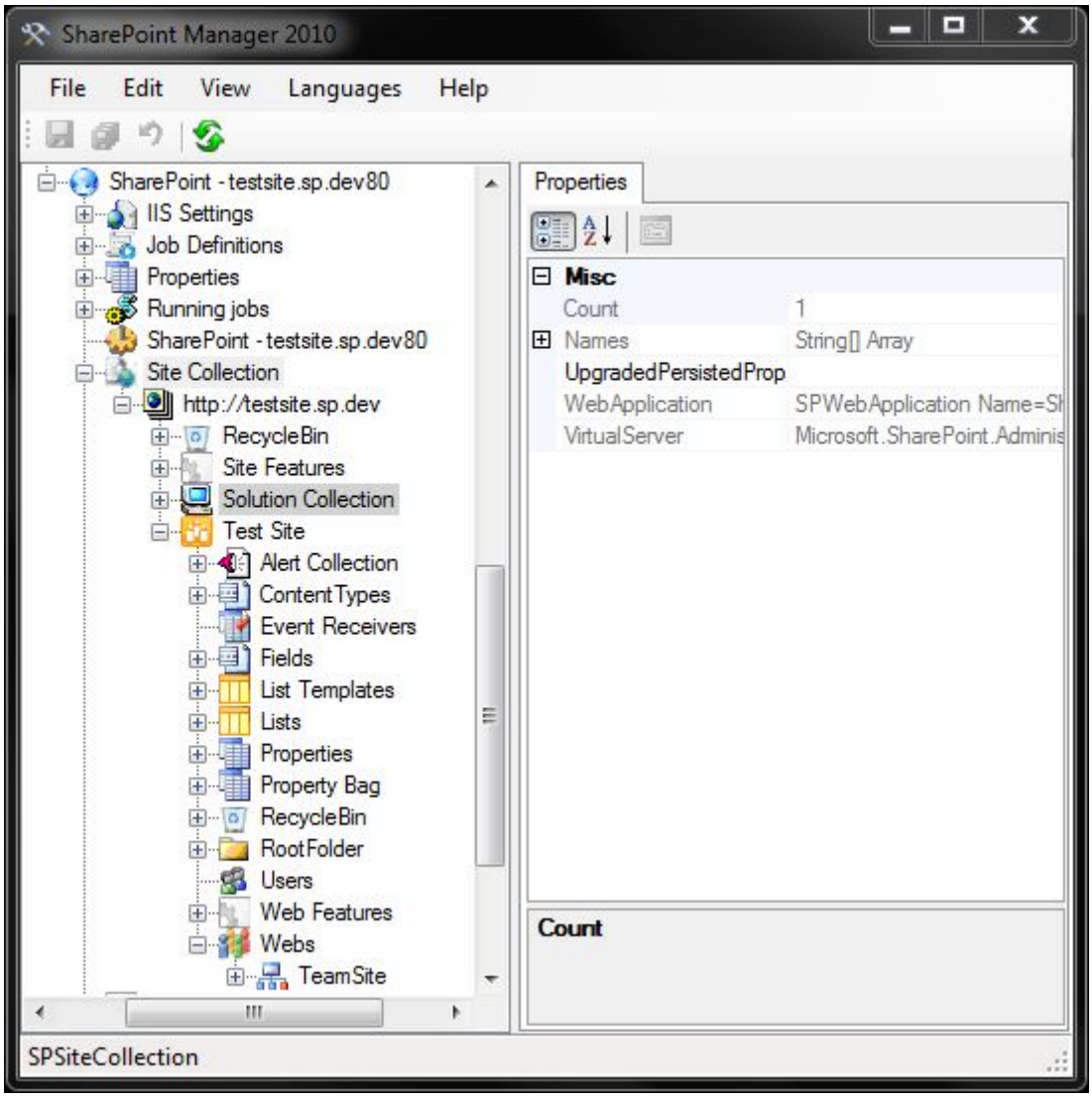
Graafiset työkalut ovat järjestelmän ylläpitäjien ja kehittäjien apuna SharePointin asennuksessa ja konfiguroinnissa.

### 4.9.1 PSCONFIGUI.EXE

PSCONFIGUI.EXE on ”SharePoint Products and Technologies Configuration Wizard”, joka tulee vastaan asennusvaiheessa ja löytyy asennuksen jälkeen Käynnistä-valikon alta. Sillä voidaan tehdä tuoreen SharePoint-asennuksen lisäksi peruskonfigurointeja sekä korjata rikkinäinen asennus.

### 4.9.2 SharePoint Manager 2010

SharePoint Manager 2010:n avulla päästään käsiksi SharePointin objektimalliin. Sillä voidaan selata paikallisen farmin sivustoja ja nähdä niiden ominaisuuksia helposti. Myös ominaisuuksien muokkaus on työkalun avulla mahdollista. (Kuva 11. SharePoint Manager 2010). [11, s. 1.]




Kuva 11. SharePoint Manager 2010

## 4.10 Global Assembly Cache

Global Assembly Cache eli GAC, on koneen laajuinen .NET-kirjastojen (DLL) välimuisti, jonka tiedostot sijaitsevat uusimmissa Windows-versioissa seuraavassa hakemistossa:

```
%SystemRoot%\Assembly
```

## 4.11 Ylläpidettävyys

Ylläpidettävyyden kannalta SharePoint-ohjelmistoprojekteissa, niin kuin tietysti missä tahansa muissakin ohjelmistoprojekteissa, on tärkeää muodostaa projektille järkevä rakenne. Käytännössä tämä tarkoittaa samantyyppisten asioiden niputtamista omiin projekteihinsa ja näin ollen myös omiin paketteihinsa. Esimerkiksi apufunktiot,

sivustomääritykset, käyttöliittymäkomponentit, resurssit ja piirteet voisi olla järkevää niputtaa omiin projekteihinsa. Jos johonkin näistä tulee muutos, niin ei tarvitse päivittää koko sivustoa, vaan vain muutetun osan päivitys riittää.

Varsinkin resurssitiedostojen eriyttäminen omaan projektiinsa on olennaisen tärkeää olettaen, että resurssitiedostoja käytetään. Resurssitiedostoilla tarkoitetaan esimerkiksi käyttäjälle näkyvien tekstien sijoituspaikkaa. Eli sen sijasta, että koodataan mahdollisesti myöhemmin vaihtuvia tekstejä koodin sekaan, käytetään resurssiviitteitä ja määritetään teksti erillisessä resurssitiedostossa. Näin tekstien kontrolli on yhdessä paikassa ja yhdessä projektissa. Muutoksien tullessa on helppo päivittää vain yksi paketti sen sijaan, että luodaan koko sivusto uudestaan. SharePointissa tietysti myös monikielisyyden tuen takia on syytä käyttää resurssitiedostoja.

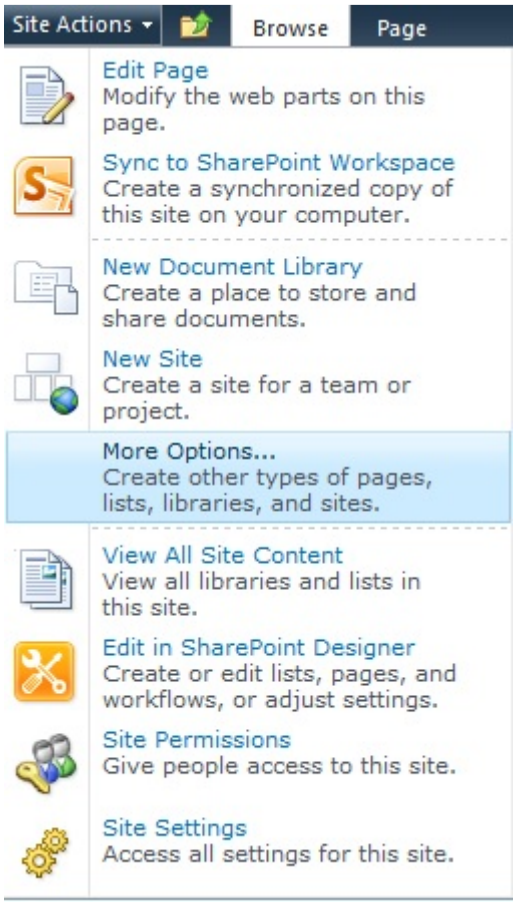
# 5 Esimerkki: Listan luonti manuaalisesti ja ohjelmallisesti

Tässä luvussa luodaan esimerkkinä lista kahdella eri tavalla. Tarkoituksena on esittää, miten eri lähestymistapoja voidaan käyttää SharePointin kanssa.

## 5.1 Custom List -listan luonti suoraan käyttöliittymästä

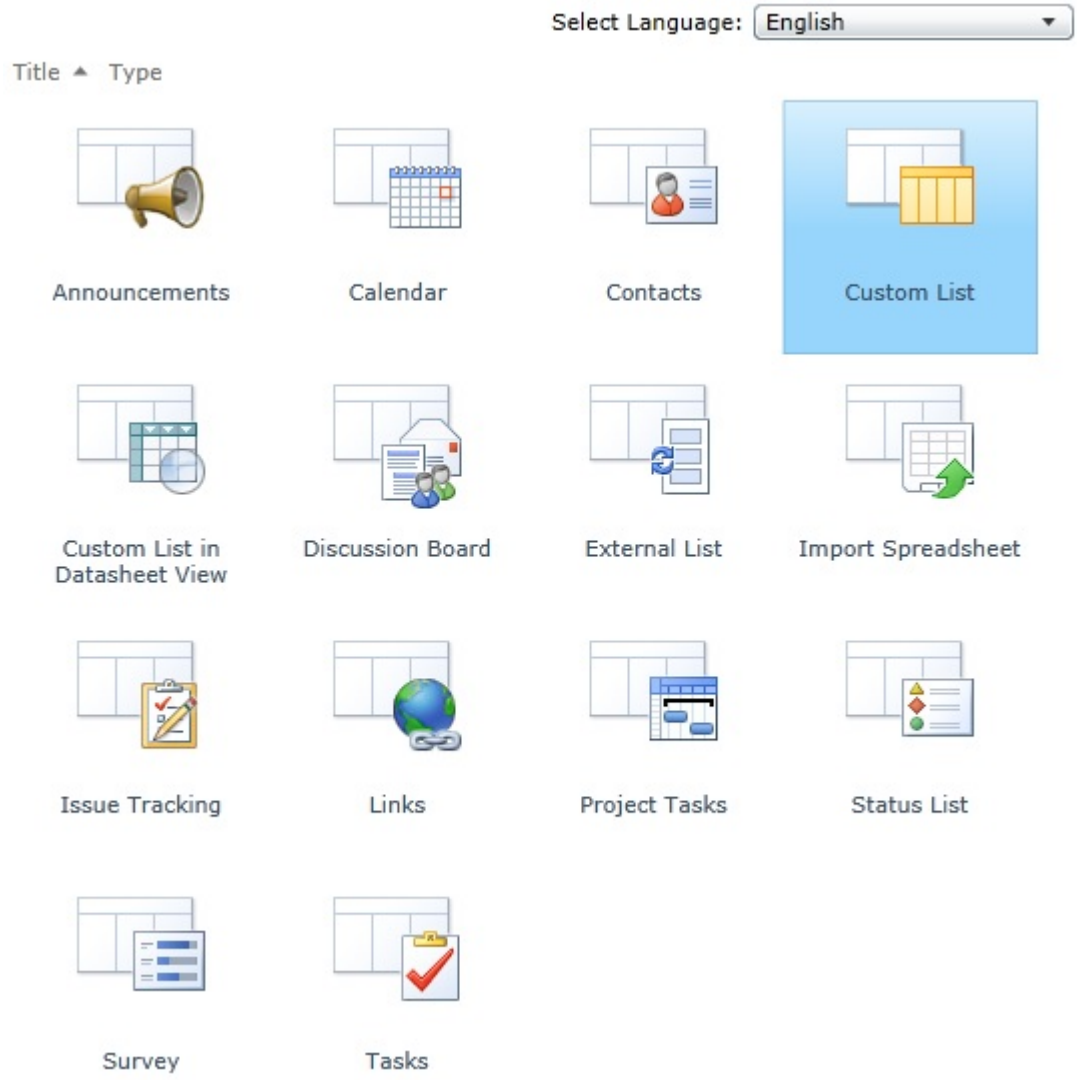
Listan luonti käyttöliittymästä on ehdottomasti helpoin tapa. Tätä tulisi käyttää, mikäli se täyttää vaatimukset.

1) SharePointin etusivulla: ”Site Actions” -> ”More Options” (Kuva 122. Site Actions -valikko)




Kuva 122. Site Actions -valikko

2) Listasta valitaan ”Custom List” sekä määritellään listan nimi (Kuva 133. ”Create” -ikkuna)



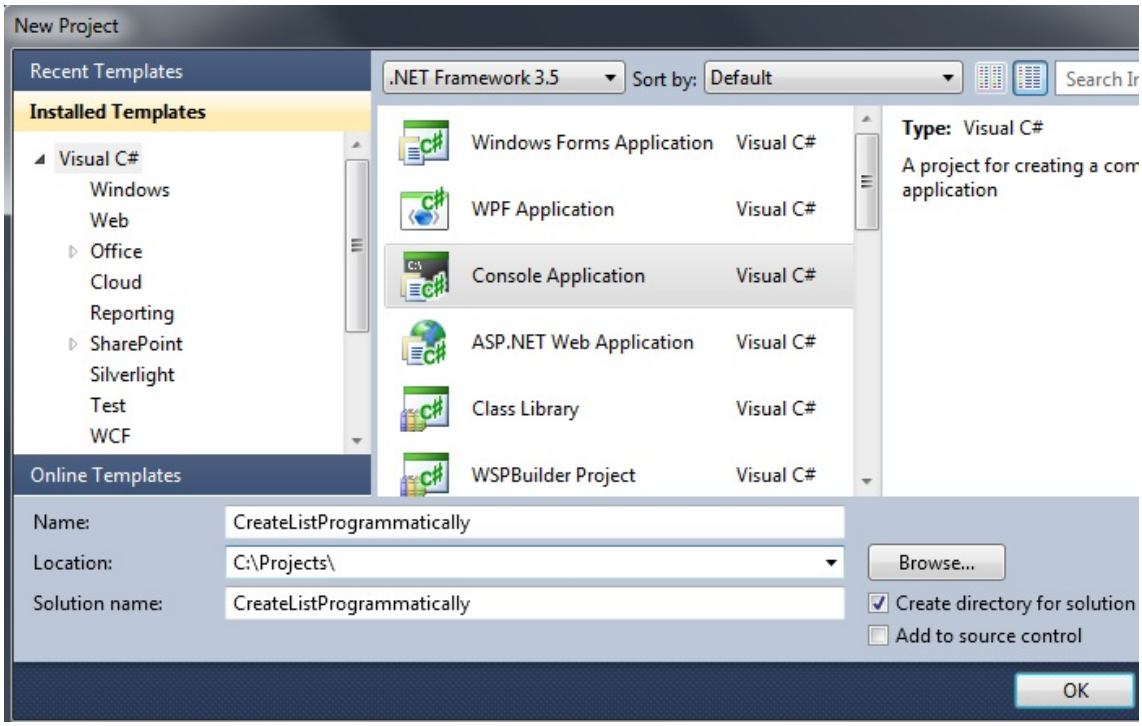

Kuva 133. ”Create” -ikkuna

3) SharePoint luo listan osoitteeseen http://testsite.sp.dev/Lists/CustomListCreatedFromUI

## 5.2 Custom List -listan luonti ohjelmallisesti koodilla

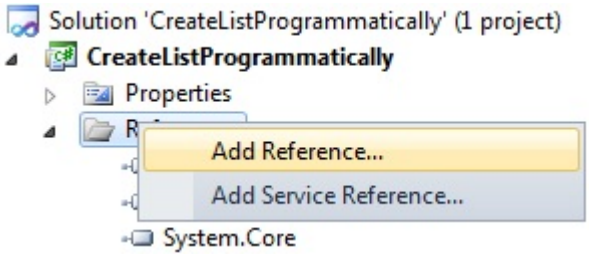
Luodaan Visual Studio:lla uusi sovellus ”Console Application”

1) Ensin luodaan Visual Studiolla uusi projekti (File -> New -> Project) ja valitaan ”Console Application”. (Kuva 144. ”New Project” -ikkuna)




Kuva 144. ”New Project” -ikkuna

2) Lisätään viite ”Microsoft.SharePoint.dll” -kirjastoon.
    a. Klikataan hiiren oikealla napilla ”References” ja klikataan ”Add reference”. (Kuva 155. ”Add Reference” -ikkuna)



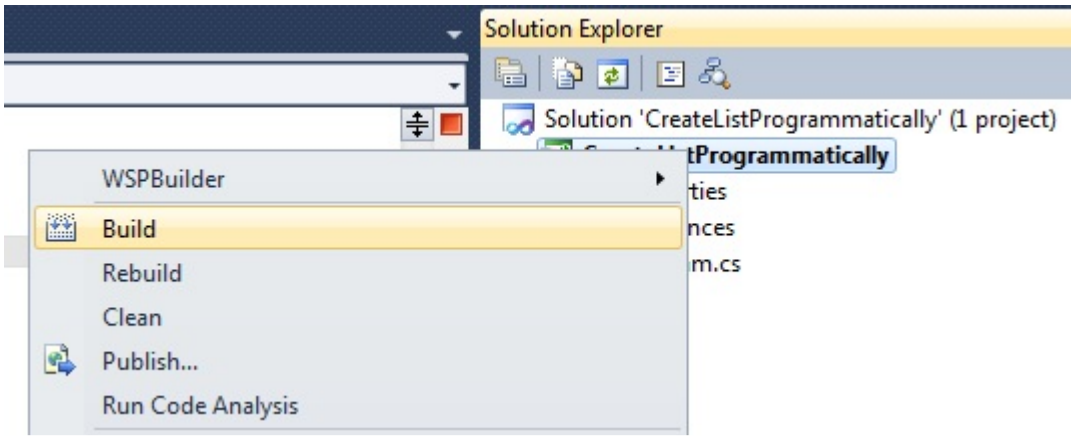

Kuva 155. ”Add Reference” -ikkuna

    b. Haetaan SharePointin ISAPI-hakemistosta (6.2 Hakemistorakenne) ”Microsoft.SharePoint.dll” -tiedosto ja lisätään se viitteisiin.
3) Lisätään ”Program.cs” -tiedostoon koodilistaus 2:n mukaiset rivit.

```
using Microsoft.SharePoint;

namespace CreateListProgrammatically
{
    class Program
    {
        static void Main(string[] args)
        {
            using (SPSite site = new SPSite("http://testsite.sp.dev"))
            {
            SPWeb web = site.OpenWeb();
            SPListCollection lists = web.Lists;

             // Luodaan uusi GenericList-tyyppinen lista nimeltään
             "CustomListCreatedProgrammatically"
            lists.Add("CustomListCreatedProgrammatically", "Custom List -
            luotu ohjelmallisesti", SPListTemplateType.GenericList);

            SPList list = web.Lists["CustomListCreatedProgrammatically"];
            }
        }
    }
}
```

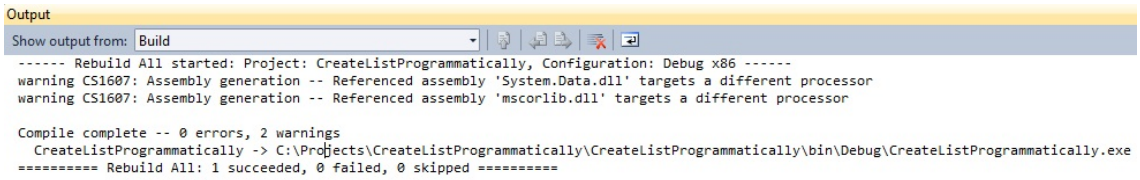
Koodilistaus 2. Listan luonti koodilla

4) Käännetään projekti klikkaamalla hiiren oikealla napilla projektin päällä ja valitsemalla ”Build” (Kuva 166. Projektin kääntäminen).




Kuva 166. Projektin kääntäminen

5) Varmistetaan, että projekti kääntyi ilman virheitä (View -> Output). (Kuva 177. ”Output” -ikkuna)

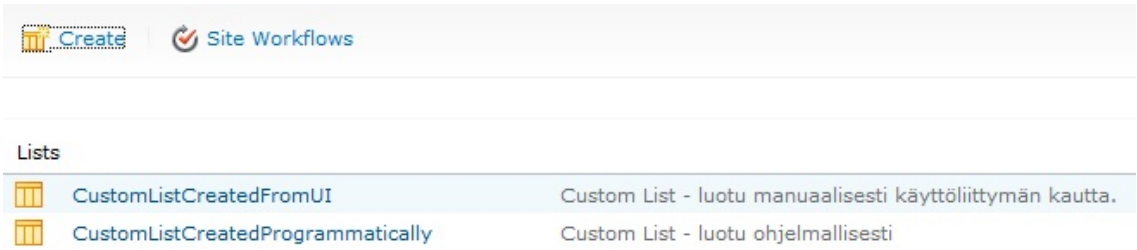



Kuva 177. ”Output” -ikkuna

6) Ajetaan käännetty ohjelma ”CreateListProgrammatically.exe” -konsolista.
7) Lista on luotu ja löytyy osoitteesta
   http://testsite.sp.dev/Lists/CustomListCreatedProgrammatically

## 5.3 Esimerkkien yhteenveto

Esimerkeissä luotiin lista ensin suoraan käyttöliittymästä ja sen jälkeen koodilla. Kummallakin lähestymistavalla muodustui identtinen lista SharePointin Lists-kirjastoon. (Kuva 188. SharePointin ”Lists” -kirjaston sisältö.)




Kuva 188. SharePointin ”Lists” -kirjaston sisältö.

SharePointilla voi lähes aina tehdä saman asian usealla eri tavalla, ja se pitää huomioida ennen kuin aloitetaan itse tekeminen. Käyttöliittymästä asiat saa tehdyksi todella nopeasti, kun taas koodilla tehtäessä vähänkään monimutkaisempaa sovellusta kuluu aikaa moninkertainen määrä. Tämän takia on tärkeää tiedostaa, mitkä asiat kannattaa tehdä suoraan käyttöliittymästä, mitkä ohjelmallisesti ja mitkä jollain muulla mahdollisella tavalla.

# 6 Yhteenveto

Tekijä tutustui SharePoint 2010 -tuotteeseen syksyllä 2010 ja on toiminut siitä lähtien SharePoint-ohjelmistokehittäjänä. Opinnäytetyö on aloitettu alkukeväästä 2011 jälkeen, jolloin kokemukset alkutaipaleelta olivat vielä tuoreena mielessä.

SharePoint 2010 on hyvin monipuolinen tuote, mutta sen räätälöinti saattaa aiheuttaa ohjelmistokehittäjälle suurta päänvaivaa, koska ihan kaikkea ei vaan yksinkertaisesti pysty toteuttamaan.

SharePoint-ohjelmistokehityksessä törmää jatkuvasti uusiin haasteisiin ja tuotteen rajoituksiin. Iso osa haasteista on kuitenkin ratkaistu ja internetistä löytyykin paljon hyviä blogeja ja artikkeleita aiheeseen liittyen.

Ennen työkalujen käteen ottamista pitää tuntea SharePointin olemassa olevat toiminnallisuudet ja rajoitukset. Vielä senkin jälkeen kannattaa käyttää tovi tiedon hakuun internetissä, ettei turhaan lähdetä ratkaisemaan asioita, jotka joku muu on jo ratkaissut.

Tämän opinnäytetyön tavoitteena oli toimia perehdyspakettina aloittelevalle SharePoint-ohjelmistokehittäjälle. Työhön on pyritty kokoamaan kaikki SharePoint-ohjelmistokehitystä koskevat asiat sekä työtä helpottavat työkalut.

Lopuksi opinnäytetyössä käytiin läpi kahdella esimerkillä, mitä eri lähestymistapoja SharePointissa on.

## Lähteet


1 Hardware and software requirements (SharePoint Server 2010). 8.7.2010. Verkkodokumentti. Microsoft TechNet. <http://technet.microsoft.com/en-us/library/cc262485.aspx>. Luettu 3.5.2011.

2 Plan browser support (SharePoint Server 2010). 7.9.2011. Verkkodokumentti. Microsoft TechNet. <http://technet.microsoft.com/en-us/library/cc263526.aspx>. Luettu 4.4.2011.

3 Döring Martin, Krause Joerg, Langhirt Christian, Pehlke Bernd, Sterff Alexander. 2010. SharePoint 2010 as a Development Platform. APRESS. Luettu 5.6.2011.

4 10175A, Microsoft® SharePoint® 2010, Application Development. 2010. Luettu 17.4.2011.

5 SharePoint 2010 Architectures Overview. 2011. Verkkodokumentti. Microsoft Developer Network <http://msdn.microsoft.com/en-en/library/gg552610.aspx>. Luettu 20.5.2011.

6 WSPBuilder. 30.5.2011. Työkalu SharePointin sovelluspakettien luontiin. Keutmann. <http://wspbuilder.codeplex.com/>. Luettu 24.5.2011.

7 .NET Reflector. Työkalu .NET-käännöksien analysointiin. Lutz Roeder. <http://www.reflector.net>. Luettu 23.3.2011.

8 Fiddler. Työkalu HTTP-liikenteen tutkiskeluun. Eric Lawrence. <http://www.fiddler2.com/fiddler2>. Luettu 25.3.2011.

9 ULS Viewer. 9.10.2009. Työkalu lokien tarkasteluun. Microsoft. <http://archive.msdn.microsoft.com/ULSViewer>. Luettu 18.5.2011.

10 Deploy solution packages (SharePoint Server 2010). 12.5.2010. Verkkodokumentti. Microsoft TechNet. <http://technet.microsoft.com/en-us/library/cc262995.aspx>. Luettu 28.4.2011.

11 SharePoint Manager. 30.5.2011. Työkalu SharePoint-sivustojen selaamiseen. Keutmann. <http://spm.codeplex.com/>. Luettu 22.4.2011.