# How To Integrate SAP Business Data Into SharePoint 2010 Using Business Connectivity Services And LINQ to SAP

Jürgen Bäurle

August 2010

Parago Media GmbH & Co. KG

#### Introduction

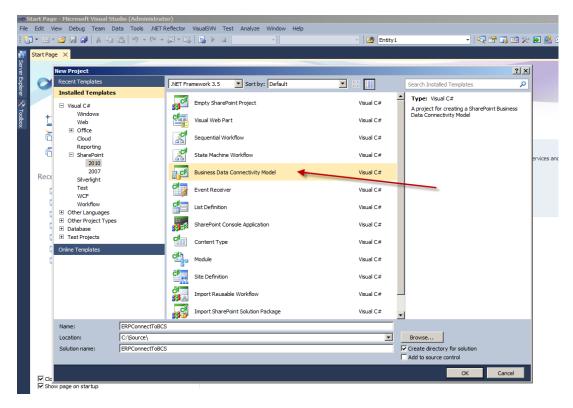
One of the core concepts of the Business Connectivity Services (BCS) for SharePoint 2010 are the external content types. They are reusable metadata descriptions of connectivity information and behaviours (stereotyped operations) applied to external data. SharePoint offers developers several ways to create external content types and integrate them into the platform. The SharePoint Designer 2010 for instance allows you to create and manage external content types that are stored in supported external systems. Such an external system could be SQL Server, WCF Data Service or a .NET Assembly Connector.

This article shows you how to create an external content type for SharePoint named *Customer* based on given SAP customer data. The definition of the content type will be provided as a .NET assembly and the data are displayed in an external list in SharePoint.

The SAP customer data are retrieved from the function module *SD\_RFC\_CUSTOMER\_GET*. In general, function modules in a SAP R/3 system are comparable with public and static C# class methods and can be accessed from outside of SAP via RFC (Remote Function Call). Fortunately we do not need to program RFC calls manually. We will use the very handy *ERPConnect* library from Theobald Software. The library includes a LINQ to SAP provider and designer that makes the our live easier.

# .NET Assembly Connector for SAP

The first step in providing a custom connector for SAP is to create a SharePoint project with the SharePoint 2010 Developer Tools for Visual Studio 2010. Those tools are part of Visual Studio 2010. We will use the *Business Data Connectivity Model* project template to create our project:



After defining the Visual Studio solution name and clicking the *OK* button, the project wizard will ask what kind of SharePoint 2010 solution you want to create. The solution must be deployed as a farm solution, not as a sandboxed solution. Visual Studio is now creating a new SharePoint project with a default BDC model (*BdcModel1*). You can also create an empty SharePoint project and add a *Business Data Connectivity Model* project item manually afterwards. This will also generate a new node to the Visual Studio Solution Explorer called *BdcModel1*. The node contains a couple of project files: The BDC model file (file extension *bdcm*), the *Entity1.cs* and *EntityService.cs* class files.

Next, we add a LINQ to SAP file to handle the SAP data access logic by selecting the LINQ to ERP item from the *Add New Item* dialog in Visual Studio. This will add a file called *LINQtoERP1.erp* to our project. The LINQ to SAP provider is internally called LINQ to ERP. Double click the LINQtoERP1.erp to open the designer. Now, drag the *Function* object from the designer toolbox onto the design surface. This will open the SAP connection dialog since no connection data have been defined so far:

ed IRPConnectToBCS - Microsoft Visual Studio (Administrator) File Edit View Project Build Debug Team Data Tools .NETReflector VisualSVN Test Analyze Window Help		
: 🔂 ▼ 🔛 ▼ 😂 🛃 👙 🕺 🐁 🗠 🕫 🕫 - < < = - < = - < = - < = - < = □ Debug - Any CPU	- M Entity1	- 🔍 🕾 📑 🖬 🛠 🖬 😫 🕴
Toobox     INQ to ERP Lep* × Bdddodel1.bdcm       D LNQ to ERP Unep* × Bdddodel1.bdcm       B WCube       Punter       D Doc       Y Query       Table       G General   LINQ to ERP Connection LINQ to ERP Info 1		Solution Explorer
There are no usab controls in this grou Drag an ilem to to text to add it to to toolbox. Password Extensions C Use Single Application Server Host SystemNumber Group / SID		
C Use HTTP / Web Service		LINQtoERP1.erp File Properties
HTTP UI		
Test Connection Cancel Ok		Build Action None  Copy to Output Directory Do not copy Custom Tool LINQtoERPGenerator Custom Tool Namespace File Name LINQtoERP1.erp
編 Server E., 父 Toolbox 愛 Error List 圓 Output 於 Find Symbol Results	•	Build Action How the file relates to the build and deployment processes.

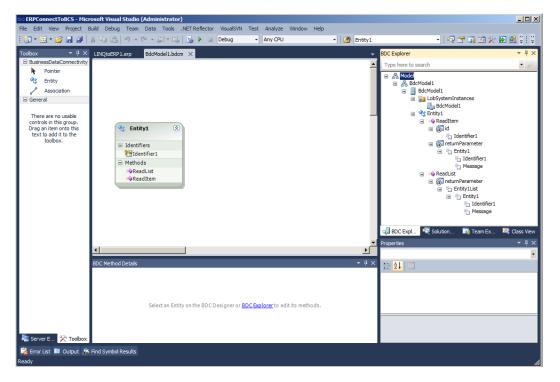
Enter the SAP connection data and your credentials. Click the *Test Connection* button to test the connectivity. If you could successfully connect to your SAP system, click the *Ok* button to open the function module search dialog. Now search for *SD\_RFC\_CUSTOMER\_GET*, then select the found item and click *Ok* to open the *RFC Function Module / BAPI* dialog:

<b>RFC Functi</b>	ion Modules / BAPI	s				
8	RFC Fur	nction Modules / BAP	ls			
Function Method Exports	SD_RFC_CUST	OMER_GET	S Modi	ier Public	Y	
				1	1	1
Pass	Туре	Name	Variable	Value	Structure Name	Edit
ঘ	String String	KUNNR NAME1				Edit Edit
			(	Cancel	Ok	

The dialog provides you the option to define the method name and parameters you want to use in your SAP context class. The context class is automatically generated by the LINQ to SAP designer including all SAP objects defined. Those objects are either C# (or VB.NET) class methods and/or additional object classes used by the methods.

For our project we need to select the export parameters KUNNR and NAME1 by clicking the checkboxes in the *Pass* column. These two parameters become our input parameters in the generated context class method named *SD\_RFC\_CUSTOMER\_GET*. We also need to return the customer list for the given input selection. Therefore we select the table parameter *CUSTOMER\_T* on the *Tables* tab. Then, click the *Ok* button on the dialog and the new objects gets added to the designer surface.

The LINQ designer has also automatically generated a class called *Customer* within the *LINQtoERP1.Designer.cs* file. This class will become our BDC model entity or external content type. But first we need to adjust and rename our BDC model that was created by default from Visual Studio. Currently the BDC model looks like this:



Rename the *BdcModel1* node and file into *CustomerModel*. Since we already have an entity class (*Customer*), delete the file *Entity1.cs* and rename the *EntityService.cs* file into *CustomerService.cs*. Next, open the *CustomerModel* file and rename the designer object Entity1. Then, change the entity identifier name from *Identifier1* to *KUNNR*. You can also use the BDC Explorer for renaming. The final adjustment result should look as follows:

BDC Explorer	<b>▼</b> ₽ ×
Type here to search	- 🔎
CustomerModel      CustomerModel      CustomerModel      CustomerModel      CustomerModel      Customer      Customer      WINR      Watel      Customer      Custome	
🚚 BDC Explo 💐 Solution E 📑 Team Expl 🧟 (	Class View

After	those	modi	ficati	ons	the	current	Sha	reP	oint	proj	ect	shou	ıld I	look	sim	nilar	to	the	next	scre	eensh	ot:

👓 ERPConnectToBCS - Microsoft Visual Studio (Administrator)	
File Edit View Project Build Debug Team Data Tools .NETReflector VisualSVN Test Analyze Window Help : 🛐 • 🗃 • 🚰 🚽 🎒 🗼 🖓 : 🖉 🖓 • 🔍 • 💭 • 🖏 🖓 🎍 📴 Debug 🔹 Any CPU	🕐 Entity1 🔹 🖓 🔐 📑 🚆
	Solution Explorer 🔹 후 🗙 👻
Server Explorer Customer Identifiers Identifiers Mode	
	Solution 'ERPConnectToBCS' (1 project)
r Customer 🔊	🛨 📲 Properties
A Identifiers	
	🗉 📲 Package
E Methods	GustomerModel     GustomerModel.bdcm
=@ReadList =@ReadItem	CustomerService.cs
- Keaditem	Sey.snk □ IINOtoERP1.erp
	LINQUERP1.erp LINQUERP1.erp LINQUERP1.erp
	LINQtoERP1.erp.diagram
	📲 BDC Expl 💐 Solution 🌆 Team Ex 🧟 Class View
BDC Method Details - Customer 👻 🕂 🛪	Properties 🔹 👎 🗙
Name Direction Type Descriptor	ERPConnectToBCS.Customer Entity
Methods	
E eadList	Name Customer
⊡ @ Parameters     ☐	Namespace ERPConnectToBCS
🔕 <add a="" parameter=""></add>	Name
	Indicates the name used in code to identify the object.
Redulsi	
🗒 Error List 🔳 Output 🔏 Find Symbol Results	
Item(s) Saved	

The last step we need to do in our Visual Studio project is to change the code in the *CustomerService.cs* class. The BDC model methods *ReadItem* and *ReadList* must be implemented using the automatically generated LINQ to SAP code. First of all, take a look at the code:

CustomerService.cs X LINQtoERP1.Designer.cs CustomerModel.bdcm	-
1 ⊡using System;	414
2 using System.Collections.Generic;	1
3	
4 ⊟namespace ERPConnectToBCS	
5 {	
6 🖻 public class CustomerService	
7 { 8 static SAPContext sc;	
9	
10 E static CustomerService()	
11 {	
12 ERPConnect.LIC.SetLic(""""""""""""""""""""""""""""""""""""	
13	
<pre>14 _sc = new SAPContext("TESTUSER", "Context");</pre>	
15 }	-
16	
17 🖻 public static Customer ReadItem(string id)	
18 {	
<pre>19 return _sc.SD_RFC_CUSTOMER_GET(id, string.Empty)[0];</pre>	
20 }	
21   22 ⊡ public static IEnumerable <customer> ReadList()</customer>	
23 {	
24 return sc.SD RFC CUSTOMER GET(string.Empty, "*");	
25 }	
26 }	
27 }	
28	
100 % • •	۲

As you can see we basically have just a few lines of code. All of the SAP data access logic is encapsulated within the SAP context class (see *LINQtoERP1.Designer.cs* file). The *CustomerService* class just implements a static constructor to set the *ERPConnect* license key and to initialize the static variable \_sc with the SAP credentials as well as the two BDC model methods.

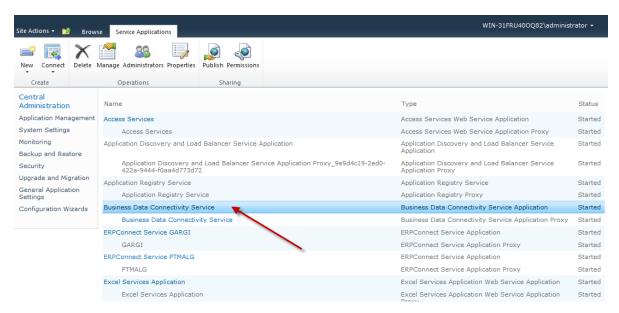
The *ReadItem* method, BCS stereotyped operation *SpecificFinder*, is called by BCS to fetch one specific item defined by the identifier *KUNNR*. In this case we just call the *SD\_RFC\_CUSTOMER\_GET* context method with the passed identifier (variable *id*) and return the first customer object get from SAP.

The *ReadList* method, BCS stereotyped operation *Finder*, is called by BCS to return all entities. In this case we just return all customer objects the *SD\_RFC\_CUSTOMER\_GET* context method returns. The returned result is already of type *IEnumerable*<*Customer*>.

The final step is to deploy the SharePoint solution. Right-click on the project node in the Visual Studio Solution Explorer and select *Deploy*. This will install and deploy the SharePoint solution on the server. You can also debug your code by just setting a breakpoint in the *CustomerService* class and executing the project with F5.

### That's all we have to do!

Now, start the SharePoint *Central Administration* panel and follow the link "Manage Service Applications" or navigate directly to the URL *http://<SERVERNAME>/\_admin/ServiceApplications.aspx*.



Click on Business Data Connectivity Service to show all available external content types:

Site Actions	- 🐋 Brows	se E	dit							WI	N-31FRU40OQ82\a	dministrator 👻
Import	Set Object Se Permissions	et Meta Permi	data Store	X Delete	Create/ Upgrade	Configure	External (	Content Types	•			
BDC Models	Perm	issions		Manage	Profil	e Pages		View				
Central Administra Application System Se Monitoring Backup and	Management ttings	Ser Nar		ication Inf	ormation	1	Business D	Data Connectivity Se	rvice			
Security		Sear	ch			<b>Q</b>						
General Ap Settings	nd Migration oplication ion Wizards		Name↑ Custome		Display N ERPConn	lame ectToBCS.C	Customer	Namespace ERPConnectToE	Vers 3CS 1.0.0		External System CustomerModel	Default Action

On this page we find our deployed BDC model including the *Customer* entity. You can click on the name to retrieve more details about the entity. Right now, there is just one issue open. We need to set permissions!

Mark the checkbox for our entity and click on the Set Object Permissions in the Ribbon menu bar. This will open the following permission dialog:

Site Actio	Set Object Permissions	□ ×	-
	You can set permissions on the objects in the BDC Metadata Store.	Help	
Import BDC Mode Centra Admini System Monitori Backup Security Upgrade General Settings	To add an account, or group, type or select it below and click 'Add'.  Add  WIN-31FRU400Q82\administrator  To remove an account, or group, select it above and click 'Remove'.  Permissions for WIN-31FRU400Q82\administrator: Edit		ion
Configu	Execute Selectable In Clients Set Permissions		
	OK Cancel		

Now, define the permissions for the users you want to allow to access the entity and click the *Ok* button. In the screen shown above the user administrator has all permissions possible.

In the next and final step we will create an external list based on our entity. To do this we open SharePoint Designer 2010 and connect us with the SharePoint web site.

🔀 🕼 भ 🤉 🐼	) 🗟 +   <del>-</del>			http://win-31fru40oq82 - Microso	it SharePoint Designer		Ξ <u>Σ</u> 3
File External Co		s				۵	0
	Z	k					
External External Content Type List	Edit Extern Content Ty	nal Delete /pe	Export BDC Model				
New	E	dit	Manage				
Navigation	<	Exter	nal Content Ty	pes 💾			×
Site Objects	^	<b>€ →</b> •	🚰 Home 🕨	External Content Types 🕨		¢	0
付 Home		Name		<ul> <li>External System</li> </ul>	▼ Type	▼ Namespace ▲	
Lists and Libraries		ERPConn	ectToBCS				
🧭 Workflows		Custome	er	CustomerModel	.NET Assembly	ERPConnectToBCS	
📴 Site Pages					······		
🔏 Site Assets							
Content Types							
Site Columns							
External Content Ty	pes						
📋 Data Sources							
Master Pages							
🕵 Site Groups							
🚮 Subsites							
🚞 All Files							
		•					Þ
8							

Click on *External Content Types* in the *Site Objects* panel to display all content types (see above). Double click on the *Customer* entity to open the details. The SharePoint Designer is reading all information available by BCS.

In order to create a external list for our entity click on *Create Lists & Form* on the Ribbon menu bar (see screenshot below) and enter *CustomerList* as name for the external list.

🔀। 🔙 🤟 🕲 🕲 🗟 📲 ।	+ http:	//win-31fru40oq82 - Microsoft Sh	arePoint Designer	_ 0 %
File External Content Typ	es			۵ 😮
Summary Operations View Design View Sys		ration Operation Title &	te Lists Go to Form List sts & Forms Profile Page Profile Page	x
Site Objects		al Content Types 🕨 Customer 🕨		(\$) (\$)
Home Lists and Libraries Workflows		and manage settings for this extr	ernal content type	
Site Pages Site Assets	External Content Type In		Permissions	^
Content Types	Key information about this	external content type.	Permissions for this external cont	Permissions
External Content Types	Display Name Namespace	ERPConnectToBCS.Customer ERPConnectToBCS		Edit, Execute, Set Permissions,
Data Sources Master Pages	Version Identifiers	1.0.0.0 KUNNR(String)	External Lists	
🕵 Site Groups 拱 Subsites	Office Item Type Offline Sync for external list		View and navigate to external lis	s bound to this external cont
External Content Types	External System	CustomerModel	Name	URL
ERPConnectToBCS	External Content Type O	perations	* Fields	^
Customer	Use this part to manage the	operations of this external conte		
	You can now create content type. You m writeback capabiliti external content type	oRT01	Type System.String System.String System.String	
	Name Type ReadList Read ReadItem Read I		STRAS	System.String
3				

Ok, now we are done!

Open the list and you should get the following result:

🔾 🗢 📴 http://win-3	1fru40oq82/Lists/CustomerL	ist/ReadList.a	aspx		•	🕨 🗙 🕒 Bing	<b>P</b>
Favorites 🔀 Customer	List - Customer List					🐴 🔹 🗟 🔹 🚍 🛻 🔹 Page 🔹 Safety 🕶	Tools + 🕡 +
ite Actions 👻 📩 Brows	List Tools se Items List					WIN-31FRU40OQ82\Admini	strator 🔸
New Polder Item	Edit tem Manage	Attach File	Workflows Approve/Reject	J Like It Tags ar	Tags & Notes		
ERPConnect Services	KUNNR 000000001	NAME1	x & Associates		ORT01 PHILADEL	STRAS	
Site Pages Shared Documents	0000000002		Ue Ü Oe Ö ss ß		Walldorf	Astorstrasse 34	
Lists	000000110	Auto Klem	ient		München	Bert-Brecht-Allee 29	
Calendar Tasks	0000000149 0000000150	Korean Au	tomotive Supply SA utomotive Supply		Havana Pyonyang	-	
Mail Templates CustomerList	0000000175 0000000224		n Multi Chemical Company e for B2C Customer		Mexico Cit LOS ANGE		
Discussions	0000000255 0000000256	Emma Bul Kenny A C			DENVER BOULDER	145 145 West 51 th street (TES 661 661 28 Street	Τ)
Feam Discussion	0000000257 0000000258	John Evan Roger Zah	-		NEDERLAN ALBUQUER		
A Recycle Bin	000000259	Laurel K. I Chelsa Qu			GREELEY ALBUQUE	610 610 8th Street RQUE 4650 4650 Pan American Freew	ay NE
and the content	000000261	Andrew W				-	

The external list shows all defined fields for our entity, even though our *Customer* class, automatically generated by the LINQ to SAP, has more than those four fields. This means you can only display a subset of information for your entity.

Another option is to just select those fields required within the LINQ to SAP designer. With the LINQ designer you can access not only SAP function modules. You can integrate other SAP objects, like tables, BW cubes, SAP Query or IDOCs. A demo version of the *ERPConnect* library can be downloaded from the Theobald Software homepage.

If you click the associated link of one of the customer numbers in the column KUNNR (see screenshot above), SharePoint will open the details view:

	List Tools				
Site Actions 👻 📷	Browse Items List				
Home > G	CustomerList - View It			□ ×	Ulike It Tags & Notes
Home Purchase F	View				• ۹
Libraries Site Pages Shared Documents	Edit Item Manage	15			Ave
Lists	-			Close	
Calendar Tasks	KUNNR NAME1	0000000110 Auto Klement			
Mail Templates CustomerList	STRAS	Bert-Brecht-Allee 29			
	ORT01	München			
Discussions Team Discussion				Close	
Recycle Bin		aurei к. наrdin helsa Quinn Yates	GREELEY	610 610 8th Street 4650 4650 Pan Ameri	can Freeway NE
			ALBUQUERQUE		

## Summary

This article has shown how easy and simple it is to integrate business data from SAP into the SharePoint platform using standard tools. Combing the powerful Microsoft Visual Studio 2010 with its SharePoint development tools and the handy LINQ to SAP provider tool from Theobald Software, you just need to write a couple of code lines to stick together all the logic we need to create an external list in SharePoint 2010 and BCS.

#### **Contact Information**

If you have any feedback or suggestions, please feel free to contact me:

Jürgen Bäurle jbaurle@parago.de http://www.parago.de/jbaurle

Parago Media GmbH & Co. KG Im Wengert 3 | 71336 Waiblingen, Germany | Phone +49.7146.861803 | Internet http://www.parago.de