



# SHAREPOINT 2010 INTEGRATION BEST PRACTICES

TECHNICAL WHITE PAPER  
NOVEMBER 2011

COPYRIGHT © 2011



## INDEX

<b>1</b>	<b>INTRODUCTION.....</b>	<b>3</b>
<b>2</b>	<b>BUSINESS CONNECTIVITY SERVICES .....</b>	<b>4</b>
2.1	INTRODUCTION .....	4
2.2	OVERVIEW .....	4
2.2.1	<i>What is the BCS feature.....</i>	4
2.2.2	<i>Authentication .....</i>	5
2.2.3	<i>Availability .....</i>	5
2.2.4	<i>Development tools.....</i>	6
2.2.5	<i>Limitations.....</i>	7
2.3	GUIDELINES .....	8
2.3.1	<i>Authentication .....</i>	8
2.3.2	<i>Connections.....</i>	10
2.4	CONCLUSION.....	16
<b>3</b>	<b>SEARCH.....</b>	<b>17</b>
3.1	INTRODUCTION .....	17
3.2	OVERVIEW .....	17
3.2.1	<i>Query Architecture.....</i>	18
3.3	GUIDELINES .....	19
3.3.1	<i>Federated Search .....</i>	19
3.3.2	<i>SharePoint Server Search .....</i>	24
3.4	CONCLUSION.....	27
<b>4</b>	<b>RECORDS MANAGEMENT .....</b>	<b>28</b>
4.1	INTRODUCTION .....	28
4.2	OVERVIEW .....	28
4.3	GUIDELINES .....	31
<b>5</b>	<b>SERVICE LAYERS.....</b>	<b>32</b>
5.1	INTRODUCTION .....	32
5.2	OVERVIEW .....	32
5.2.1	<i>Why a service layer? .....</i>	32
5.3	MOST COMMONLY USED SERVICE LAYERS .....	34
5.3.1	<i>SharePoint Service Applications .....</i>	34
5.3.2	<i>BizTalk Server (ESB).....</i>	35
5.3.3	<i>Duet Enterprise .....</i>	36
5.4	CONCLUSION.....	37
<b>6</b>	<b>SUMMARY .....</b>	<b>38</b>
<b>7</b>	<b>REFERENCES.....</b>	<b>39</b>
<b>8</b>	<b>ABOUT THE AUTHORS AND MOTION10.....</b>	<b>40</b>

## 1 INTRODUCTION

Microsoft has positioned SharePoint 2010 in the market as the information worker's dashboard. The command center from where you can find, view and edit information not only stored in SharePoint but also in many line of business (LOB) systems.

With that in mind, systems integration can play the greater role in SharePoint projects. Luckily, SharePoint offers some great tools to communicate with LOB systems.

This technical white paper contains guidelines on how to use these SharePoint tools with the best practices in mind. We will discuss Business Connectivity, Search, Records Management and Service Layers.

Architecting, designing and implementing your enterprise portal with the best practices described in this paper in mind will greatly improve the chance that you will end up with a manageable, scalable and future proof integrated SharePoint environment.

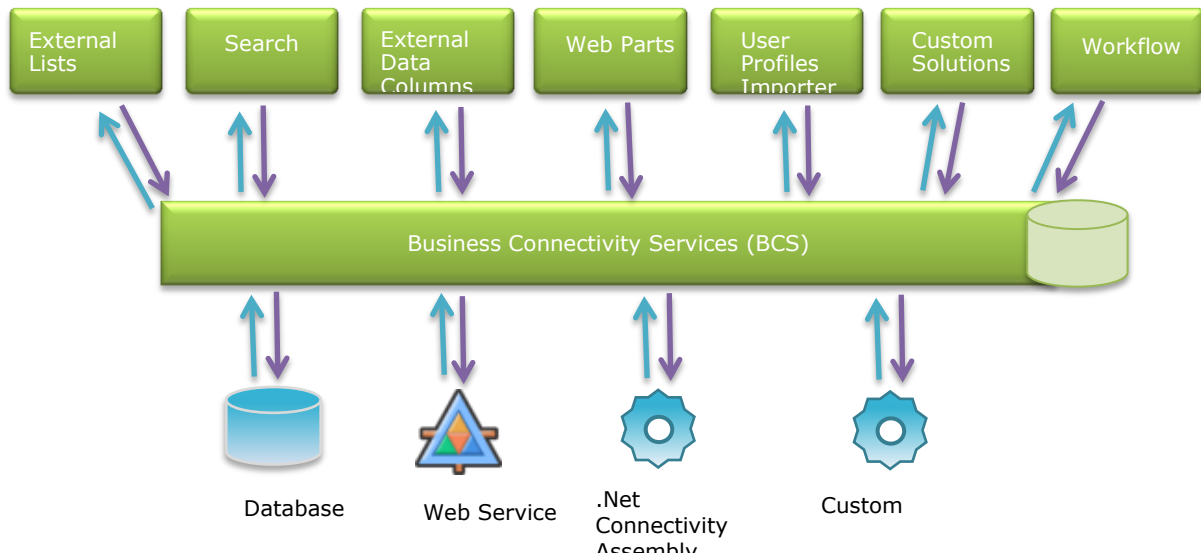
## 2 BUSINESS CONNECTIVITY SERVICES

### 2.1 INTRODUCTION

Business Connectivity Services (BCS) is the successor to the SharePoint 2007 Business Data Catalog (BDC). BCS comes with a lot of enhancements in order to simplify and enhance the process of getting LOB information to show up inside your SharePoint 2010 environment. This chapter will start with an overview of the BCS feature and continues to discuss some well known, and some lesser known options and guidelines to make BCS work for you. We will not discuss the BCS Client connectivity possibilities.

### 2.2 OVERVIEW

#### 2.2.1 WHAT IS THE BCS FEATURE



**Figure 1**

BCS can be considered a broker between your LOB systems and SharePoint. Note that the arrows in Figure 1 are pointing in both directions, which tells us that we can perform both read and write actions to your LOB systems. Another thing to note is that we can now create our own .Net Connectivity Assembly to invoke all actions to the LOB systems and even a Custom Connector that allows us to write our own External Content Type Definition interpreter. This enables us to connect to any system that exposes a web service or for which a .Net provider is available.

2.2.2 AUTHENTICATION

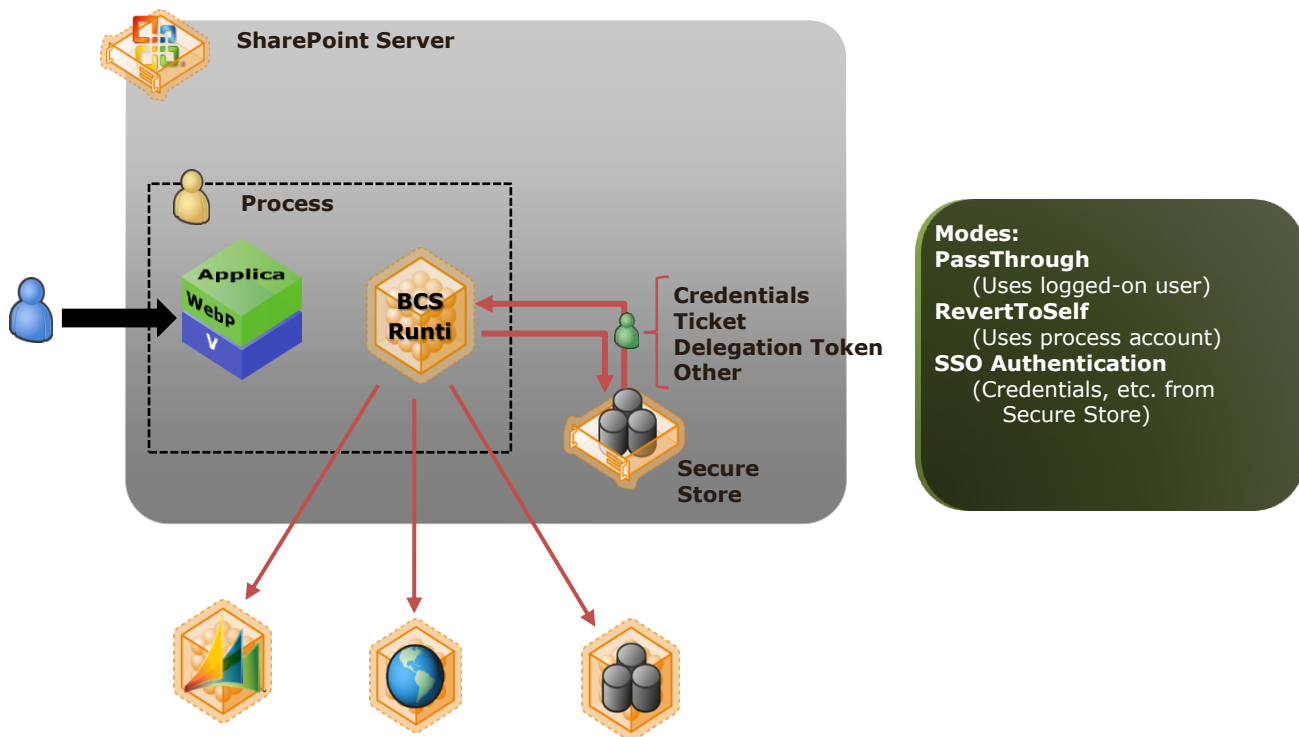


Figure 2

BCS comes with multiple options to authenticate ourselves to our LOB systems. We can provide the credentials of the currently logged on user, the process account or fetch the credentials from the Secure Store Service.

2.2.3 AVAILABILITY

BCS comes with SharePoint Foundations which in turn comes for free with your operating system license. So BCS is virtually a free product. One important thing to note though is that SharePoint Foundations does not come with a Secure Store Service. This seriously limits your authentication possibilities. This issue can be solved however by installing Microsoft Search Server Express 2010, which is a free add on to SharePoint Foundation that also adds the Secure Store Service.

2.2.4 DEVELOPMENT TOOLS

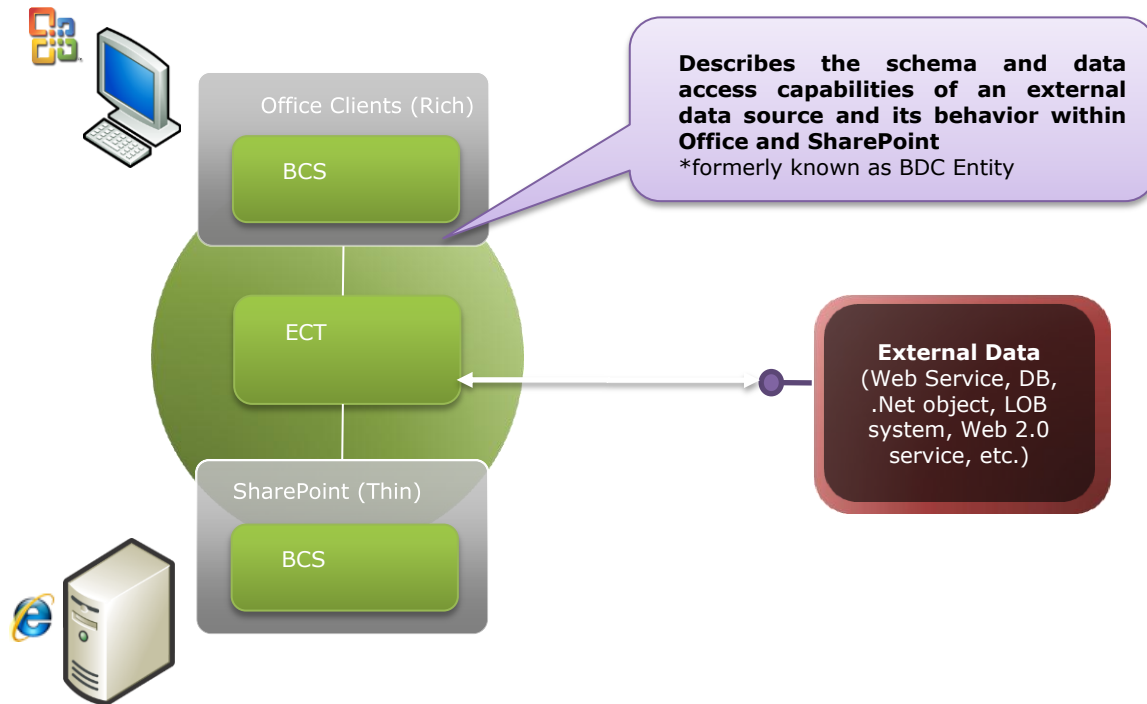


Figure 3

To let the BCS know how to interact with your LOB system, you define an External Content Type (ECT). An ECT defines the schema and operations to perform on the LOB data. There are two ways to create your ECT. If you create connections to SQL Server or Web Services the easiest way to create your definition is to make use of the free SharePoint Designer. If you need to create your own .Net assembly to interact with your LOB system, or would like to create a custom connector, you will have to use Visual Studio 2010.

2.2.5 LIMITATIONS

There are a few limitations you need to be aware of when you consider using BCS. First limitation is the amount of data you send over the wire in one request and second are the timeout values.

Type	Description	Scope	Default	Maximum
Connections	Total number of connections allowed to external systems	Global	100	500
Items	Number of rows returned from a database query	Database	2.000	25.000
Timeout	Database connection timeout	Database	60 sec	600 sec
Size	Size of returned data	WCF	3Mb	150Mb
Timeout	Web service connection timeout	WCF	60 sec	600 sec

(Rizzo, Alirezaei, Fried, Swider, Hillier, & Schaefer, 2010)

Another limitation considers caching. With the BCS Server instance you have no built in support for caching LOB data. The BCS designer in Visual Studio 2010 may trick you into believing there is, because you get to see the "Is Cached" property (see below).

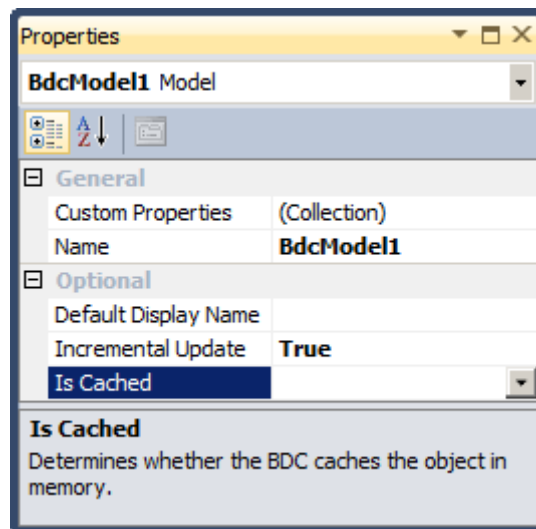


Figure 4

That property however defines whether to cache the Meta data model and not the results. There simply is no way of caching the results unless you program it yourself.

## 2.3 GUIDELINES

### 2.3.1 AUTHENTICATION

As mentioned in the overview, BCS comes with three different types of authentication to use. Each of these have a defined purpose and we'll explain them one by one, together with their considerations and some best practices.

#### 2.3.1.1 PASS THROUGH

Pass Through uses the credentials of the currently logged on user and might seem the ideal solution to comply with security guidelines. There are some caveats however. If you connect to SQL Server using different credentials for each connection, you lose the benefits of connection pooling. Creating new connections for all your users can get expensive in terms of resource usage. Another caveat is that you have to configure Kerberos on your servers to allow the credentials from your web front end to be passed on to your SQL server backend. NTLM does not allow the "double hop". There are not a lot of companies that take the hassle to configure Kerberos however. If you would like to use Pass Through authentication you cannot use Claims Authentication. As BCS does not use the Claims to Windows Token Service (C2WTS) it cannot convert your SharePoint claim to a Windows claim. You will notice that BCS will try to access the LOB system with the credentials of the anonymous user account.

#### 2.3.1.2 REVERT TO SELF

"Revert to Self" uses the credentials of the IIS application pool that is servicing the request. In external lists, this means the application pool for the content page; for timed workflows it means the process that runs the workflow; and so on. Because the application pool of a SharePoint farm is a very highly privileged account, it is not a recommended security practice to use "Revert to Self" in deployment environments. Luckily the "Revert to Self" option is disabled by default. You need to run a script from the SharePoint 2010 Management Shell to allow for "Revert to Self" authentication. But again, you should not use "Revert to Self".

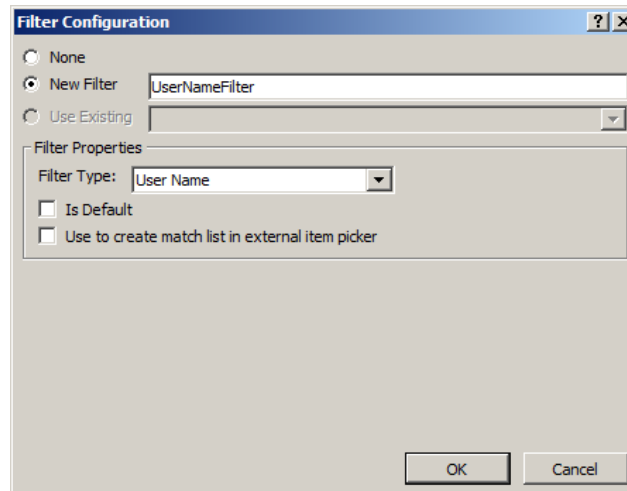
#### 2.3.1.3 SECURE STORE

Secure Store uses credentials that are stored for a user, group, or application inside an encrypted database. This essentially avoids the double hop problem of Pass Through. Since we can use the credentials in Secure Store to sign in the user again on the server, the double hop becomes two single hops. The reason to still choose for Pass Through - if possible - is that each time the user's Windows Credentials change, this user will be asked for new credentials by the Secure Store service for each application definition you created.

You can store credentials per user, per group, or per application (a so called application account) in the Secure Store application definition. Again, if you do decide to store the credentials per user, you lose the benefits of connection pooling. If you however decide to store the credentials per group, or per application you lose auditing possibilities because the backend system has no idea which specific user performs the action.

#### 2.3.1.4 FILTERS

One not very well known option to allow for auditing and authorization is the use of filters. You can use a built in BCS filter that adds the name of the logged in user as a parameter to your query.



**Figure 5**

Keep in mind though that these filters are only as secure as the connection to the external system and are no real security measure. It is up to the backend system to actually use the added parameter for authorization or auditing purposes.

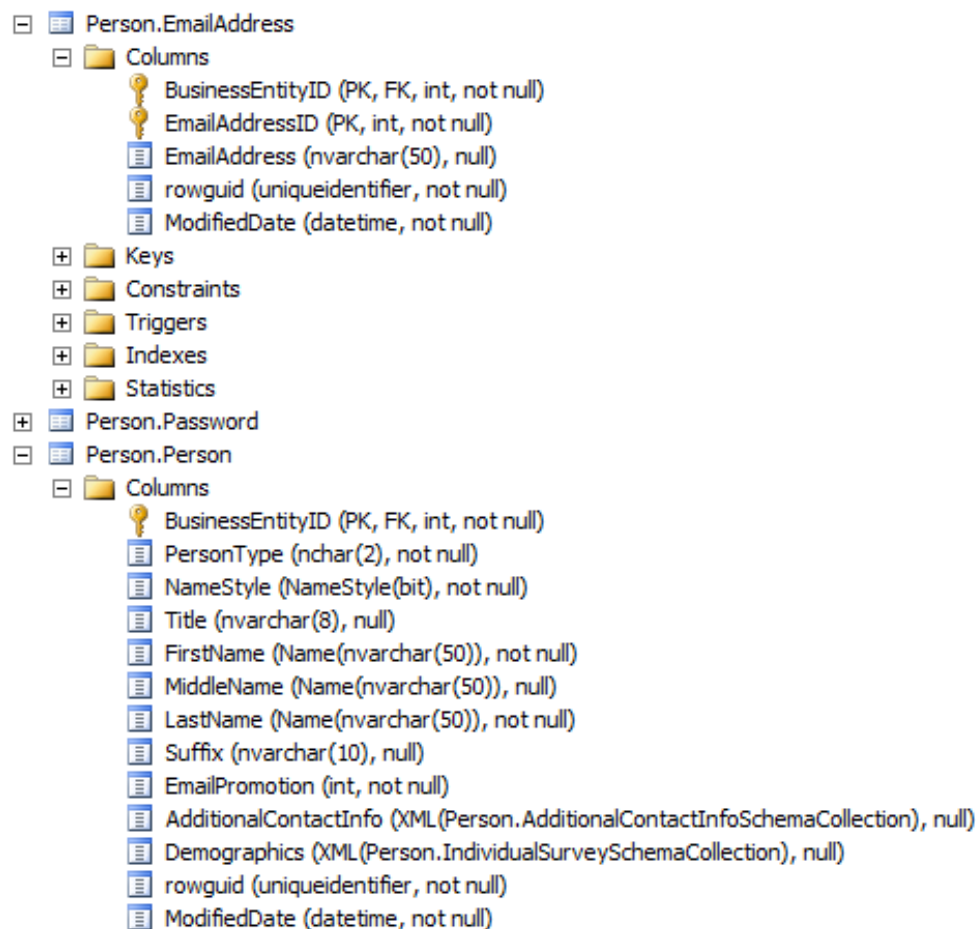
### 2.3.2 CONNECTIONS

There are four types of connections for External Content Types defined. All come with their pros and cons which we'll describe one by one.

#### 2.3.2.1 SQL SERVER

Creating an ECT that connects to SQL Server can be accomplished right from within SharePoint Designer. As awesome this might seem at first there are some limitations with this connector type. The SQL Server connector does not allow you to compose your ECT from multiple tables. This means that you'll always end up with database entities instead of domain entities.

If the database is normalized in any way, and most databases are, you'll probably end up with a number of completely useless ECTs. To work around this you could ask the owner of the database to create some database views that compose the database entities to domain entities. Views are almost never updateable though. You cannot, for example, create a simple "Person" ECT from within SharePoint Designer, which contains both the first email address and the first name from the AdventureWorks database, without making actual changes to the database.



**Figure 6 (AdventureWorks Database Schema)**

You'll also have to make a choice for authentication. If you decide to authenticate with the current user's identity, you can audit changes and can be sure that the user can perform only those actions, he or she is entitled to by the database administrator. The tradeoff is that you cannot benefit from connection pooling.

If you decide to use an application account from the Secure Store, you do benefit from connection pooling but you'll have to rely on BCS filters for auditing and have to be careful not to allow users to perform actions on the database that they normally would not be permitted to.

It might be best to create a read-only ECT that connects using an application account and a secondary read-write ECT that connects using the identity of the user. In that way you pick the best from both worlds. It does add an extra layer of maintenance however.

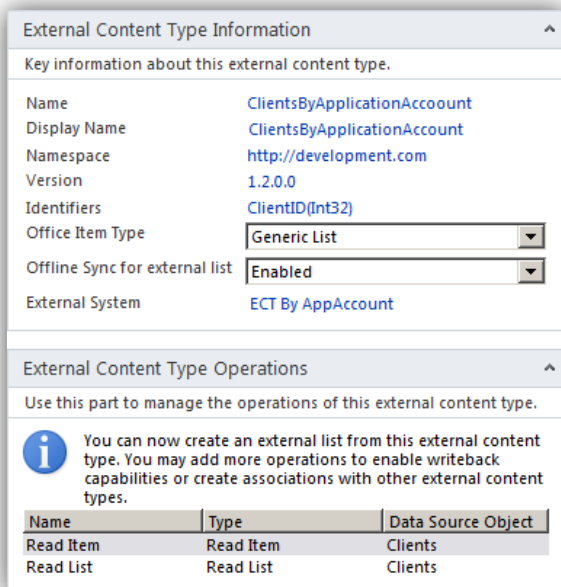


Figure 7 (read-only by app. account)

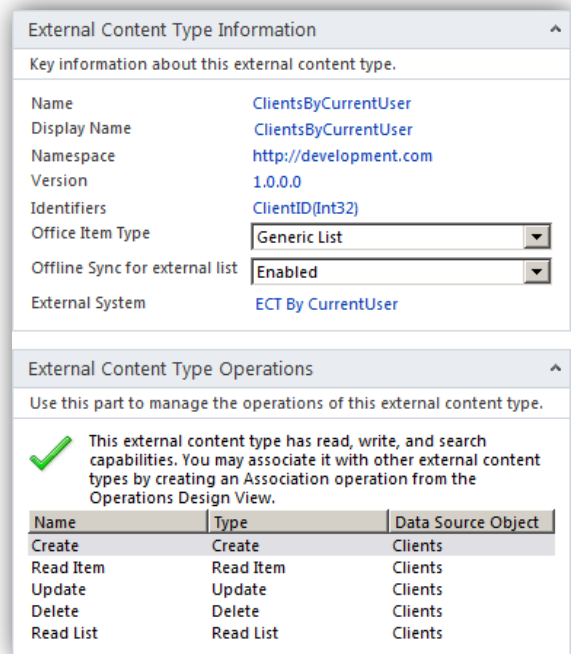
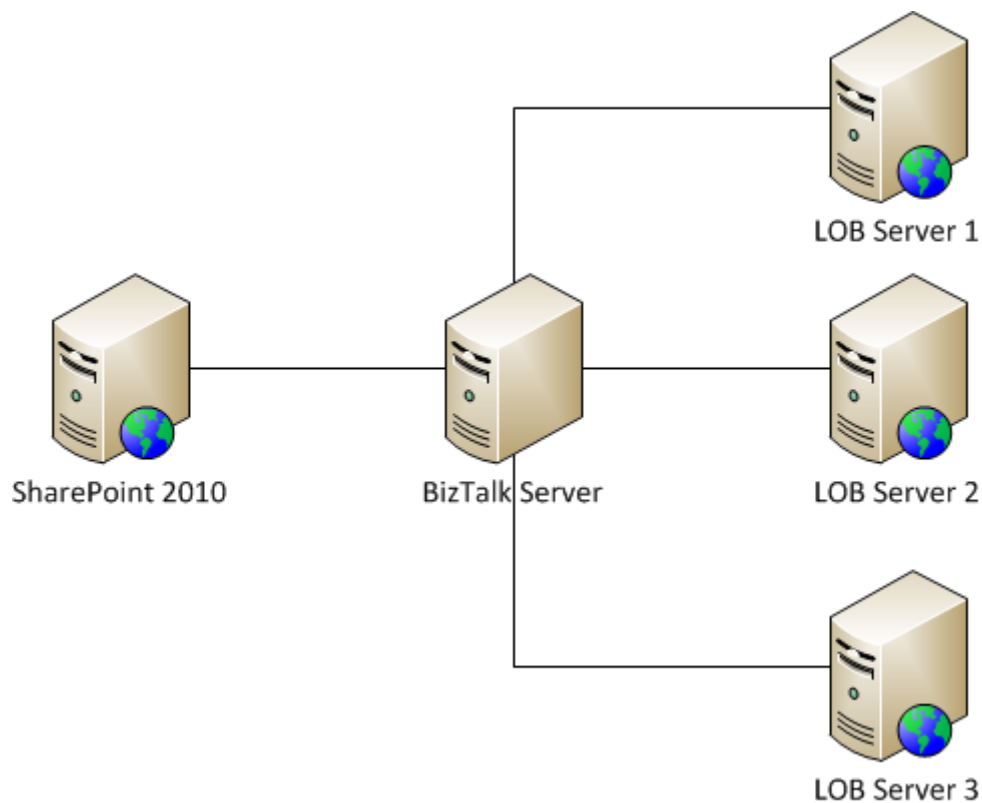


Figure 8 (read-write by user identity)

### 2.3.2.2 WCF

As with the SQL connector, you can also easily add an ECT that uses a WCF connector through SharePoint Designer. You'll have to take in account though that, as with the SQL connector, you cannot use composition. That means that every ECT maps to a single data contract exposed by your WCF service. WCF services however are expected to expose domain entities so this should not be a real issue. The caveat is in the fact that a lot of services however expose **complex** domain entity types, which can be used as ECTs but show just the first layer of the flattened hierarchy in the default SharePoint view- and edit forms.

Although BCS can handle these non flat types in various ways with some changes to the property and type descriptors (Mechelke, 2010), your best option is to create a service layer (see chapter 5) that flattens the hierarchy and composes the ECT out of multiple orchestrated requests to (different) LOB systems if necessary as in below figure.

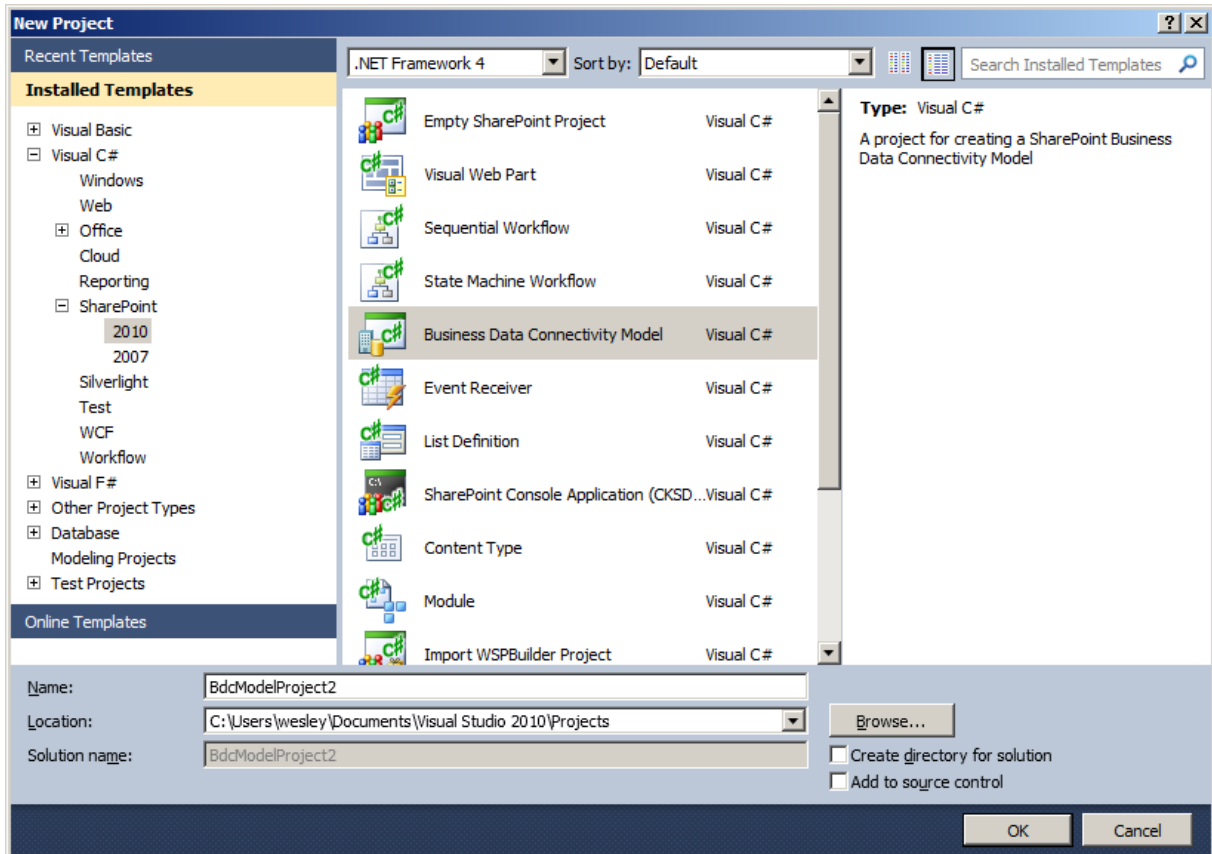


**Figure 9 (Example architecture using BizTalk Server as a service layer)**

One other good option is to use InfoPath Forms Services for complex ECT. This also doesn't come for free however.

### 2.3.2.3 .NET ASSEMBLY

A completely different option is to create an ECT with the Visual Studio 2010 Business Data Connectivity Model project template.



**Figuur 10 (Creating an ECT)**

This project template aids a developer in creating a .Net Assembly that returns data from any LOB system for which a .Net provider is available. You can also use this assembly to perform request composition or format your data. Remember that it is not a trivial task to orchestrate asynchronous requests and again a service layer can be of great value in this case.

In essence you are completely free to design the classes that will be returned to SharePoint. If you want to be able to easily create External Lists for your ECT, your classes must be “flat” and contain only simple typed properties.

Although this all seems very flexible you have to be aware that the definitions of your classes are rigid. You’ll need a developer again as soon as there’s an extra property you would like to add to your ECT. If your classes, and the back-end systems they come from, are not likely to change however, you can build a good ECT solution using a .Net Assembly connector. Adding a service layer could help you with request composition, throttling and monitoring the data traffic. A service layer also adds flexibility, scalability and configuration to your solution (see chapter 5 for more detail). Keep in mind that your custom assembly does run on your front-end web server so it should be light weight.

2.3.2.4 CUSTOM CONNECTOR

Using a Custom Connector for your ECT is both the most difficult and the most flexible solution. A Custom Connector can be seen as an ECT definition interpreter. Below figure explains the workflow of a Custom Connector from the point of view of the Custom Connector.

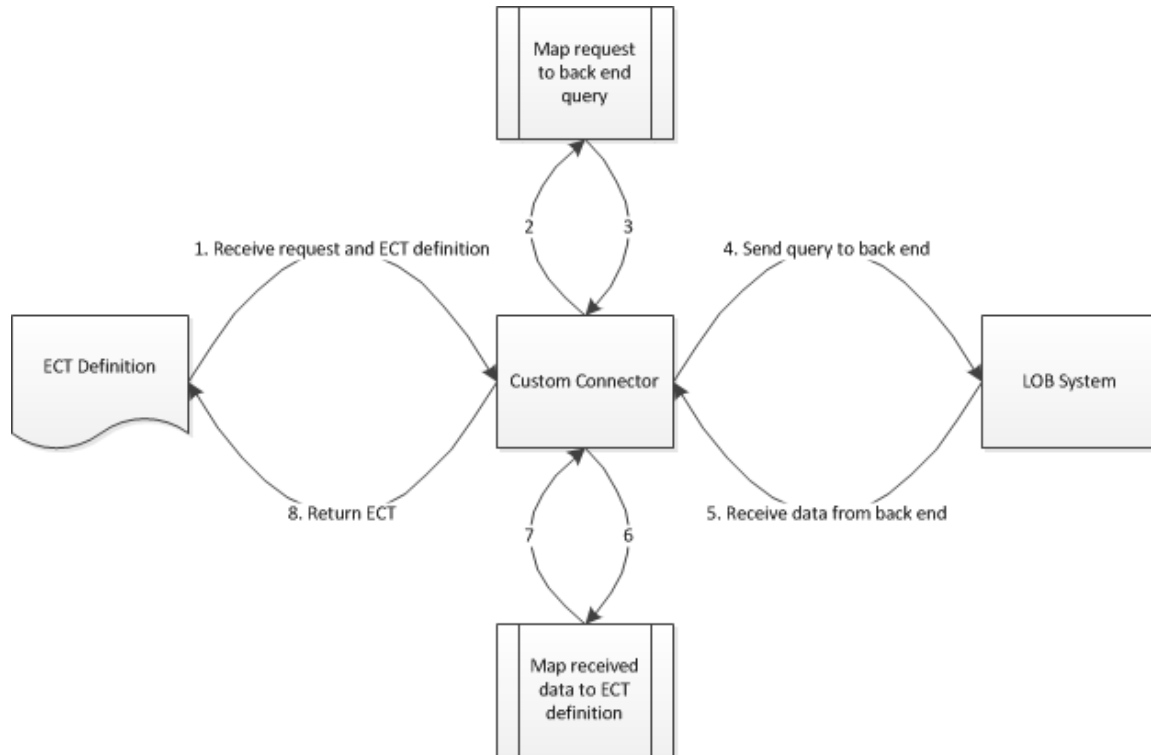


Figure 11

A good candidate for a Custom Connector would be a Microsoft Dynamics CRM (CRM) connector. Each and every CRM implementation is different in the defined entities and the properties that are defined for those entities. This makes it impossible to create a single .Net Assembly connector to accommodate the retrieval of data from CRM. You would have to create a custom .Net Assembly for each of your customers and your customers would have to rely on you to deliver a new version as soon as they add an extra property to one of their CRM entities.

If we create a Custom Connector instead of a .Net Assembly, all that needs to be customized for each customer is the ECT definition. The Custom Connector will use this definition to send the back-end specific queries to the Microsoft Dynamics CRM web services and translate the results to the defined ECT.

Building a truly generic Custom Connector is very difficult. You have to take in account that, although SharePoint external lists need simple types, BCS allows for complex type definitions. You are basically parsing, translating and mapping an XML definition and XML, as we all know, is very flexible. Custom Connectors are probably not something you would like to build for one time use only, and are far more interesting for Microsoft partners that are specialized in system integration. The next page displays a sample implementation.

```

<LobSystem Name="CrmLOB" Type="Custom">
  <Properties>
    <Property Name="SystemUtilityTypeName" Type="System.String">motion10.Bcs.Crm.CrmConnector,
      motion10.Bcs.Crm, Version=1.0.0.0,
      Culture=neutral,
      PublicKeyToken=13be468f99f8d374</Property>
    <Property Name="SystemUtilityInstallDate" Type="System.DateTime">2010-12-25 00:00:00Z</Property>
  </Properties>
  <LobSystemInstances>
    <LobSystemInstance Name="CrmConnector">
      <Properties>
        <Property Name="CrmServer" Type="System.String">http://sp2010:5555</Property>
        <Property Name="CrmOrganization" Type="System.String">Development</Property>
      </Properties>
    </LobSystemInstance>
  </LobSystemInstances>
  <Entities>
    <Entity EstimatedInstanceCount="1000" Name="Account" Namespace="Crm" Version="1.0.0.0">
      <Properties>...</Properties>
      <Identifiers>...</Identifiers>
      <Methods>
        <Method Name="GetAccount">
          <Parameters>
            <Parameter Direction="In" Name="accountId">
              <TypeDescriptor TypeName="System.Guid" Name="accountId" LobName="accountid" IdentifierName="AccountId"/>
            </Parameter>
            <Parameter Direction="Return" Name="Account">
              <TypeDescriptor TypeName="Microsoft.BusinessData.Runtime.DynamicType" Name="Account" LobName="account">
                <TypeDescriptors>
                  <TypeDescriptor DefaultDisplayName="ID" Name="accountid" TypeName="System.Guid" IdentifierName="AccountID"/>
                  <TypeDescriptor DefaultDisplayName="Name" Name="name" TypeName="System.String" ReadOnly="true"/>
                  <TypeDescriptor DefaultDisplayName="Email" Name="emailaddress1" TypeName="System.String" ReadOnly="true" />
                  <TypeDescriptor DefaultDisplayName="# of Employees" Name="numberofemployees" TypeName="System.Int32" ReadOnly="true" />
                  <TypeDescriptor DefaultDisplayName="Created On" Name="createdon" TypeName="System.DateTime" ReadOnly="true" />
                <Interpretation>
                  <NormalizeDateTime LobDateTimeMode="UTC" />
                </Interpretation>
              </TypeDescriptor>
            </Parameter>
          </Parameters>
        </Method>
      </Methods>
    </Entity>
  </Entities>
</LobSystem>

```

Figure 12 (Custom Connector ECT Definition)

```

public void ExecuteStatic(IMethodInstance methodInstance, ILobSystemInstance lobSystemInstance,
    object[] methodSignatureArgs, IExecutionContext executionContext) {
    parameter validation

    string serverName = lobSystemInstance.GetCrmServerName();
    string organizationName = lobSystemInstance.GetCrmOrganizationName();

    var entityType = methodInstance.GetCrmEntityTypeName();
    var lobAttributes = methodInstance.GetLobAttributes().ToArray();
    var propertyMapping = methodInstance.GetPropertyMappings();

    int lastArgument = methodSignatureArgs.Length - 1;

    switch (methodInstance.MethodInstanceType) {
        case MethodInstanceType.Finder:
        case MethodInstanceType.IdEnumerator:
            using (CrmService service = CrmService.GetCrmService(serverName, organizationName)) {
                methodSignatureArgs[lastArgument] = service.GetDynamicEntities(entityType, lobAttributes)
                    .ToDynamicTypes(propertyMapping)
                    .ToArray();
            }
    }
}

```

Figure 13 (Custom Connector Assembly)

2.4 CONCLUSION

Business Connectivity Services offer great value for money as it comes essentially free with your server’s operating system! It is also very flexible in the ways it offers to connect and authenticate to your LOB systems. The problem with BCS lies in the fact that in most cases, without adding some sort of a service layer, your External Content Types cannot be used for SharePoint external lists.

For the creation of these service layers you have multiple options. One not so familiar option is the fact that SharePoint 2010 has built in support for creating managed service applications. These service applications can be installed on your application servers. By doing so, your applications integrate fluently with the SharePoint 2010 platform and thus become scalable and reliable. The SharePoint 2010 Topology Service takes care of the load balancing of your service application.

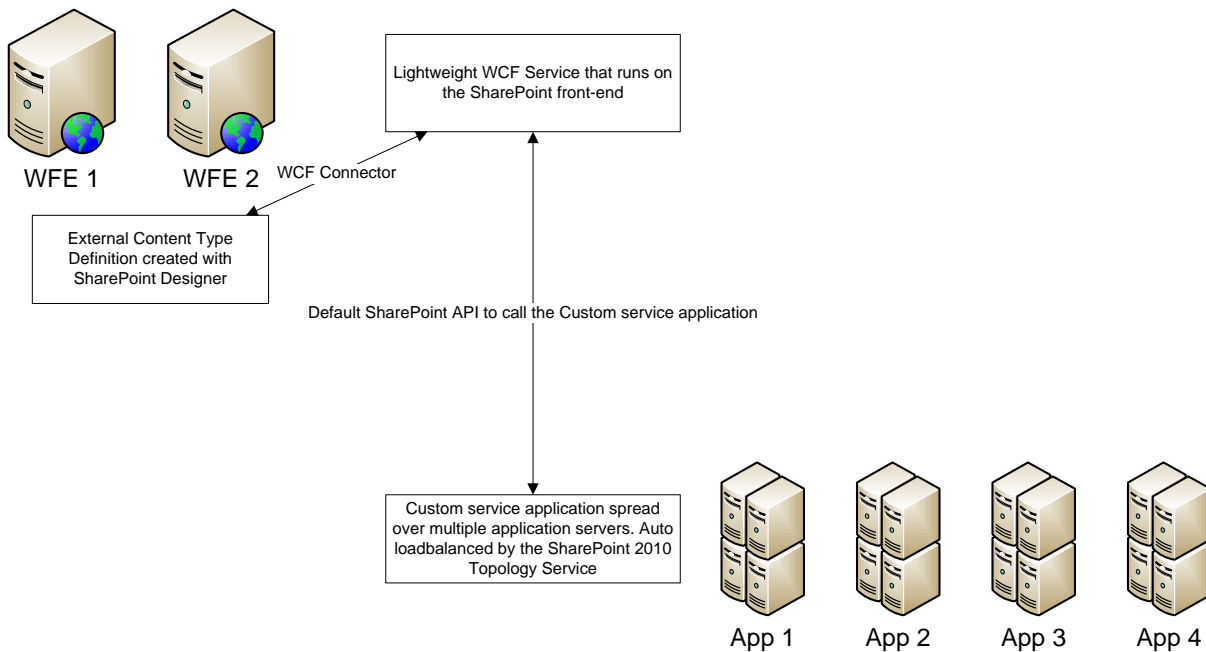


Figure 10

Although this sounds tempting, it is not all that easy because of the lack of good documentation or code samples (also see 5.3.1).

### 3 SEARCH

#### 3.1 INTRODUCTION

Integration of LOB applications in SharePoint starts with the availability of the business entities. If your LOB system contains lots of entities the discovery of these entities becomes more and more important. This chapter describes how to use search in your integration projects.

#### 3.2 OVERVIEW

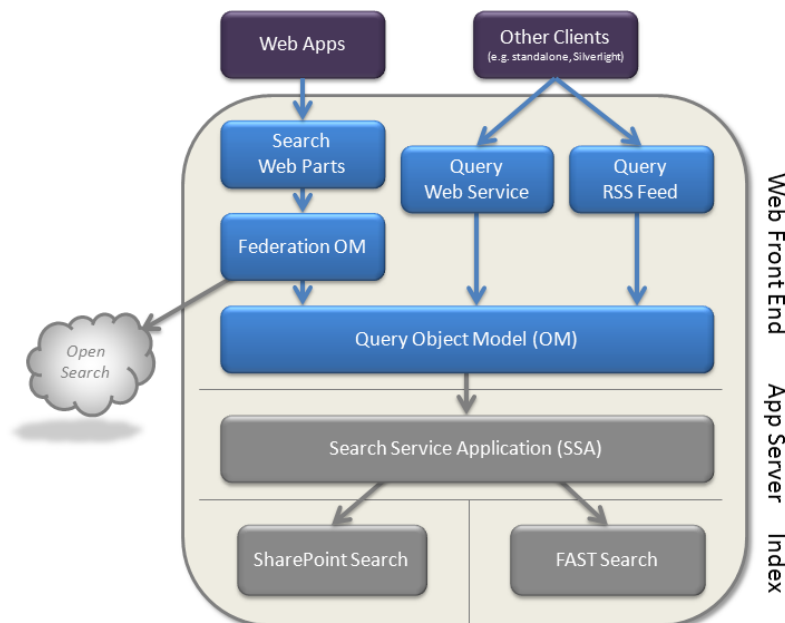
One of the very powerful features of SharePoint is Search. Search comes in different tastes, but for this white paper we'll stick to the SharePoint Server Search.

Feature	SharePoint Foundation 2010	SharePoint Server 2010	FAST Search Server 2010
Basic search	Yes	Yes	Yes
Document preview	No	No	Yes
Indexing sites	Yes	Yes	Yes
Indexing external content	No	Yes	Yes
People search	No	Yes	Yes
Query federation	No	Yes	Yes
UI-based administration	Limited	Yes	Yes
Visual best bets	No	Limited	Yes

**Figure 15 (Krause, Langhirt, Sterff, Pehlke, & Döring, 2011)**

As the above table clearly shows, SharePoint Server Search offers a lot of features. It comes included with the product and offers easy configuration.

### 3.2.1 QUERY ARCHITECTURE



**Figure 11**

Figure 11 gives a nice overview of the SharePoint Server Search query architecture. The important things to look at in terms of integration are the Federation OM and Open Search. Although you can index your BCS External Content Types, you will always have to deal with the fact that you have no real time search results when you do so. We'll get back to that later.

The Federation OM is a new object model in SharePoint 2010 that provides a unified interface to search against different locations that comply with the Open Search specification. The Open Search standard, as its name suggests, is open, and many search providers comply with this standard.

In the next chapter you will find explanations and guidelines regarding search.

### 3.3 GUIDELINES

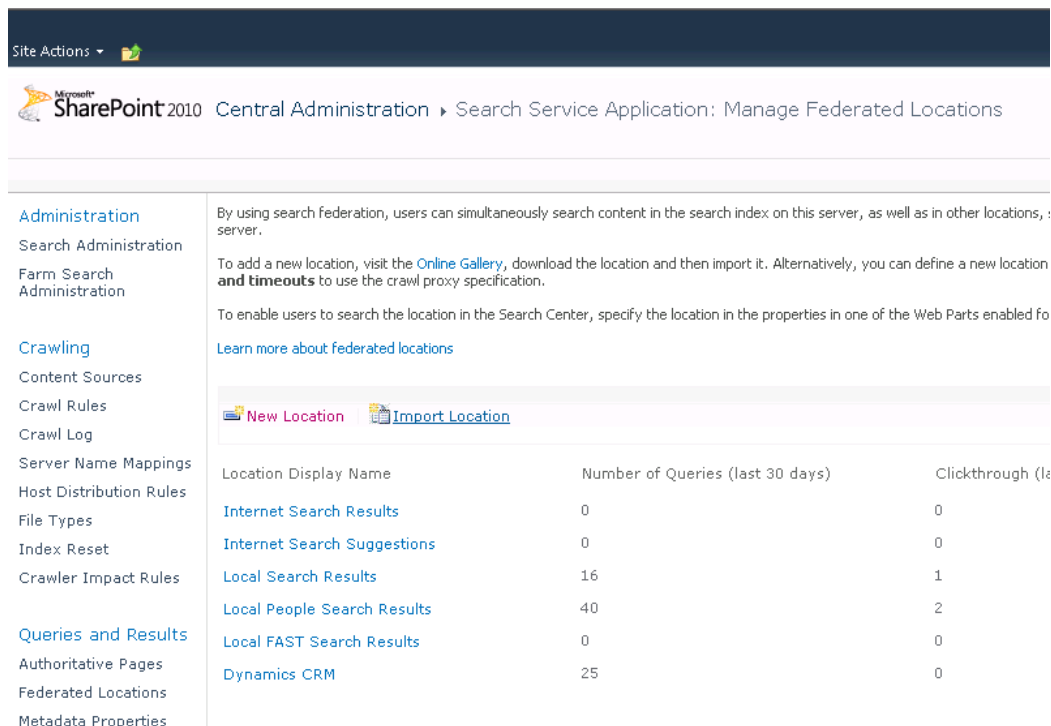
#### 3.3.1 FEDERATED SEARCH

The Open Search specification defines both a Search Services Description file and a Search Result Definition. Below you'll find an example of a simple description file.

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
  <ShortName>CRM Search</ShortName>
  <Description>Use OData to search Dynamics CRM.</Description>
  <Tags>OData CRM</Tags>
  <Contact>services@development.com</Contact>
  <Url type="application/atom+xml"
    template="http://example.com/?q={searchTerms}"/>
</OpenSearchDescription>
```

The result definition basically states that as long as the query URL returns an RSS or Atom feed you are good to go. So how does this work in SharePoint 2010?

With SharePoint 2010 you've got the option to import an Open Search Description file or to manually fill in the required elements for a federated search location. You can do this inside SharePoint Central Administration.



**Figure 12**

If you decide to add the federated location manually you just have to define a name, a search URL template and an XSLT that defines how the results should be transformed, together with the credentials that should be used to execute the query.

**Specify Credentials**

Specify the access credentials for this location.

Select **Anonymous** when the location does not require authentication.

Select an authentication protocol under **Common** when the location requires authentication, and your company uses a single account that all end-users use to authenticate against the location.

Select an authentication protocol under **User** when the location requires authentication and each user has a unique account to authenticate against the location.

Anonymous: This location does not require authentication

Common:

- Basic Authentication - Specify a user name and password
- Digest Authentication - Specify a user name and password
- NTLM - Use Application Pool Identity
- NTLM - Specify a username and password
- Form Authentication - Specify form credentials
- Cookie Authentication - Use cookie for authentication

User:

- Kerberos - User credentials passed automatically
- Basic Authentication - User provides user name and password
- Digest Authentication - User provides user name and password
- NTLM - User provides user name and password
- Form Authentication - User provides form credentials
- Cookie Authentication - User provides cookie for authentication

**Figure 13**

Especially the fact that you can use Kerberos to forward the credentials of the current user to the search service can come in very handy as you will see in the following real life example.

### 3.3.1.1 REAL LIFE FEDERATED SEARCH EXAMPLE

There is a really good reason to get very happy about the fact that you can add federated search locations. A lot of applications nowadays come with an Open Data Protocol (OData) web service to query the application for business entities. Most of these OData web services offer their results in a format to your liking, such as RSS, Atom and JSON.

If we combine both the open search protocol, that allows us to define a URL template for querying, and the OData protocol that returns an RSS or Atom feed, we can search any application that exposes an OData web service, without writing a single line of code.

As an example we'll use Microsoft Dynamics CRM 2011 (CRM) that exposes an OData services at the following URL:

```
http://{domain}/{organisation}/XRMServices/2011/OrganizationData.svc
```

By adjusting the URL we can query for specific data, which will be returned in the ATOM format. A complete reference on the supported query specification can be found on [MSDN](#). For this example we'll query for some basic account information for each account where the name begins with a specific search term:

```
{ServiceUrl}/AccountSet?
$select=AccountId,Name,AccountNumber,Address1_Line1,Address1_Line2,Address1_Line2,
      Address1_City,Address1_Country
&&$filter=startswith(Name, '{searchTerms}')
&&$top={count}
```

You can simple try this out by pasting the URL in your browser and replacing the {searchTerms} placeholder with a character and the {count} placeholder with an integer. Figure 14 shows a sample result.

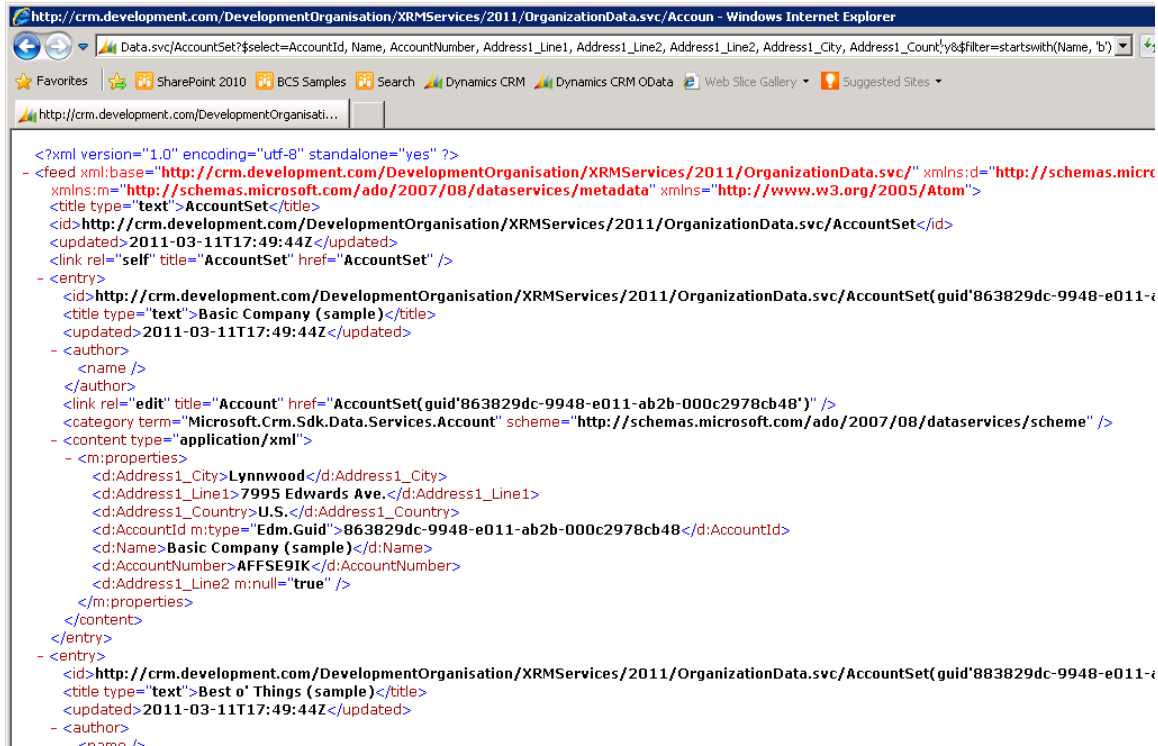


Figure 14

We can paste this now tested URL into the URL template field while we define a new Federated Search location inside Central Administration. Please notice the {searchTerms} and {count} placeholders in the Text Editor window in Figure 15. SharePoint Search web parts will replace these placeholders with the text that is entered into the search box and the item limit property of the Core Results web part.

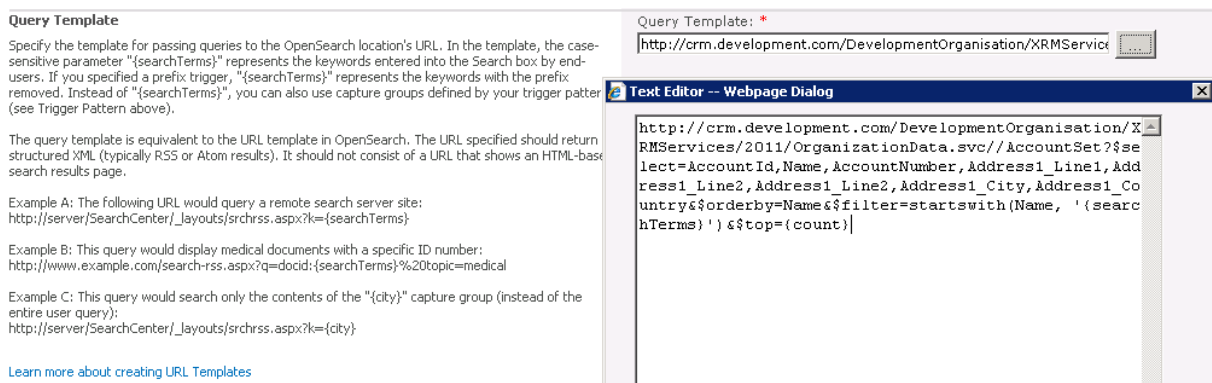


Figure 15

After editing the XSLT templates with SharePoint Designer and making sure we use Kerberos to send the current users' client credentials to the CRM service, we can now securely query CRM from within SharePoint.

It all results in security trimmed and completely up to date data from our LOB system right inside SharePoint. Who said search integration is hard?

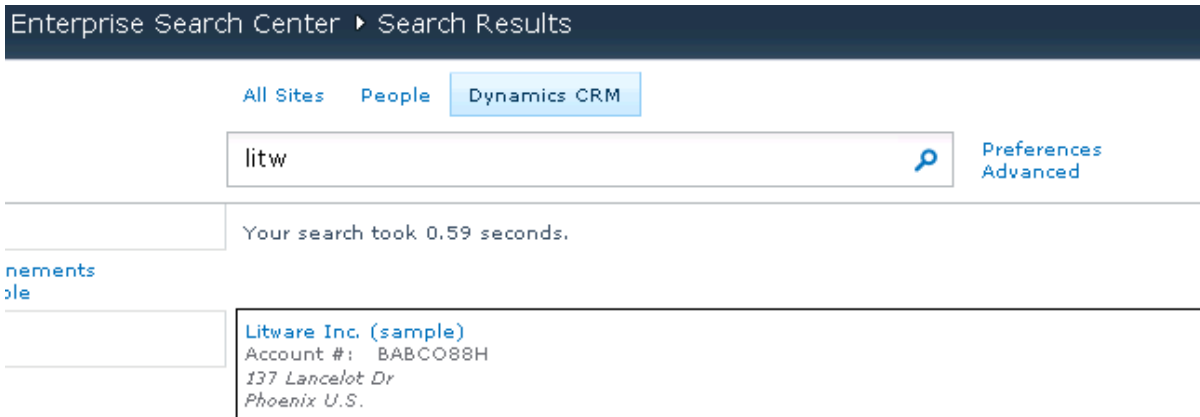


Figure 16

We do have to keep in mind though that our federated search is as strong as the federated search service we use. The CRM OData service for example does not allow you to search inside attachments. It is also very unfortunate that the OData protocol defines a "\$skip" parameter that can be combined with the "\$top" parameter to get paging, while the OpenSearch specification defines a "startIndex" placeholder and a "count" placeholder. Although the "count" placeholder maps one to one with the "\$top" parameter, the "startIndex" placeholder is 1 based while the "\$skip" parameter is zero based. This means that if we map the "startIndex" placeholder to the "\$skip" parameter, ...&\$skip={startIndex}..., we will always lose the first search result. So we cannot configure paged search results. Do not forget to insert the "\$top" parameter to limit the returned results though! Search also does not use the C2WTS so it cannot pass your Windows credentials when you use claims authentication. We either use the Secure Store to pass user credentials (Figure 17) or we take the "Man-in-the-middle" approach displayed in Figure 18 and Figure 19, where the "Man in the middle" could be replaced by a simple application page or even better an HttpHandler

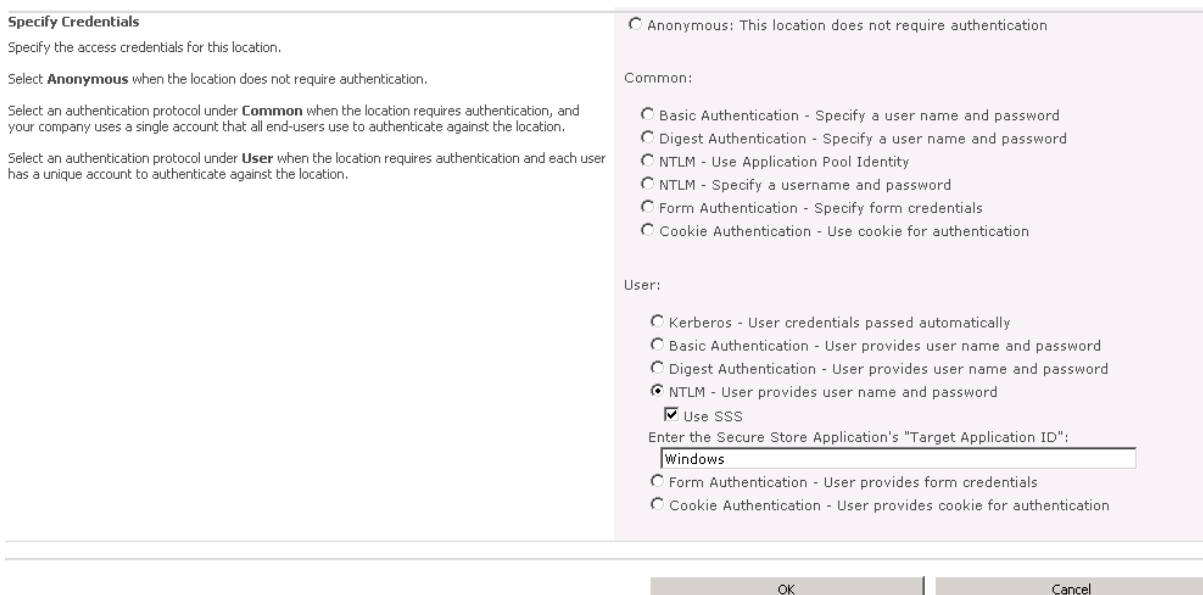


Figure 17

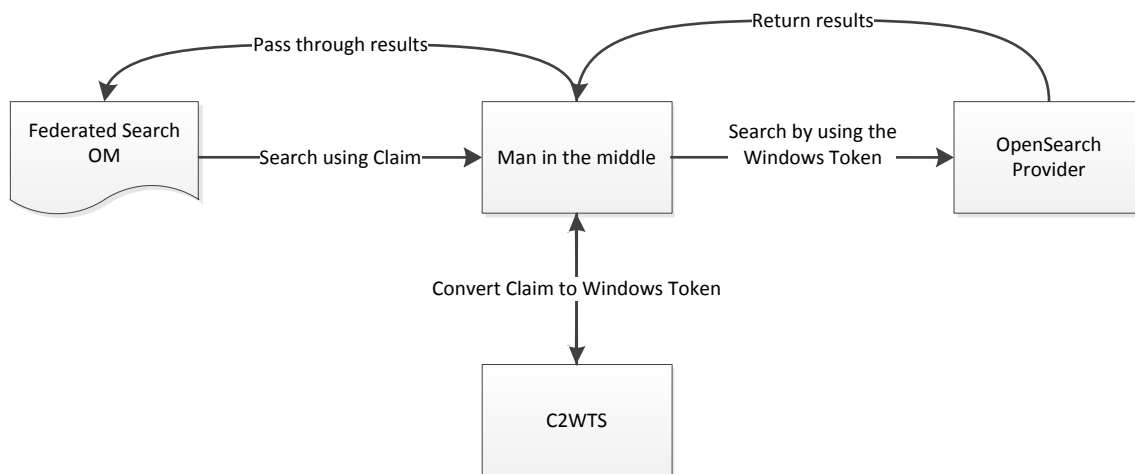


Figure 18 (Man in the middle approach)

```

IClaimsIdentity identity = (ClaimsIdentity)Thread.CurrentPrincipal.Identity;
string upn = null;
foreach (Claim claim in identity.Claims) {
    if (StringComparer.Ordinal.Equals(ClaimTypes.Upn, claim.ClaimType)) {
        upn = claim.Value;
    }
}

// Perform the UPN logon through the c2WTS.
WindowsIdentity windowsIdentity = null;
if (!String.IsNullOrEmpty(upn)) {
    try {
        windowsIdentity = S4UClient.UpnLogon(upn);
    } catch (SecurityAccessDeniedException) {
        Console.WriteLine("Could not map the upn claim to a valid windows identity.");
        return;
    }
} else {
    throw new Exception("No UPN claim found");
}

string result = null;

using (WindowsImpersonationContext ctxt = windowsIdentity.Impersonate()) {
    WebClient client = new WebClient();
    client.UseDefaultCredentials = true;

    result = client.DownloadString(forwardUrl);
}
  
```

Figure 19 (Using C2WTS to call a web service with the Windows Identity of the currently logged on Claims Authenticated User)

### 3.3.2 SHAREPOINT SERVER SEARCH

SharePoint Server Search indexing works by using so called “protocol handlers”. A protocol handler enables the indexing of specific types of information sources. A BCS protocol handler allows us to index content that is made available through BCS, and a File System protocol handler allows us to index file shares for example. Another very important protocol handler is the Http protocol handler that allows us to query web sites.

There are a few more protocol handlers, but in this white paper we will limit ourselves to BCS which is most commonly used in integration scenarios.

#### 3.3.2.1 BCS

We’ve discussed BCS already but never mentioned search so far. To use BCS in combination with search we will need to make some adjustments to our BCS model. There are also a number of things to keep in mind when we start to crawl our external content type (ECT).

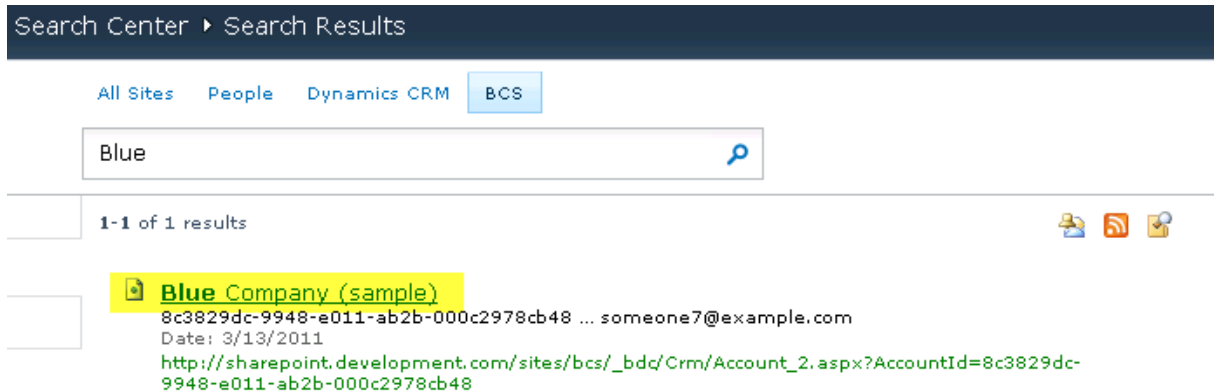
##### 3.3.2.1.1 VISIBILITY

First of all we need to add the “ShowInSearchUI” property to the LobSystemInstance element. We also need to specify either a “RootFinder” property on one of the Finder method instances or define an “IdEnumerator” method instance. If we don’t, our External Data Source will not show up in the search UI at all. Figure 20 highlights these attributes.

```
<LobSystemInstances>
  <LobSystemInstance Name="CrmConnector">
    <Properties>
      <Property Name="ShowInSearchUI" Type="System.Boolean">true</Property>
      <Property Name="CrmServer" Type="System.String">http://crm.development.com</Property>
      <Property Name="CrmOrganization" Type="System.String">DevelopmentOrganisation</Property>
    </Properties>
  </LobSystemInstance>
</LobSystemInstances>
<Entities>
  <Entity EstimatedInstanceCount="1000" Name="Account" Namespace="Crm" Version="1.0.0.0">
    <Properties>
      <Property Name="__BdcLastModifiedTimestamp" Type="System.String">modifiedon</Property>
      <Property Name="Title" Type="System.String">name</Property>
    </Properties>
    <Identifiers>...</Identifiers>
    <Methods>
      <Method Name="GetAccount">...</Method>
      <Method Name="GetAccounts">...</Method>
      <Method Name="GetAccountIds">
        <Parameters>...</Parameters>
        <MethodInstances>
          <MethodInstance Type="IdEnumerator"
            ReturnParameterName="AccountIds"
            ReturnPropertyDescriptorPath="AccountIds"
            Name="AccountsIdEnumerator">
          </MethodInstance>
        </MethodInstances>
      </Method>
    </Methods>
  </Entity>
</Entities>
```

**Figure 20**

Also note the “\_\_BdcLastModifiedTimestamp” and the “Title” properties on our Entity in Figure 20. These attributes ensure that the search engine can perform incremental updates and that it knows which field to display as title in our search results (Figure 21).



**Figure 21**

The last thing we need to do for search to actually work is to make sure that the profile pages are created. There are some common things to keep in mind though.

#### 3.3.2.1.2 INCREMENTAL SEARCH

Although the `IdEnumerator` speeds up the incremental crawl if we define the `__BdcLastModifiedTimestamp` property, it still needs to retrieve all entity id's and their last modified date from the backend. Search is not subjected to query throttling so indeed all records will be retrieved and processed. Although this is far better than a full crawl, it does result in a lot of data traveling over the wire if we have millions of records. Luckily there is a better solution.

SharePoint 2010 BCS comes along with filters. We can use these filters to store the last run date in a so called synchronization cookie.

```
<FilterDescriptors>
  <FilterDescriptor Name="LastRunDate" Type="InputOutput">
    <Properties>
      <Property Name="SynchronizationCookie"
        Type="System.String">ChangedItemCookie</Property>
    </Properties>
  </FilterDescriptor>
</FilterDescriptors>
<Parameters>
  <Parameter Name="@LastRunDate" Direction="InOut">
    <TypeDescriptor Name="LastRunDateTypeDescriptor" TypeName="System.DateTime"
      AssociatedFilter="LastRunDate">
      <Interpretation>
        <NormalizeDateTime LobDateTimeMode="Local" />
      </Interpretation>
    </TypeDescriptor>
  </Parameter>
</Parameters>
```

You can then use `@LastRunDate` parameter to retrieve the records that have been changed since the last crawl. In that way SharePoint needs to maintain just one value (the last run date) and only those records that have actually changed will travel over the

wire. This results in a much more efficient incremental update. Eric White (Microsoft) wrote a must read article on the subject (White, 2010).

### 3.3.2.1.3 SECURITY

Crawling takes place with the credentials of the configured content access account. This causes our search index to contain all records that the content access account is allowed to index. Without taking measures the query engine returns those records without bothering about permissions of the user that is firing the search request. So despite our security measures taken while we defined the External Content Type (page 8) we now have a huge gap in our security system! SharePoint comes with two possible solutions to fill the gap.

If the backend accounts can be mapped one-to-one with Windows accounts on the SharePoint frontend (for example if both applications run in the same domain and both use windows authentication), we can add a SecurityDescriptor field to our ECT type descriptor. We can then add the WindowsSecurityDescriptorField property that references the field to the Finder and SpecificFinder method instances. This ensures that search will return only those records that the user performing the search is allowed to see.

```
<TypeDescriptor TypeName="System.Byte[]" IsCollection="true"
                Name="SecurityDescriptor">
  <TypeDescriptors>
    <TypeDescriptor TypeName="System.Byte" Name="SecurityDescriptorElement" />
  </TypeDescriptors>
</TypeDescriptor>
...
<MethodInstance Type="Finder"
                ReturnParameterName="Accounts"
                ReturnPropertyDescriptorPath="Accounts"
                Name="AccountsFinder">
  <Properties>
    <Property Name="WindowsSecurityDescriptorField"
              Type="System.String">SecurityDescriptor</Property>
  </Properties>
</MethodInstance>
```

If the backend accounts cannot be mapped 1:1 to Windows accounts on the frontend we will have to define our own security trimmer. This can easily be done by creating a new class that implements the ISecurityTrimmer2 interface. This interface contains just two members: Initialize and CheckAccess.

In short your security trimmer receives a list with all search results and the identity of the user that is performing the search. Your security trimmer then needs to return a bit array, one bit for each URI, that defines if the user is allowed to view the URI (bit says 1) or not (bit says 0). Your security trimmer can be configured using the Windows PowerShell cmdlet SPEnterpriseSearchSecurityTrimmer. (Microsoft, 2010)

The easiest way to implement a BCS security trimmer is by defining an AccessChecker method instance on your ECT.

You can then use the built-in `SPBdcSecurityTrimmer` that uses this method instance to check access before returning the results.

```
$searchapp = Get-SPEnterpriseSearchServiceApplication
new-spenterprisesearchsecuritytrimmer -SearchApplication $searchapp -TypeName
  "Microsoft.Office.Server.Search.Connector.BDC.SPBC.SPbdcSecurityTrimmer,
  Microsoft.Office.Server.Search.Connector, Version=14.0.0.0, Culture=neutral,
  PublicKeyToken=71e9bce111e9429c" -RulePath bdc3://* -Id 1
```

### 3.4 CONCLUSION

SharePoint Server Search offers great opportunities to open up a window to your backend applications.

With built in security trimming your end users will only see results for items they are allowed to access. The BCS search connector allows you to index any LOB system for which you've already created the ECT. You just have to be careful on how much data you are indexing on each index run.

Federated Search offers real time, security trimmed search results. One other thing to keep in mind is that federated search puts the processing of the search requests at the servers that are closest to the data and at the applications that have the most knowledge of the data structure. This could potentially lead to much faster/better query processing.

## 4 RECORDS MANAGEMENT

### 4.1 INTRODUCTION

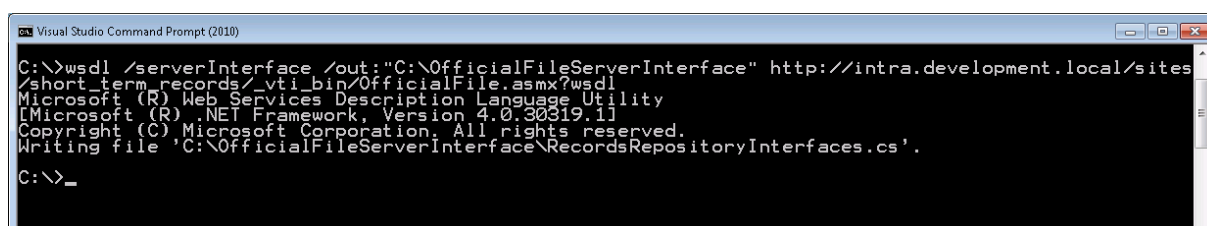
So far we have discussed possibilities that all deliver a result from a LOB system to SharePoint. What we tend to forget is that there are very good reasons for data traffic in the opposite direction. We could use all sorts of custom coded web parts, event receivers and custom actions to send data from SharePoint to another LOB system, but there is built in support for "Send To" connections in SharePoint 2010. This functionality is originally designed to send items from any SharePoint site to a SharePoint Records Center site, but we can leverage this functionality to send our data to anywhere we need. In this chapter we will demonstrate one of the easiest ways to send data from SharePoint to another LOB system. Or in other words: One of the easiest ways to integrate your custom Enterprise Records Management system into SharePoint.

### 4.2 OVERVIEW

SharePoint Document Libraries in Team Sites are usually not intended for long term storage. Document Libraries are there to store the items that are active. Once you are finished working actively on an item, most of the times you should send it to some sort of records repository. SharePoint 2010 does come with a much better, much more scalable records repository solution. You can now create hierarchical data storage by routing your document from one Records Center site to the other, based on retention policies and/or content organizer rules.

By configuring Send To connections in the Central Administration site you can add multiple Send To locations for your web applications. You configure them by entering the display name, action and a URL to the **.asmx** web service (Figure 23). The web service must adhere to the Official File Web Service Protocol Specification (Microsoft). The Records Center site template contains an officialfile.asmx web service that does this.

Implementing your own custom records repository is as simple as creating a web service that implements the specified interface. The easiest way to create such a web service is by letting the **WSDL** tool create the interface definition for you by using the **/serverInterface** option.



```
Visual Studio Command Prompt (2010)
C:\>wsdl /serverInterface /out:"C:\OfficialFileServerInterface" http://intra.development.local/sites
/short_term_records/vti_bin/OfficialFile.asmx?wsdl
Microsoft (R) Web Services Description Language Utility
[Microsoft (R) .NET Framework, Version 4.0.30319.11
Copyright (C) Microsoft Corporation. All rights reserved.
Writing file 'C:\OfficialFileServerInterface\RecordsRepositoryInterfaces.cs'.
C:\>_
```

Figure 22 (Create the interface using the WSDL tool)

The created interface can then be implemented by your custom web service.

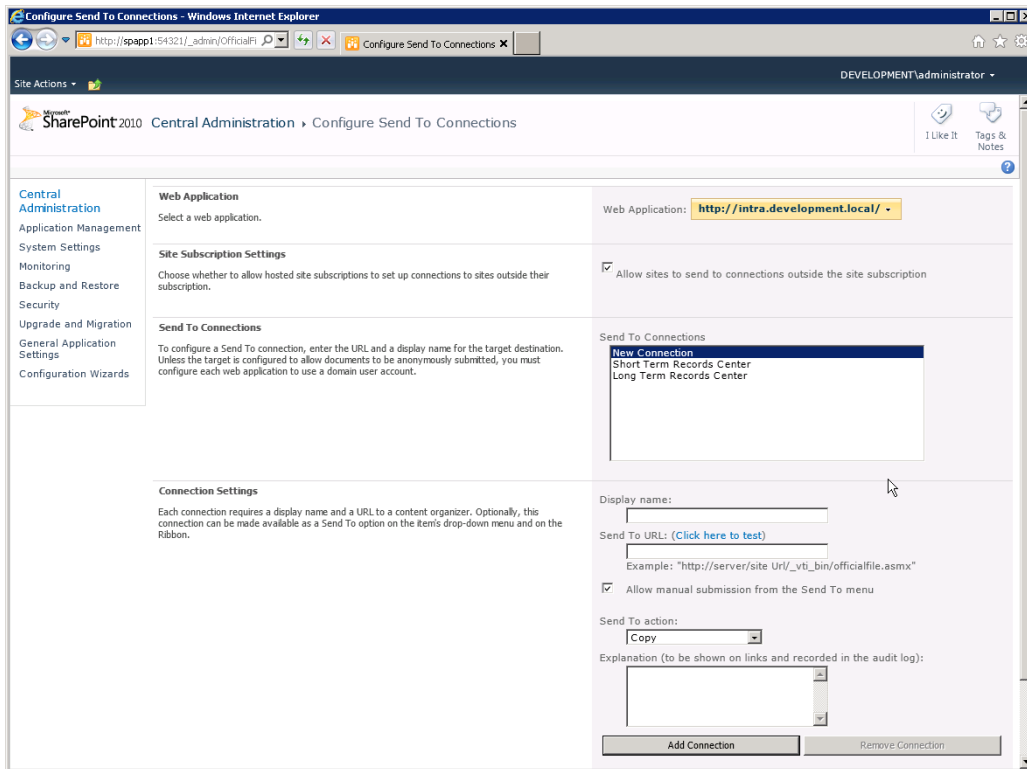


Figure 23 (Configure Send To Connections)

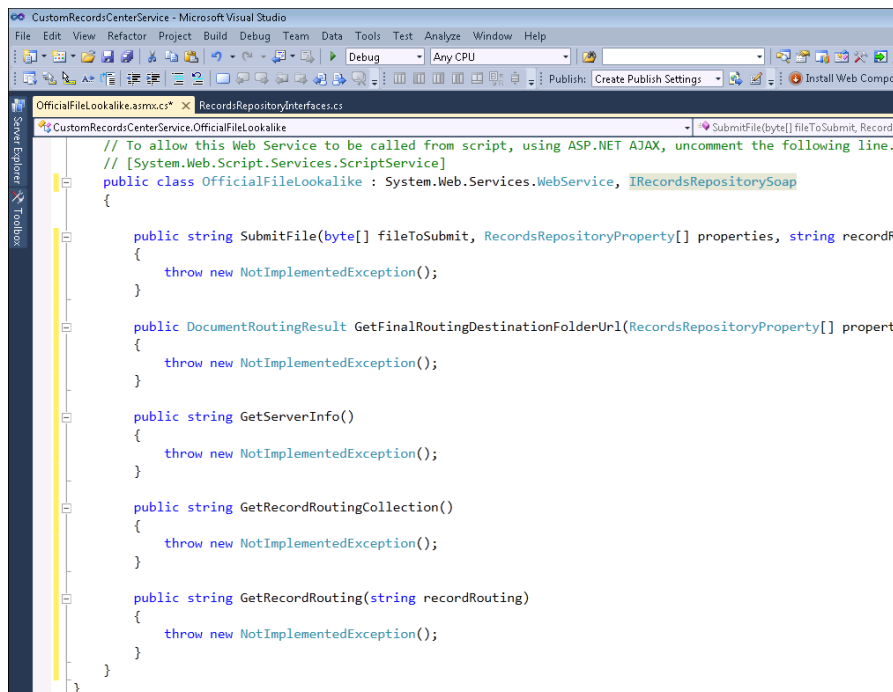


Figure 24 (Implementing the IRecordsRepositorySoap interface)

Once you've written the code to actually store the documents and return the results you can now very easily send records to your very own custom records repository, from multiple locations within SharePoint. The following screenshots show some of those integration areas.

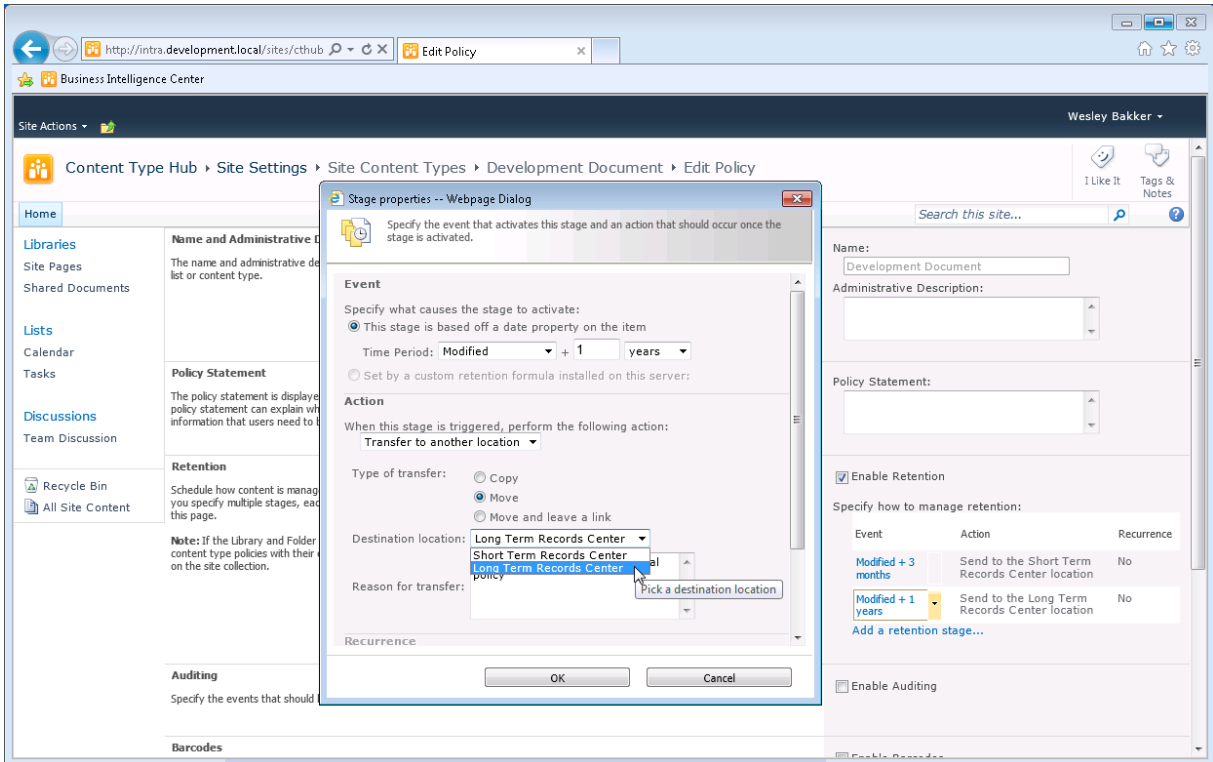


Figure 25 (Submit file by Retention Policy)

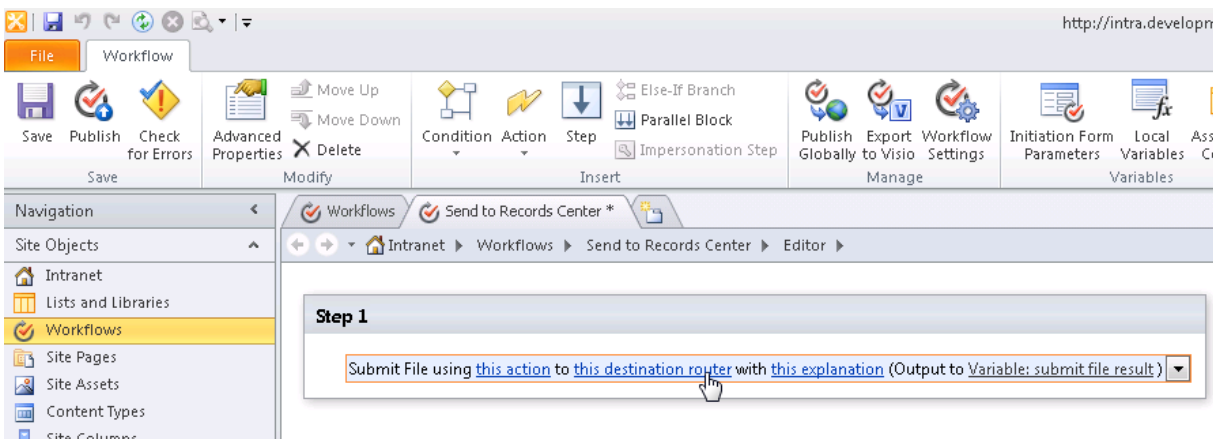


Figure 26 (Submit file by Workflow Action)

Shared Documents

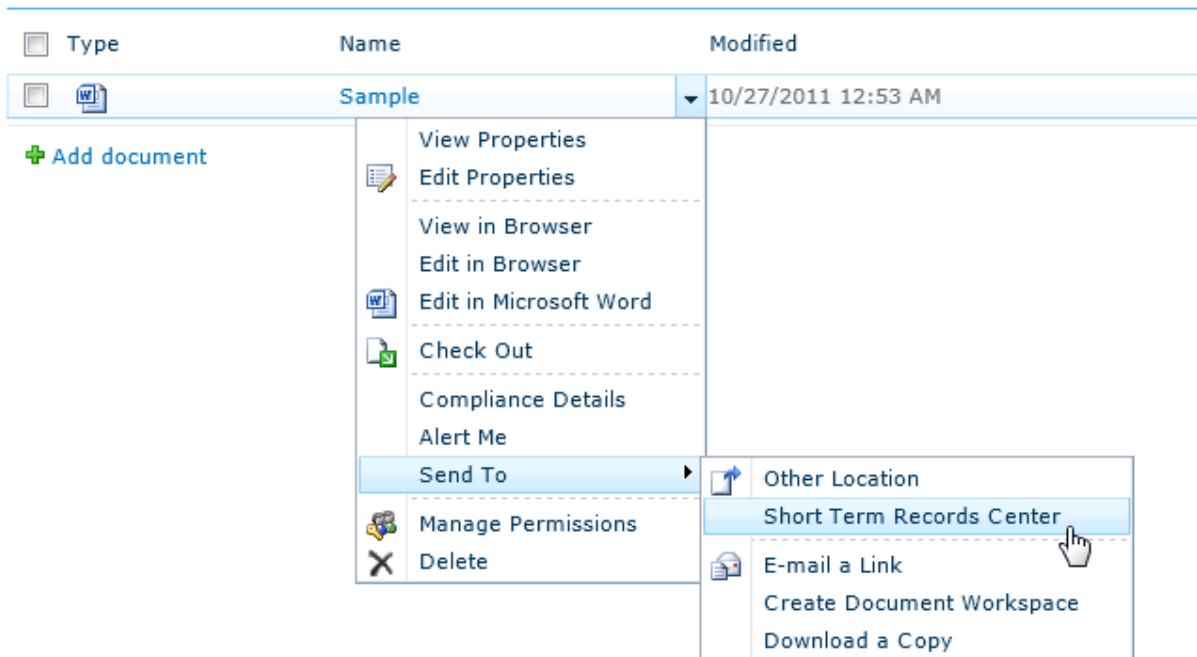


Figure 27 (Submit file from the ECB)

4.3 GUIDELINES

In this case there is just one thing that you should be aware of. You do have to be very careful to whom you open up your records repositories. As soon as you send a document to a records center any document specific permissions are lost!

## 5 SERVICE LAYERS

### 5.1 INTRODUCTION

Throughout this document we've mentioned service layers to proxy requests, and take care of request composition. It's easy to forget about the use and purpose of these service layers in integration projects even though they could and a lot of times should play an important role in these types of projects. This chapter briefly describes why service layers are important and takes a quick look at the possibilities of three commonly used service layers in SharePoint integration projects.

### 5.2 OVERVIEW

#### 5.2.1 WHY A SERVICE LAYER?

When you take a look at the online BCS samples, most of them show you how easy it is to connect to SQL Server tables. Nice indeed, but what if you are not the owner of this SQL database? What if an application update changes column or table names? Are you going to update all ECT you've defined previously? If you even can (think about the trouble a table split would cause you) it might incur a lot of rework in all sorts of places. Another sample on WCF connections: What if your products are now stored in two warehouses while they were stored in one previously? You would need to rewrite your BCS connection and change it to a custom .Net Assembly connector. Besides that, what if a third comes around, or what if the version or schema changes?

Traditionally applications were connected in an ad hoc point to point fashion (Figure 28)

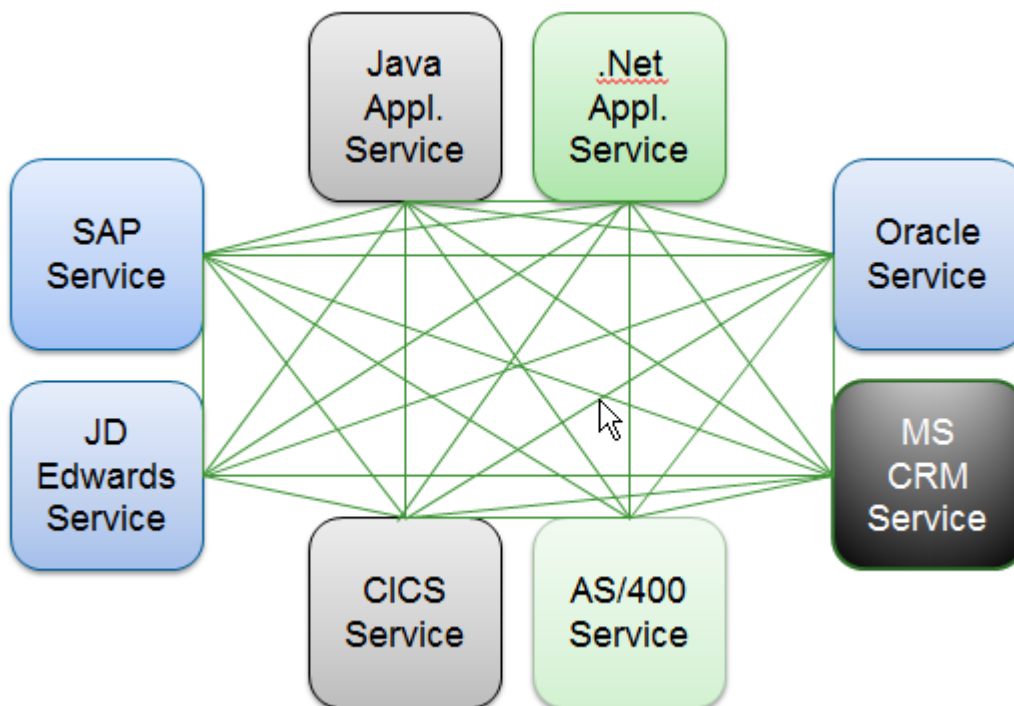


Figure 28 (Point to point connections)

As you can see in Figure 28 a change in one application causes a chain reaction that involves rework in all applications that are connected to it. To protect yourself from such disasters you should implement a service layer. Service layers can do a lot more for you though.

Some examples of what a service layer can do for you are:

- Location & Version Transparency
- Invocation & Orchestration
- Transaction handling
- Transport Protocol Conversion
- Error Handling & Repair
- Data Format Transformation
- Message Interactions Support
- Activity Monitoring

The next paragraphs describe the three most commonly used service layers beneath a SharePoint enterprise portal.

### 5.3 MOST COMMONLY USED SERVICE LAYERS

#### 5.3.1 SHAREPOINT SERVICE APPLICATIONS

SharePoint itself has built in support for creating “service layers”. With SharePoint 2010 comes a whole new topology mantra where we can manage resources by moving a service application from one “application” server to the other. If a service application on one server can’t keep up with the work to be done, we simply start it on another server as well and load balancing takes care of the message routing.

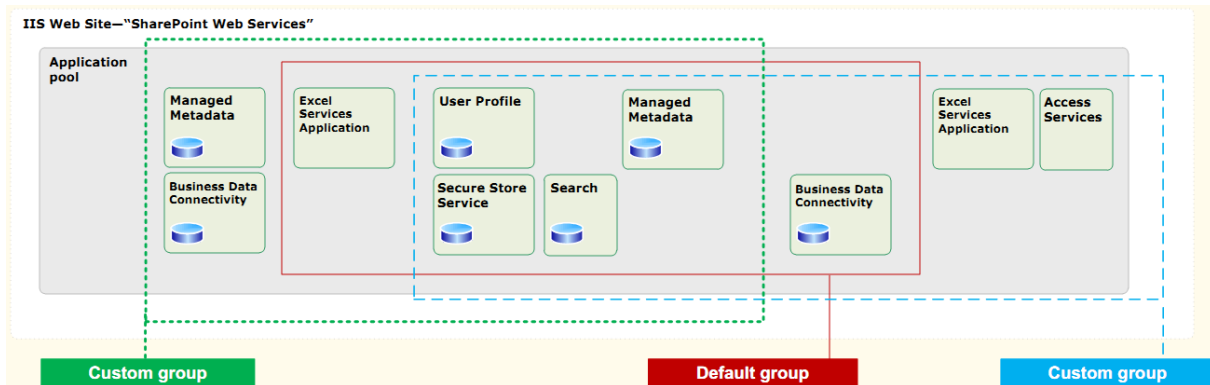


Figure 29 (Services configured in proxy groups)

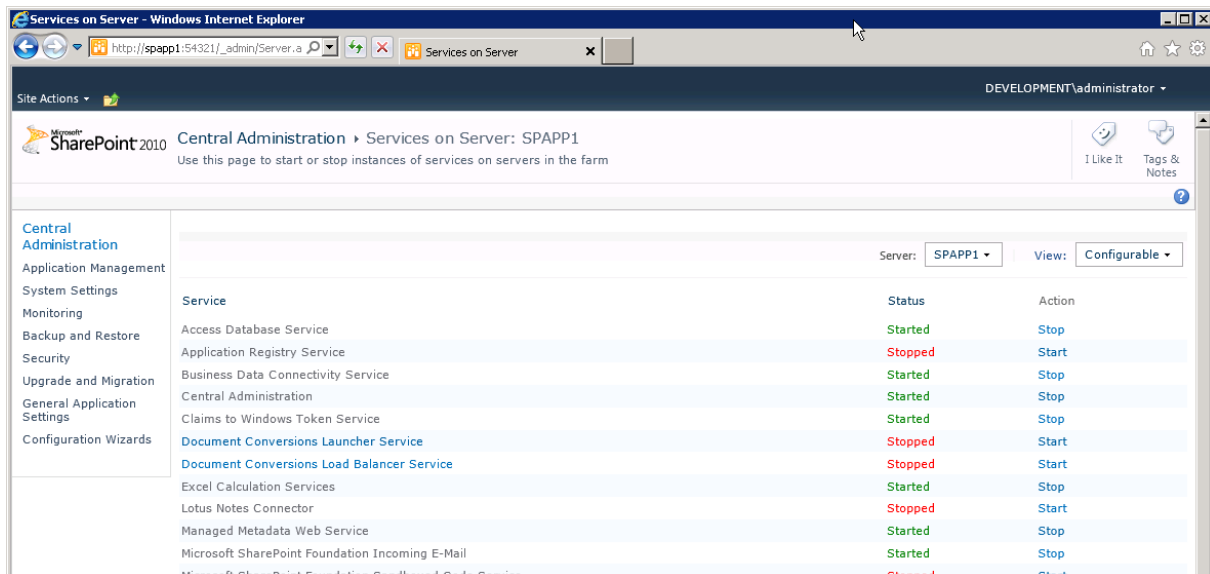


Figure 30 (Manage services on server)

Unfortunately however there is still no good documentation to date on how to create a decent service application. It involves many undocumented steps and as such is not the recommended approach even though it looks really tempting.

5.3.2 BIZTALK SERVER (ESB)

BizTalk Server as an integration server is a key part of Microsoft's application platform. It forms the heart of .Net technology-based Service Oriented Architectures (SOA) and is often used as an Enterprise Service Bus (ESB). The ESB is the technology you need in order to be able to implement a SOA within a company. SOA is the architectural style; ESB is the 'plumbing'.

BizTalk Server is perfect for creating an abstraction layer between applications and services (frontend and backend). Through 'loose' couplings, flexible routing, format and protocol conversion and proactive exception handling, any SharePoint enterprise portal can be turned into a manageable entity once again.

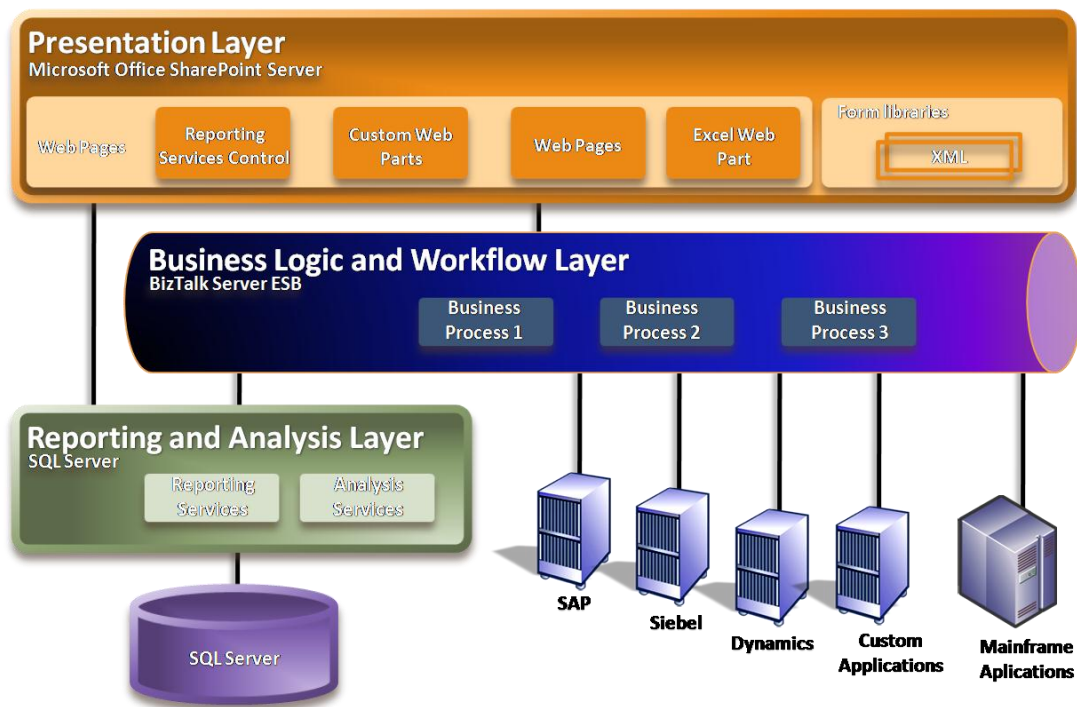


Figure 31 (ESB as a service layer)

Business processes can be modeled within the ESB, which can combine the various backend systems' functionalities and is able to expose them as composite services for consumption by SharePoint. The business process that runs in the ESB is responsible for orchestrating the interaction with all interfaces and presenting this process as a single transaction that can be rolled back if necessary. BizTalk also has a built-in business rules engine (BRE) and facilitates business activity monitoring (BAM).

Also, SharePoint's workflow capabilities can be invoked from this orchestration within the ESB. This allows the user to be involved within the automated processes at points where human interaction is required, such as when data deviates from standard business rules, or when the manual input of information is required.

The BizTalk Server ESB itself is an integration platform that ensures high availability and scalability, thanks to its built-in cluster architecture and grouping capabilities. For more details, please refer to (Veld, 2010) and (Veld, Bakker, 2009).

5.3.3 DUET ENTERPRISE

A lot of enterprise sized companies use SAP for their business management and SharePoint as their collaboration portal. Microsoft and SAP joined their forces and came up with a new product called DUET Enterprise to combine both process and collaboration. DUET Enterprise consists of several components which together enable seamless integration between SAP and SharePoint.

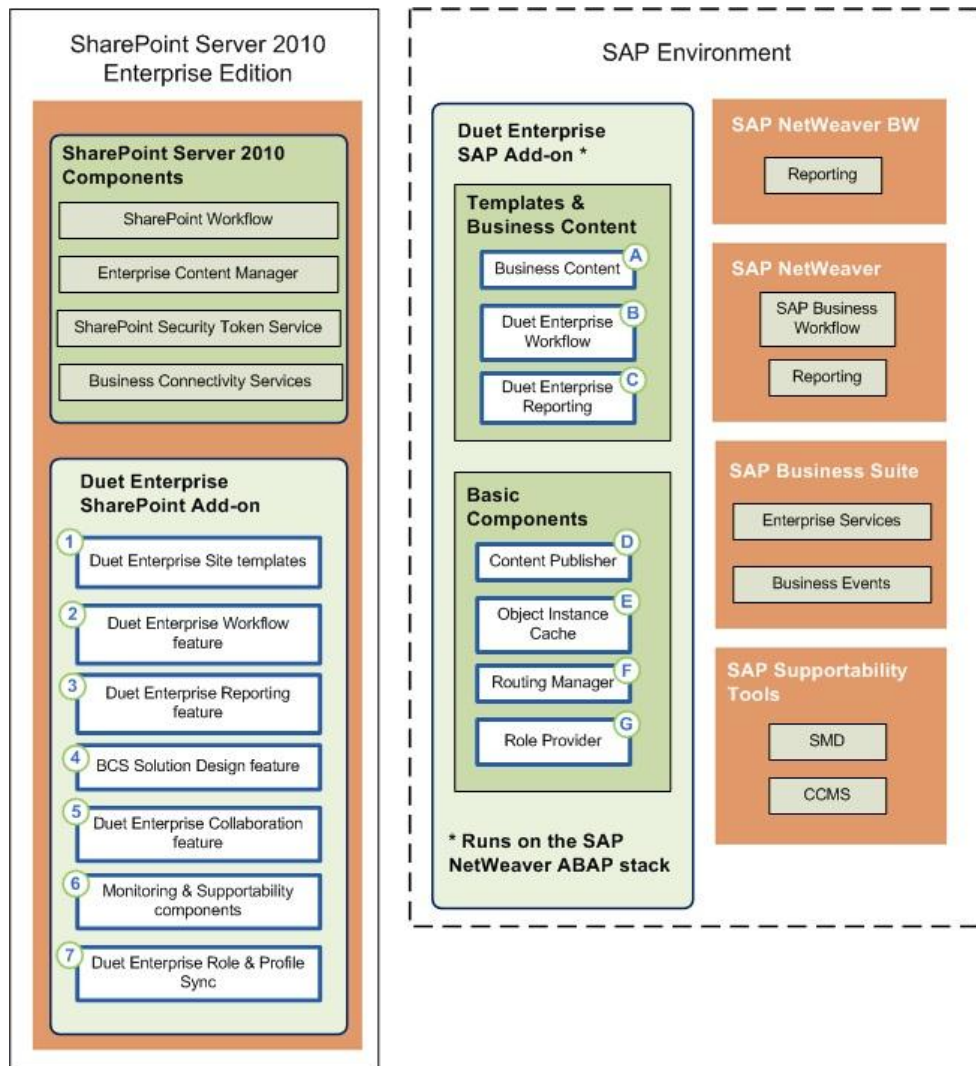


Figure 32 (DUET Components )

If we take a look at the components depicted in above figure, you'll notice that it is not only content you'll get from SAP. You also get workflow and reporting integration. The real bonus you get is the Role Provider that allows you to use SAP roles for SharePoint authorization. In that way you can store SAP artifacts on SharePoint with the correct SAP permissions and thus allow SharePoint collaboration on SAP content.

For more details, please refer to (Microsoft, 2010, December).

## 5.4 CONCLUSION

Service layers are invaluable in integration projects. Without service layers you're risking a lot of rework when you deploy future updates and you'll miss a lot of the built in features of service layers such as request composition, data format conversion, transactionality, auditability, etc.

Creating these features yourself in a custom built service layer can be a risky and time consuming business. BizTalk server and DUET Enterprise are the recommended way to go. Use DUET Enterprise for integration in SAP-centric environments and BizTalk server for integration in environments with multiple important back-end applications and other (cloud) services. The broad range of readily available adapters and built-in functionality for orchestration, transformation, auditability, security and scalability can save you a lot of development work.

On a last note, hybrid service layers are of course also possible, combining them and leveraging them in a fit-for-purpose way.

## 6 SUMMARY

SharePoint is more and more used as collaboration platform and enterprise portal, providing a true central workspace within many companies.

For that purpose, SharePoint offers a lot of integration handles. We've discussed BCS, Search and Record Repositories but there are more (with the Workflow External Data Exchange you can now call back into workflows for example, or you can use the Client Object Model to integrate your client applications into SharePoint). All these integration handles do have some specifics you need to be aware of when you start using them. This document tried to sum up most of these specifics and also tried to point out why you should consider using a service layer in integration projects.

Unfortunately we could not cover all topics as extensively as we wanted to. The size of this white paper would simply grow out of proportion. We believe however that this white paper can be a good starting point for those about to start a SharePoint integration project and hope you've enjoyed reading!

## 7 REFERENCES

- Veld, G., Bakker, W. (2009, September). *Biztalk + SharePoint: 1+1=3 Integration Best Practices*. Retrieved from Microsoft:  
<http://www.microsoft.com/download/en/details.aspx?id=21320>
- Krause, J., Langhirt, C., Sterff, A., Pehlke, B., & Döring, M. (2011). *SharePoint 2010 as a Development Platform*. Apress.
- lionelro. (2010, March 12). *Authenticating to Your External System*. Retrieved from blogs.msdn.com:  
<http://blogs.msdn.com/b/bcs/archive/2010/03/12/authenticating-to-your-external-system.aspx>
- Mechelke, T. (2010, 02 10). *Working with Complex Data Types in Business Connectivity Services*. Retrieved from MSDN:  
<http://blogs.msdn.com/b/bcs/archive/2010/02/11/working-with-complex-data-types-in-business-connectivity-services.aspx>
- Microsoft. (2010, December). *Duet Enterprise for Microsoft SharePoint and SAP Poster*. Retrieved from microsoft.com:  
<http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=3503>
- Microsoft. (2010, May). *Writing a Custom Security Trimmer for SharePoint Server Search*. Retrieved from msdn.microsoft.com: <http://msdn.microsoft.com/en-us/library/ee819930.aspx>
- Microsoft. (n.d.). *Official File Web Service Protocol Specification*. Retrieved from MSDN:  
[http://msdn.microsoft.com/en-us/library/cc313141\(v=office.12\).aspx](http://msdn.microsoft.com/en-us/library/cc313141(v=office.12).aspx)
- Rizzo, T., Alirezaei, R., Fried, J., Swider, P., Hillier, S., & Schaefer, K. (2010). *Professional 2010 Development*. Wrox.
- Veld, G. (2010, April). *ESB for SharePoint*. Retrieved from motion10:  
<http://www.motion10.com/bibliotheek/docs/ESB-for-SharePoint.pdf>
- White, E. (2010, april 27). *Searching External Data in SharePoint 2010 Using Business Connectivity Services*. Retrieved from blogs.msdn.com:  
<http://blogs.msdn.com/b/ericwhite/archive/2010/04/28/searching-external-data-in-sharepoint-2010-using-business-connectivity-services.aspx>

## 8 ABOUT THE AUTHORS AND MOTION10



Wesley Bakker ([wesley.bakker@motion10.com](mailto:wesley.bakker@motion10.com)) works as a Consultant, Lead Developer and Microsoft Certified Trainer at motion10 and has been involved with a number of the largest and most complex SharePoint 2010 implementations and migrations in the Netherlands. In May 2011 he became a Microsoft Certified Master (MCM) SharePoint 2010.



Gijs in 't Veld ([gijs.intveld@motion10.com](mailto:gijs.intveld@motion10.com)) is CTO at business integration specialist motion10 and as such responsible for all things technology and technological strategy. Since 2001, he has been involved in a great number of integration projects on the Microsoft Application Platform worldwide. Gijs has been awarded Microsoft Most Valuable Professional (MVP) 6 times.

motion10 specializes in integrating and presenting information from different sources. The solutions implemented by the company link enriching and disseminating information from business-critical applications and ensure that information is easily accessible, thus bringing more value from the information obtained. Motion10 has extensive experience with complex integrated environments (hundreds of BizTalk Server and SharePoint implementations) and can thereby eliminate the complexity of IT environments. motion10's solid, reliable IT environment has established that all employees have the personalized skills and technology to help with applications based on Microsoft technology platform utilizing BizTalk, SharePoint and SQL Server and related (cloud) services. For more information: [www.motion10.com](http://www.motion10.com).