



## Diplomarbeit

# **Microsoft SharePoint 2010 als Applikationsplattform**

vorgelegt bei: Dipl.-Inf. Jürgen Kühnlein  
von: Patrick Weber  
Matr.-Nr.: 877310  
Anschrift: Lissinger Str. 50, 54568 Gerolstein  
Abgabetermin: 01.05.2011

# Inhaltsverzeichnis

<b>Glossar</b>	<b>III</b>
<b>Abkürzungsverzeichnis</b>	<b>VII</b>
<b>Tabellenverzeichnis</b>	<b>IX</b>
<b>Abbildungsverzeichnis</b>	<b>X</b>
<b>1 Einleitung</b>	<b>1</b>
1.1 Problemstellung . . . . .	1
1.2 Zielsetzung . . . . .	1
1.3 Abgrenzung . . . . .	2
1.4 Aufbau . . . . .	2
1.5 Über Rowa Automatisierungssysteme GmbH . . . . .	3
<b>2 Grundlagen</b>	<b>4</b>
2.1 Was sind Collaborationsplattformen . . . . .	4
2.2 Collaboration im Unternehmen . . . . .	5
2.2.1 Bedeutung von Collaboration . . . . .	5
2.2.2 Web 2.0 im Unternehmen . . . . .	6
2.2.3 Auswirkungen auf das Unternehmen . . . . .	7
2.3 Was ist SharePoint 2010 . . . . .	8
2.4 Historie von SharePoint . . . . .	11
<b>3 SharePoint als Applikationsplattform</b>	<b>13</b>
3.1 Warum SharePoint als Applikationsplattform . . . . .	14
3.1.1 Gründe . . . . .	14
3.1.2 Vorteile . . . . .	14
3.1.3 Nachteile . . . . .	15
3.1.4 Wofür ist es geeignet . . . . .	16
3.1.5 Wofür ist es nicht geeignet . . . . .	16
3.2 Möglichkeiten der Lösungsbereitstellung . . . . .	18
3.2.1 Grundlagen . . . . .	18
3.2.2 Farm Lösung . . . . .	19
3.2.3 Sandbox Lösung . . . . .	20
3.2.4 Features . . . . .	21
3.3 Möglichkeiten der Entwicklung . . . . .	24
3.3.1 Business Connectivity Services . . . . .	24
3.3.2 WebPart . . . . .	35
3.3.3 Workflow Foundation . . . . .	46
3.3.4 Windows Communication Services . . . . .	59
3.3.5 Access Services . . . . .	67
3.3.6 Excel Services . . . . .	76
<b>4 Bewertung und Ausblick</b>	<b>85</b>
4.1 Bewertung der Ergebnisse . . . . .	85
4.2 Ausblick . . . . .	86
<b>Literaturverzeichnis</b>	<b>88</b>

<b>Quellcode Verzeichnis</b>	<b>94</b>
<b>A Anhang</b>	<b>95</b>
A.1 SharePoint 2010 Development Platform Stack . . . . .	95
A.2 Historie von SharePoint . . . . .	96
A.3 External Content Type im SharePoint Designer . . . . .	97
A.4 Temporäre BCS Verbindungen . . . . .	97
A.5 BCS Foreign-Key Beschränkung . . . . .	98
A.6 Beziehungen zwischen External Content Types . . . . .	99
A.7 Definition eines Custom Timer Jobs . . . . .	100
A.8 Definition eines FeatureReceivers zum Erstellen Custom Timer Jobs	104
A.9 WebPart Eigenschaften in der Tool Pane . . . . .	110
A.10 Verbundene WebParts . . . . .	111
A.11 Grafischer Silverlight Designer in Visual Studio . . . . .	112
A.12 SharePoint Silverlight Anwendung . . . . .	113
A.13 Site Workflows im Site Settings Menü . . . . .	120
A.14 Task Wizard des SharePoint Designers . . . . .	121
A.15 Personalanforderungsworkflow . . . . .	121
A.16 Definition eines AssociationForm . . . . .	124
A.17 Definition der InfoPath Task Formulare in der Workflowdefinition .	127
A.18 Definition eines Task Content Types . . . . .	128
A.19 Definition eines ASPX Task Formulars . . . . .	130
A.20 Erstellen eines Tasks mit Content Type . . . . .	135
A.21 Schnittstellendefinition einer Custom Workflow Activity . . . . .	137
A.22 Quellcode einer Custom Workflow Activity . . . . .	138
A.23 WebPart zum Anzeigen von Daten per REST Schnittstelle . . . . .	141
A.24 Anlegen von Websites per WCF Dienst . . . . .	145
A.25 Jobüberwachung per WCF Dienst . . . . .	153
A.26 WCF Dienst zum Hochladen und Berechtigen von Dokumenten per Client Objekt Modell . . . . .	156
A.27 Festlegen eines Navigationsformulars einer Web Datenbank . . . . .	161
A.28 Elemente einer Web Datenbank . . . . .	161
A.29 Website Inhalt einer Web Datenbank . . . . .	162
A.30 Access Services Fehlermeldung bei unzureichenden Berechtigun- gen . . . . .	162
A.31 Startseite der Schulungsdatenbank . . . . .	163
A.32 InfoPath Designer 2010 mit Listenformular . . . . .	163
A.33 Vertrauenswürdige Speicherorte der Excel Services . . . . .	164
A.34 Unattended Service Account der Excel Services . . . . .	164
A.35 Identität einer Web Application . . . . .	165
A.36 Excel Services Parameter und benannte Bereiche . . . . .	165
A.37 Reporting Services Bericht . . . . .	166
A.38 Bereitstellungsziel eines Reporting Services Berichts . . . . .	166
<b>B Eidesstattliche Erklärung</b>	<b>167</b>

## Glossar

**Active Directory** Verzeichnisdienst von Microsoft zur Zuordnung von Benutzerdaten zu Anmeldeinformationen sowie zur Berechtigungsverwaltung.

**Application Pool** Eine Sammlung von URL Adressen (Webseiten), die von einem IIS Prozess (Worker Prozess) verarbeitet werden.

**ASP.NET** Advanced Server Pages, Skriptsprache zum serverseitigen Erstellen von Webseiten. Alle gängigen .NET Sprachen wie C# oder Visual Basic werden unterstützt.

**AtomPub** Atom Publishing Protocol. XML basiertes Format zum plattformunabhängigen Austausch von Informationen.

**Bread crumb Navigation** Die Brotkrumen Navigation ist eine Navigationsleiste, die den zurückgelegten Navigationspfad (meist im oberen Bereich einer Seite) anzeigt.

**Business Connectivity Service** Eine Servicearchitektur in SharePoint 2010 zum Anbinden von Geschäftsdaten aus Drittsystemen.

**Business Intelligence** computergestützte Sammlung, Auswertung und Darstellung von Geschäftsdaten zur Unterstützung unternehmenswichtiger Entscheidungen.

**Central Administration** Administrationsumgebung für SharePoint Server. Die Central Administration selbst basiert auf dem SharePoint Server.

**Client Objekt Modell** SharePoint Objektmodell für Clientanwendungen. Hierbei handelt es sich um eine Fassade für zahlreiche Web Services. Es ist nur ein Subset des kompletten Objektmodells.

**Collaboration** beschreibt einen Leistungsbereich von Software, welche Zusammenarbeit von Mitarbeitern unterstützt.

**Content Type** Wiederverwendbare Sammlung von Metadaten, Workflows, Formularen etc. Metadaten werden so an einem Ort deklariert und können in anderen Listen wiederverwendet werden.

**Enterprise 2.0** bezeichnet das Nutzen von Web 2.0 Techniken und social Software im Unternehmen.

**Enterprise Wiki** bezeichnet eine Funktion in SharePoint. Es ist ein Wiki, welches durch Veröffentlichungsfunktionen wie etwa Genehmigungsworkflows und Metadatenverwaltung erweitert wird.



**External Content Type** Externer Inhaltstyp zur Beschreibung von Geschäftsdaten aus Drittsystemen.

**Groupware** klassische Systeme zur Zusammenarbeit wie E-Mail oder freigegebene Kalender.

**JSON** JavaScript Object Notation. Von Menschen lesbares, textbasiertes Datenformat zum Datenaustausch zwischen Anwendungen.

**Master Page** Definition des Layouts von Webseiten. Beim Erzeugen der Ausgabe wird die Master Page und die angeforderte Webseite zusammengeführt.

**Master Pages** Vorlagenseite in ASP.NET mit Platzhaltern und Layout Definitionen.

**MS Content Management Server** Web Content Management System von MS zur Verwaltung von Inhalten im Intra- und Internet, welches in die SharePoint Entwicklung eingeflossen ist.

**MS Digital Dashboard** Portallösung von MS, welches in die SharePoint Entwicklung eingeflossen ist.

**MS Exchange Server** Collaborations- und Nachrichtendienst von MS. Dient hauptsächlich als E-Mail Server mit Kontakt- und Kalenderverwaltung.

**MS Office SharePoint Server 2007** Collaborationsplattform von MS. Basiert auf den WSS, bietet zusätzliche Funktionen für Unternehmen wie social media und BI.

**MS Workflow Foundation** Workflow Umgebung mit Laufzeitumgebung und Designer zur Implementierung langlaufender Prozessabläufe.

**Multidimensional Expressions** Datenbanksprache zur Abfrage von Data Warehouse Systemen auf Basis der Microsoft Analysis Services.

**MySite** Profilseiten von Mitarbeitern. Auf diesen Seiten stellen Mitarbeiter Informationen über sich bereit. Stellt eine Implementierung eines sozialen Netzwerkes für Unternehmen dar.

**phonetische Suche** Suchmechanismus, bei dem eine Suchmaschine anders geschriebene, aber gleich ausgesprochene Suchbegriffe als Alternative vorschlägt.

**Pivot Tabelle** Dynamische, durch den Benutzer anpassbare tabellarische Darstellungsform von Daten ohne die Notwendigkeit, die Strukturen der ursprünglichen Daten anpassen zu müssen..

**REST** Spezielle Art von Web Services. Einzelne Objekte werden per URI mit Parameterübergabe direkt adressiert.

**Ribbon UI** Menüleiste mit Registerkarten im oberen Bildbereich, vergleichbar mit der Oberfläche von MS Office Systemen.

**Secure Store Service** Dienst der BCS Architektur zur verschlüsselten Speicherung von Benutzerkennungen.

**SharePoint** Produktfamilie zur webbasierten Collaboration von Microsoft.

**SharePoint Designer** frei erhältliches Entwicklungswerkzeug für SharePoint von MS.

**SharePoint Foundation 2010** Grundfunktionen der aktuellen SharePoint Plattform und Bestandteil von Microsoft Windows 2008 Server. Basis für SharePoint Server 2010.

**SharePoint Online** SharePoint System, welches von einem Provider gegen Gebühr bereitgestellt wird.

**SharePoint Server 2010** Funktionserweiterung der SharePoint Foundation 2010 für Unternehmen.

**Silverlight** Anwendungsplattform von MS zum Erstellen und Ausführen von Rich Internet Applications. Die Laufzeitumgebung ist als Browser Plugin für die gängigen Webbrowser erhältlich.

**Tagging** oder Folksonomie beschreibt die freie Verschlagwortung von Inhalten durch einen Benutzer.

**Taxonomie** Schlagwort eines Inhaltes, welches in einer Hierarchie von Schlagwörtern eingeordnet ist. Taxonomie ist das Gegenteil von Folksonomie.

**UserControl** Wiederverwendbares Benutzersteuerelement, das von einer ASP.NET Webanwendung ausgeführt wird.

**Visual Studio 2010** Entwicklungswerkzeug von Microsoft.

**Web 2.0** Beschreibung für ein verändertes Nutzen von internetbasierenden Systemen. Benutzer erstellen und bearbeiten Inhalte selbst.

**Web Frontend Server** Webserver, der Anfragen von Clients entgegennimmt. Hier wird auch der ASP.NET Code ausgeführt.

**Web Services** Dienst, der über eine URI eindeutig identifiziert werden kann und Nachrichten per XML austauscht.

**Web Storage System** Hierarchisches Ordnersystem zum Speichern beliebiger Inhalte, Speichersystem des Exchange Server 2000.

**WebPart** ist ein mit ASP.NET erstelltes Steuerelement oder Programmfunktion, welches der Benutzer auf einer Seite frei positionieren kann. Der Benutzer ist so in der Lage, Inhalte und Programmfunktionen nach seinen Wünschen auf einer Seite anzuordnen.

**Windows Communication Foundation** Dienstorientierte Kommunikationsplattform, welche verschiedene Kommunikationsstandards unter einer API zusammenfasst.

## Abkürzungsverzeichnis

**AtomPub** Atom Publishing Protocol.

**BCS** Business Connectivity Service.

**BDC** Business Data Catalog.

**BI** Business Intelligence.

**CA** Central Administration.

**CAML** Collaborative Application Markup Language.

**CAS** Code Access Security.

**ECT** External Content Type.

**ERP** Enterprise Ressource Planning.

**GAC** Global Assembly Cache.

**IIS** Internet Information Services.

**JSON** JavaScript Object Notation.

**LOB System** Line-of-Business System.

**MDX** Multidimensional Expression.

**MOSS 2007** Microsoft Office SharePoint Server 2007.

**MS** Microsoft.

**MSMQ** Microsoft Message Queue.

**REST** Representational State Transfer Architektur.

**Rowa** Rowa Automatisierungssysteme GmbH.

**SDK** Software Development Kit.

**SSS** Secure Store Service.

**URN** Uniform Resource Name.

**WCF** Windows Communication Foundation.

**WF** Windows Workflow Foundation.

**WSS 2.0** Windows SharePoint Services 2.0.

**WSS 3.0** Windows SharePoint Services 3.0.

**XML** eXtensible Markup Language.

## Tabellenverzeichnis

1	BCS Funktionen nach SharePoint Versionen (eigene Darstellung)	32
2	Vergleich der Entwicklungswerkzeuge für BCS (eigene Darstellung)	33
3	Empfehlungen für die Nutzung von BCS (eigene Darstellung) . . .	34
4	WebPart Arten (eigene Darstellung) . . . . .	44
5	Allgemeine Empfehlungen bei der Entwicklung von WebParts (eigene Darstellung) . . . . .	45
6	Workflow Funktionen nach SharePoint Versionen (eigene Darstellung) . . . . .	56
7	Vergleich der Entwicklungswerkzeuge für Workflows (eigene Darstellung) . . . . .	57
8	Handlungsempfehlungen beim Entwickeln von Workflows (eigene Darstellung) . . . . .	58
9	Handlungsempfehlungen zur Verwendung der Service Factorys (eigene Darstellung) . . . . .	65
10	Handlungsempfehlungen beim Entwickeln und Nutzen von WCF Diensten für SharePoint (eigene Darstellung) . . . . .	66
11	Handlungsempfehlungen bei der Verwendung von Access Services (eigene Darstellung) . . . . .	75
12	Handlungsempfehlungen bei der Verwendung der Excel Services (eigene Darstellung) . . . . .	84

# Abbildungsverzeichnis

2.1	Collaborationswerkzeuge und Anforderungen (Paulke u. a. [2008], S.15)	6
2.2	Funktionsbereiche von SharePoint 2010 (Microsoft [2010d])	8
2.3	SharePoint 2010 Versionsübersicht (eigene Darstellung)	9
2.4	Quantitativer Vergleich des Funktionsumfangs der SharePoint Versionen (eigene Darstellung)	11
3.1	Hierarchieebenen in SharePoint (eigene Darstellung)	18
3.2	MS SQL Profiler zeigt, dass der Datenzugriff sofort stattfindet (eigene Darstellung)	27
3.3	WebPart Zonen (eigene Darstellung)	35
3.4	Silverlight Anwendung (eigene Darstellung)	43
3.5	Sequentieller Workflow (eigene Darstellung)	46
3.6	State Machine Workflow (eigene Darstellung)	47
3.7	Workflow Definition im SharePoint Designer (eigene Darstellung)	51
3.8	Definition des Workflows zum Einreichen von Verbesserungsvorschlägen (eigene Darstellung)	52
3.9	Eigenschaften der CustomActivity (eigene Darstellung)	56
3.10	WebPart stellt Daten per REST API dar (eigene Darstellung)	62
3.11	Web Datenbank im Browser (eigene Darstellung)	68
3.12	Vergleich des Layouts einer Website mit einer Web Datenbank (eigene Darstellung)	70
3.13	Besucherverwaltung mit Berichten (eigene Darstellung)	71
3.14	Kompatibilitätsfehlermeldungen (eigene Darstellung)	72
3.15	Kundenfeedback Formular (eigene Darstellung)	74
3.16	Excel Arbeitsmappe im Browser (eigene Darstellung)	76
3.17	Fehlermeldung der Excel Services bei Verwendung der Secure Store Services (eigene Darstellung)	77
3.18	Benannte Zelle in Excel (eigene Darstellung)	78
3.19	Excel Arbeitsmappe mit Filter WebPart (eigene Darstellung)	79
3.20	Belegungsplan der Verladerampen (eigene Darstellung)	80
3.21	Reporting Services Bericht in SharePoint (eigene Darstellung)	83
A.1	SharePoint 2010 Development Platform Stack (MSDN [2010g])	95
A.2	Die Geschichte von SharePoint (Joining Dots [2006])	96
A.3	Mit dem SharePoint Designer können External Content Types erstellt werden (eigene Darstellung)	97
A.4	Temporäre Verbindungen zum Drittsystem bleiben nach Beendigung der Entwicklung zurück (eigene Darstellung)	97
A.5	Fehlermeldung wenn Foreign-Key Beschränkung verletzt wird (eigene Darstellung)	98
A.6	Auswirkung von Löschoperationen in den BCS (eigene Darstellung)	99
A.7	WebPart Eigenschaften in der Tool Pane (eigene Darstellung)	110
A.8	Verbindung zwischen zwei WebParts (eigene Darstellung)	112
A.9	Grafischer Silverlight Designer (eigene Darstellung)	112
A.10	Site Workflows im Site Settings Menü (eigene Darstellung)	120
A.11	Task Wizard des SharePoint Designers (eigene Darstellung)	121
A.12	erstes Task Formular (InfoPath) (eigene Darstellung)	121
A.13	State Machine der Personalanforderung (eigene Darstellung)	122
A.14	Inhalt des Initialen States (eigene Darstellung)	122

A.15 Inhalt der EventDriven Aktivität beim Ändern des Tasks (eigene Darstellung) . . . . .	123
A.16 Festlegen des Navigationsformulars (eigene Darstellung) . . . . .	161
A.17 Elemente einer Web Datenbank (eigene Darstellung) . . . . .	161
A.18 Website Inhalt einer Web Datenbank (eigene Darstellung) . . . . .	162
A.19 Meldung bei unzureichenden Berechtigungen (eigene Darstellung)	162
A.20 Navigationsformular der Schulungsdatenbank (eigene Darstellung)	163
A.21 InfoPath Designer 2010 mit Listenformular (eigene Darstellung) . .	163
A.22 vertrauenswürdige Speicherorte (eigene Darstellung) . . . . .	164
A.23 Unattended Service Account (eigene Darstellung) . . . . .	164
A.24 Identität der Web Application (eigene Darstellung) . . . . .	165
A.25 Definition der verfügbaren Elemente einer Excel Arbeitsmappe (eigene Darstellung) . . . . .	165
A.26 Definition eines Reporting Services Berichts (eigene Darstellung) .	166
A.27 Bereitstellungsziel eines Reporting Services Berichts (eigene Darstellung) . . . . .	166



# 1 Einleitung

Mit zunehmender Verbreitung der Internettechnologie und besonders der Web 2.0 Funktionen hat die Bedeutung von Portalen und Collaborationsplattformen im Unternehmen zugenommen. So halten nach einer Studie von Kelton Research <sup>1</sup> 53% der Unternehmen Collaboration für wichtiger als die Spezialisierung von Mitarbeitern und sehen darin einen wichtigen Schlüssel zu langfristigem Unternehmenserfolg <sup>2</sup>. Microsoft (MS) hat sich in diesem Bereich mit seiner SharePoint Produktreihe positioniert und am 12.05.2010 seine neuesten Versionen, die SharePoint Foundation 2010 und SharePoint Server 2010, veröffentlicht.

## 1.1 Problemstellung

SharePoint hat seit dem letzten Release (MS Windows SharePoint Services 3.0 (WSS 3.0)) eine weite Verbreitung erreicht und ist laut zahlreichen Meldungen, wie etwa von Mario Thiessenhusen <sup>3</sup>, das am schnellsten wachsende Serverprodukt von Microsoft. Jedoch steht, entsprechend einer Studie von Pentadoc, bei der Einführung von SharePoint oftmals das reine Dokumenten- und Wissensmanagement im Vordergrund <sup>4</sup>. Auch bei bestehenden Lösungen werden die Potentiale von SharePoint als Applikations- und Integrationsplattform nicht ausgeschöpft. Die Rowa Automatisierungssysteme GmbH (Rowa) setzt bereits seit einigen Jahren SharePoint ein, nutzt aber ebenfalls hauptsächlich das Dokumentenmanagement. Zudem besteht die gewachsene IT Landschaft bei Rowa trotz Konsolidierung zum Teil aus Insellösungen und selbstentwickelten Systemen, die den aktuellen Anforderungen nicht mehr gerecht werden. Zahlreiche Versuche, bestehende Systeme oder Geschäftsprozesse in dem bestehenden SharePoint abzubilden, sind fehlgeschlagen. Meist wurde bei der Umsetzung eine Technologie gewählt, die den Anforderungen nicht gerecht wurde. Dies führte zu einer Neuentwicklung oder dem endgültigen Scheitern der Integration.

## 1.2 Zielsetzung

Diese Arbeit soll eine Entscheidungsgrundlage für zukünftige Projekte bieten und Anwendungsfälle für die verschiedenen Technologien skizzieren. Es wer-

---

<sup>1</sup>DreyBig [2010]

<sup>2</sup>Bube [2010]

<sup>3</sup>Thiessenhusen [2008]

<sup>4</sup>Tylla [2010]

den verschiedene Methoden und Technologien vorgestellt, mit denen SharePoint als Applikations- und Integrationsplattform genutzt werden kann. Zudem werden die notwendigen Schritte zur Implementierung zwecks Aufwandsschätzung beschrieben und je Technologie Anwendungsszenarien skizziert. Durch diese Betrachtung soll die Auswahl der Technologie, mit der zukünftige Anforderungen am geeignetsten umgesetzt werden können, erleichtert werden.

### **1.3 Abgrenzung**

Der Fokus dieser Arbeit liegt in den Möglichkeiten zur Implementierung von Geschäftsprozessen und Integration von bestehenden IT Systemen. Es werden keine Standardfunktionen von SharePoint vorgestellt, sondern nur ausgewählte Technologien. Jede Technologie wird kurz dargestellt sowie die schematische Vorgehensweise beschrieben. Ein vollständiges oder umfangreiches Beispiel wird nicht gegeben.

### **1.4 Aufbau**

Nach der Eingrenzung der Aufgabenstellung und Zielsetzung im ersten Kapitel, werden im zweiten Kapitel einige Grundlagen erläutert. Zunächst wird dargestellt, was man unter Collaborationsplattformen versteht, welche Formen es gibt und wie diese im Unternehmen eingesetzt werden. Anschließend werden die Auswirkungen von Collaborationsplattformen auf das Unternehmen erklärt. Im Anschluss wird das Produkt MS SharePoint eingeordnet.

Das dritte Kapitel befasst sich mit den Möglichkeiten zur Implementierung von Geschäftsprozessen und Integration von Drittsystemen. Nach der Darlegung von Gründen, warum SharePoint als Applikationsplattform genutzt werden kann, werden die Möglichkeiten der Lösungsbereitstellung erläutert. Danach werden verschiedene Technologien beschrieben. Hier liegt der Fokus auf der Darstellung von Anwendungsfällen für die jeweilige Technologie sowie die erforderlichen Schritte zur Implementierung, welche als Entscheidungsgrundlage für zukünftige Projekte dienen.

Das abschließende vierte Kapitel bewertet die Ergebnisse der Arbeit.

## 1.5 Über Rowa Automatisierungssysteme GmbH

Die Rowa Automatisierungssysteme GmbH ist Marktführer für automatische Warenlager in Apotheken. Unter dem Leitsatz „Rowa automatisch besser“ werden im Stammsitz Kelberg in der Eifel innovative automatische Warenlager zur Verbesserung des Warenflusses in Apotheken hergestellt. Bereits jetzt haben sich über 3250 Apotheken für ein Automatisierungssystem von Rowa entschieden. Durch die Optimierung der Geschäftsprozesse mit einem Rowa Automatisierungssystem haben Apotheker entsprechend der Philosophie „eine Automatisierung mit dem Menschen im Mittelpunkt“ mehr Zeit zur individuellen Kundenberatung.

Unternehmensfakten:

- Unternehmensgründung 1996
- 314 Mitarbeiter, davon 293 im Stammsitz in Kelberg
- 41 Servicestandorte in Deutschland, Österreich und der Schweiz
- 17 internationale Vertretungen durch Partner mit insgesamt etwa 200 Mitarbeitern
- über 3250 automatisierte Apotheken in 23 Ländern
- Durchschnittlich 46,3 Neukunden pro Monat

Rowa Automatisierungssysteme GmbH

Rowastraße

D-53539 Kelberg

Telefon +49 2692 - 92 06 0

Telefax +49 2692 - 92 06 1299

info@rowa.de

www.rowa.de

## 2 Grundlagen

### 2.1 Was sind Collaborationsplattformen

Collaboration beschreibt einen Leistungsbereich von Software zur Unterstützung der zielgerichteten Zusammenarbeit von Mitarbeitern<sup>5</sup>. Es ist die Gesamtheit aller Informations- und Kommunikationssysteme, die zur Steigerung der Produktivität in der Teamarbeit geeignet sind<sup>6</sup>. Durch Collaborationssysteme soll die Prozesstransparenz gesteigert und der Informationsfluss verbessert werden. Galuschka [2009] beschreibt Collaboration als Sammelbegriff für alle Methoden und Konzepte des modernen Informationsmanagements innerhalb von Unternehmen. Dabei ist nicht nur die Verbesserung der Kommunikation zwischen Mitarbeitern und Teams möglich, sondern auch die Kooperation mit Partnern. Klassische Groupware Lösungen fallen zwar bereits unter den Begriff Collaboration, allerdings umfasst dieser weit mehr. Collaborationsplattformen sind Groupware Lösungen mit neuer Technologie, Struktur und Funktionen des Web 2.0 sowie des Dokumenten- und Content Management. Aufgrund unterschiedlicher Beschreibungen und des Beginns in der Infrastruktur ist die Eingrenzung des Begriffs schwierig. Ziele von Collaborationsplattformen sind die Verteilung von Informationen, Wissensaustausch sowie die Verbesserung der Kommunikation und Zusammenarbeit. Collaboration unterstützt Kommunikation, Koordination und Kooperation (3K). Es lassen sich zwei Arten von Collaboration unterscheiden<sup>78</sup>.

**Asynchrone Collaboration** beschreibt die versetzte Zusammenarbeit, bei der die Prozessschritte chronologisch bearbeitet werden bzw. die Kommunikation nicht in Echtzeit geschieht. Zu der asynchronen Collaboration gehören Dienste wie

- E-Mail
- Diskussionsforen
- Wiki
- Blog
- Bewertungen und Kommentare

---

<sup>5</sup>Angermeier [2010]

<sup>6</sup>Namesnik [2007]

<sup>7</sup>Galuschka [2009]

<sup>8</sup>Comundus

- gemeinsame Ressourcenplanung (etwa Gruppenkalender)
- Team- und Projekträume (online, nicht physische Räume)
- Dokumenten- und Inhaltsverwaltung
- Instant Messaging und Chat
- soziale Netzwerke

**Synchrone Collaboration** beschreibt Echtzeitanwendungen wie Telefon- und Videokonferenzen. Hierunter fällt auch das zeitgleiche, gemeinsame Bearbeiten von Dokumenten.

## 2.2 Collaboration im Unternehmen

Das Thema Collaboration ist in den Unternehmen angekommen und genießt einen hohen Stellenwert. So zeigt eine Studie von Kelton Research <sup>9</sup>, dass für 53% der befragten Manager Collaboration wichtiger ist als die Spezialisierung von Mitarbeitern. Neben den bisher am häufigsten verwendeten klassischen Tools wie E-Mail, Telefon und freigegebene Ordner gesellen sich neue Technologien. Im Jahr 2006 waren bereits 58% aller Berufstätigen auf einen Computer angewiesen. In technologiebasierten Bereichen nutzen fast 100% aller Arbeitnehmer den Computer für ihre tägliche Arbeit <sup>10</sup>, so dass einer weiteren Bedeutungszunahme technisch nichts im Wege steht.

### 2.2.1 Bedeutung von Collaboration

Die Hauptgründe für die Investition in Collaboration sind gemäß der Studie das Einsparen von Kosten und Zeit sowie die Produktivitätssteigerung. Neue Arbeitsformen wie Telearbeit oder abteilungs- und unternehmensübergreifende Projektteams gehören heute zum Arbeitsalltag in vielen Unternehmen <sup>11</sup>. Die größten Zeitkiller sind E-Mail Kommunikation und Meetings. Bisherige Werkzeuge wie Outlook werden den Anforderungen nicht mehr gerecht. Durch neue Werkzeuge zur Collaboration soll dieser Aufwand reduziert werden. Wichtige Anforderungen

---

<sup>9</sup>Dreyßig [2010]

<sup>10</sup>Paulke u. a. [2008]

<sup>11</sup>SaaS-Forum [2010]

an Collaborationslösungen sind die Unterstützung der Kommunikation, Koordination und Kooperation <sup>12</sup> (Abbildung 2.1). Aus der Studie lassen sich einige Trends

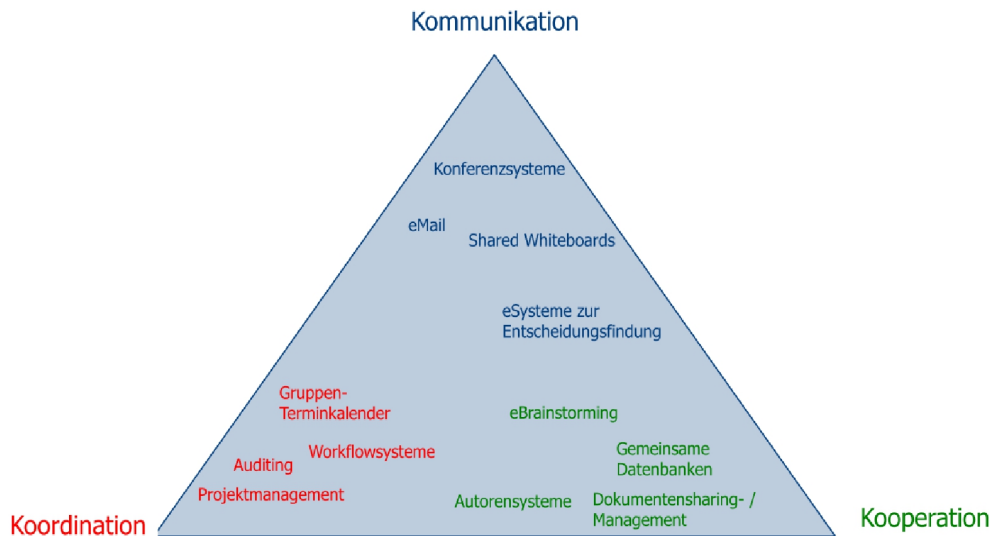


Abbildung 2.1: Collaborationswerkzeuge und Anforderungen (Paulke u. a. [2008], S.15)

ableiten, welche die Bedeutung von Collaboration in Zukunft weiter steigern werden. Zum einen findet ein Wandel in der Organisation der Unternehmen, weg von hierarchischen Strukturen, hin zu flachen Strukturen statt. Die Selbstorganisation der Mitarbeiter nimmt hierbei eine zentrale Rolle ein. Zum anderen wandeln sich Unternehmen von geschlossenen Ökosystemen zu offenen Plattformen, bei der selbst der Verbraucher in die Prozesse des Unternehmens einbezogen wird.

## 2.2.2 Web 2.0 im Unternehmen

Bereits heute nutzen viele Unternehmen Intranet Lösungen, um ihre Mitarbeiter mit Informationen zu versorgen. Der Mitarbeiter ist hier aber nur Informationskonsument und trägt selbst nicht zur Verbesserung der Informationen bei <sup>13</sup>. Durch den Umstieg auf Web 2.0 Technologien erhalten Mitarbeiter die Möglichkeit, selbst Informationen bereitzustellen und die Qualität vorhandener Informationen durch aktive Teilhabe zu verbessern. Tim O'Reilly <sup>14</sup> versteht unter dem Begriff Web 2.0 ein verändertes Nutzerverhalten, bei dem Nutzer durch neue Internetanwendungen Inhalte aktiv selbst erstellen können. Der wesentliche Aspekt dabei ist das Zusammenwirken mehrerer Personen, die den Inhalt weiterentwickeln sowie die Tatsache, dass der Benutzer keine spezielle Software benötigt.

<sup>12</sup>Paulke u. a. [2008]

<sup>13</sup>Rohles [2008]

<sup>14</sup>O'Reilly [2005]

Durch die Umsetzung des Web 2.0 Gedanken im Unternehmen entsteht das Enterprise 2.0. Dabei ist der Einsatz von Wiki Systemen am weitesten verbreitet<sup>15</sup>.

### 2.2.3 Auswirkungen auf das Unternehmen

Collaboration ist weniger ein Technologiethema, sondern betrifft vielmehr die Kultur und Arbeitsweise des Unternehmens. Für die erfolgreiche Umsetzung bedarf es einen Wandel zu einer partizipativen Unternehmenskultur. Die Nutzung der neuen Systeme ist oft eine Generationenfrage. Gerade junge Mitarbeiter nutzen vergleichbare Systeme privat und erwarten diese auch im Unternehmen<sup>16</sup>. Durch die unterstützte direkte Kommunikation werden verschiedene Abteilungen und Hierarchiestufen eng miteinander verknüpft. Mitarbeiter und Vorgesetzte müssen die bisherigen Informationswege verlassen und die Möglichkeiten der direkten Kommunikation nutzen. Die aktive Mitwirkung von Führungskräften bei der Einführung ist für den Erfolg entscheidend<sup>17</sup>. Der Wert von Collaboration Lösungen liegt in den durch Mitarbeiter zusammengetragenen Informationen. Um die Mitarbeiter zu motivieren, ihr Fachwissen zu teilen, müssen Vorgesetzte dies aktiv vorleben. Werden Informationen überwiegend redaktionell zusammengetragen oder Rückmeldungen von Mitarbeitern zu den Informationen nicht bzw. negativ beachtet, wird die Akzeptanz des Systems schnell sinken<sup>18</sup>. Wie jede Organisationsänderung wird auch die Einführung von Collaboration Werkzeugen oft von Bedenken und Hindernissen begleitet. Die Sicherheit der Systeme und Daten muss besonders beachtet werden, damit keine Betriebsinterna nach außen geraten. Sensible Daten dürfen trotz Collaboration nur einem eingeschränkten Mitarbeiterkreis zur Verfügung stehen. Durch die vielen zusätzlichen Kommunikationskanäle können Mitarbeiter überfordert werden und sich in der Informationsflut verlieren. Zudem muss berücksichtigt werden, dass verschiedene Abteilungen oft einen unterschiedlichen Kommunikationsbedarf haben<sup>19</sup>. Besonders ältere oder wenig technikaffine Mitarbeiter können durch die neuen Werkzeuge abgeschreckt oder ausgegrenzt werden. Dem muss durch eine einfache Bedienung, die einen schnellen Einstieg ermöglicht, begegnet werden. Die bisher vorherrschende sozialhierarchische Ebene kann zum Hindernis werden, wenn Mitarbeiter davor zurückschrecken Kollegen in anderen Bereichen oder Positionen zu kontaktieren<sup>20</sup>.

---

<sup>15</sup>Niemeier [2010]

<sup>16</sup>DreyBig [2010]

<sup>17</sup>Niemeier [2010]

<sup>18</sup>Paulke u. a. [2008]

<sup>19</sup>Bauer u. Rief [2010]

<sup>20</sup>Bube [2010]

Vorgesetzte sind gefordert, den kulturellen Rahmen für die direkte Kommunikation zu schaffen<sup>21</sup>. Andererseits wird von vielen Vorgesetzten befürchtet, dass Mitarbeiter ihre eigenen Verantwortlichkeiten schleichend abgeben, wenn Kollegen schnell und einfach mittels Collaboration Werkzeugen erreichbar sind, und Verantwortungen unklar geregelt sind.

## 2.3 Was ist SharePoint 2010

Microsoft beschreibt SharePoint als die Business Plattform für Zusammenarbeit im Unternehmen und im Web<sup>22</sup>. Es erleichtert die Zusammenarbeit von Projektteams innerhalb und außerhalb des Unternehmens. Sowohl Intranet, Extranet und der Internetauftritt lassen sich mit SharePoint realisieren. Zudem ermöglicht es eine dokumenten- und informationszentrierte Arbeit und die Automatisierung von Geschäftsprozessen.

SharePoint soll die Aggregation von Informationen, Zusammenarbeit und Suche an einem Punkt bündeln. MS teilt SharePoint, wie in Abbildung 2.2 dargestellt, in sechs Funktionsbereiche ein<sup>23</sup>. Der tatsächliche Funktionsumfang hängt von der



Abbildung 2.2: Funktionsbereiche von SharePoint 2010 (Microsoft [2010d])

Version des SharePoint ab. Die Unterschiede werden weiter unten dargestellt.

- „Sites“ bilden die Grundlage des SharePoint. Der Bereich beinhaltet alle Tools, die Infrastruktur und das Seitenmanagement. Funktionen wie Mehr-

---

<sup>21</sup>Rohles [2008]

<sup>22</sup>Microsoft [2010c]

<sup>23</sup>Microsoft [2010d]



sprachigkeit, breiter Browsersupport und die Zielgruppenadressierung fallen ebenfalls in diesen Bereich.

- „Communities“ beinhaltet alle Enterprise 2.0 Funktionen wie Profilseiten von Mitarbeitern, Wiki, Teamseiten und Tagging. Auch das zeitgleiche Bearbeiten von Dokumenten und die Offline Möglichkeiten zählen hierzu.
- „Content“ deckt den Bereich Content Management ab. Es umfasst die Versionsverwaltung von Inhalten, Metadatenverwaltung, Workflows und Taxonomie.
- „Search“ beinhaltet die schnelle und einfache Suche nach Wissen, Dokumenten, Personen und Daten. Ergebnisse lassen sich durch Berücksichtigung von Ratings und der Metadatenavigation verfeinern. Auch eine phonetische Suche ist möglich.
- „Insights“ umfasst den Bereich Business Intelligence (BI). Daten aus verschiedenen Quellen lassen sich aufbereitet darstellen und durchsuchen.
- „Composites“ beschreibt funktionale, kombinierbare Bausteine, mit denen sich Lösungen ohne Code umsetzen lassen, wie etwa SharePoint Designer, InfoPath Formularservice oder Sandbox Solutions.

Die einzelnen Funktionsbereiche greifen ineinander und lassen sich nicht klar abgrenzen. Grundsätzlich lassen sich zwei Versionen von SharePoint unterscheiden, die SharePoint Foundation 2010 und SharePoint Server 2010 (Abbildung 2.3).

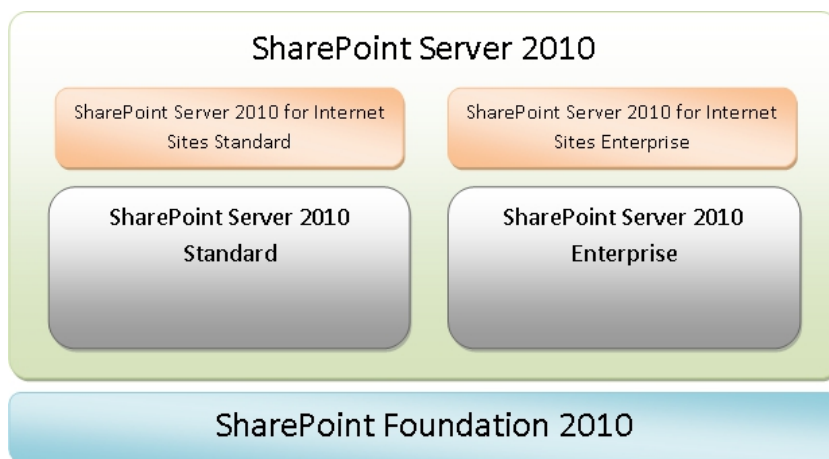


Abbildung 2.3: SharePoint 2010 Versionsübersicht (eigene Darstellung)

Die SharePoint Foundation 2010 bildet die Basis für SharePoint Server 2010 und bedient, mit Ausnahme von Insights, jeden oben dargestellten Funktionsbereich. Das Produkt bedarf keinerlei zusätzlichen Lizenzierung, da SharePoint Foundation 2010 bereits in der Windows Server Lizenz enthalten ist. Während die Bereiche Sites und Composites nahezu den vollständigen Funktionsumfang bieten, gibt es deutliche Einschränkungen in den Bereichen Communities, Contents, Insights und Search. Die SharePoint Foundation 2010 eignet sich für einzelne Teams oder kleine Unternehmen. Der SharePoint Server 2010 lässt sich in zwei Varianten unterteilen, dem SharePoint Server 2010 Standard und SharePoint Server 2010 Enterprise. Der Unterschied liegt im Funktionsumfang und in der Lizenzierung. Für beide Varianten ist eine Lizenz erforderlich. Der SharePoint Server 2010 Standard bietet gegenüber der SharePoint Foundation 2010 deutliche Verbesserungen wie etwa Zielgruppenadressierung, verbesserte Suche durch Metadatenavigation, bereichsübergreifende Suche und phonetische Suche. Der Funktionsbereich Communities wird unter anderem durch das Enterprise Wiki, MySite und Newsfeeds erweitert. SharePoint Server 2010 Enterprise stellt eine Funktionserweiterung zu SharePoint Server 2010 Standard dar. Neben einzelnen Verbesserungen im Bereich Search liegt der Fokus klar auf den Bereichen Insights und Composites. Nur in dieser Version sind Funktionen wie Access Services (Veröffentlichen von Access Datenbanken und Formularen), Excel Services (Excel Dokumente in einem WebPart betrachten) oder InfoPath Services (InfoPath Formulare im Browser nutzen) verfügbar. Zudem bietet SharePoint Server 2010 Enterprise zahlreiche WebParts zur Anzeige von Geschäftsdaten an. Ein vollständiger Vergleich der einzelnen SharePoint Versionen und Varianten findet sich bei Microsoft [2010b]. Die hier aufgeführten Funktionen werden von MS als besonders erwähnenswert empfunden. Dies bedeutet aber nicht, dass beispielsweise die SharePoint Foundation 2010 zur Inhaltsverwaltung ungeeignet ist.

Die Abbildung 2.4 stellt einen quantitativen Vergleich der Funktionen dar. Hier lässt sich erkennen, welche Bereiche durch welche SharePoint abgedeckt werden. Um die hier dargestellten Versionen und Funktionen in einem Extra- oder Internetauftritt nutzen zu können, sind die SharePoint Server 2010 for Internet Sites Standard bzw. die SharePoint Server 2010 for Internet Sites Enterprise erforderlich. Da es sich hierbei um keine eigenständige Versionen, sondern um reine Lizenzierungsmodelle handelt, entfällt eine umfassende Beschreibung.

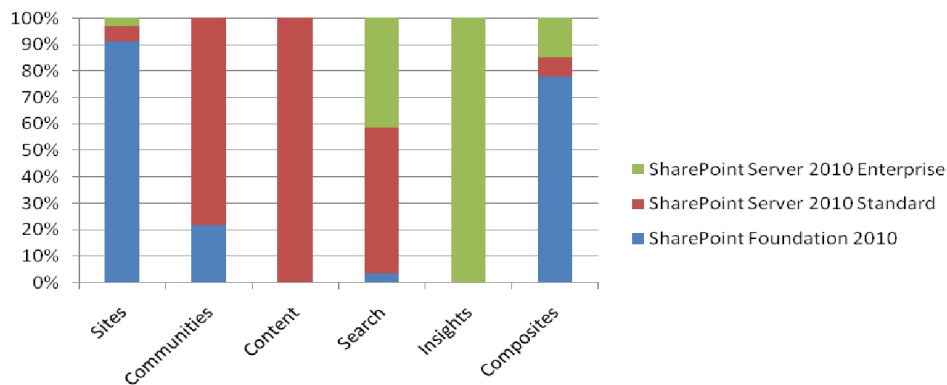


Abbildung 2.4: Quantitativer Vergleich des Funktionsumfangs der SharePoint Versionen (eigene Darstellung)

## 2.4 Historie von SharePoint

Jeff Teper beschreibt in seinem Artikel <sup>24</sup> die lange Geschichte von SharePoint. Ende der 90er Jahre wurde eine Reihe von kleinen Tools bei MS genutzt, die zusammen ein Portal bildeten und die internen Entwicklungsteams koordinierten. Mit diesen ersten Erfahrungen begann die Entwicklung von SharePoint Team Services und dem SharePoint Portal Server, welche 2001 veröffentlicht wurden. Der Fokus bei der Entwicklung lag auf dem Dokumentenmanagement und der Suche. Zur Speicherung von Inhalten wurde das Web Storage System des MS Exchange Server verwendet, welches von MS aber nicht mehr weiterentwickelt wurde. Nach Danny Uytgeerts <sup>25</sup> lag das größte Problem dieser Version in der Art, wie Inhalte und Konfiguration gespeichert wurden. Die Inhalte wurden auf dem Dateisystem, die Metadaten in einer SQL Datenbank abgelegt. Die Konfiguration von SharePoint wurde in der Registry des Web Servers abgelegt. Da der Inhalt, die Metadaten und die Konfiguration auf dem WebServer abgelegt werden mussten, ließ sich diese Version von SharePoint schlecht skalieren. Das MS Digital Dashboard, welches in den SharePoint Server 2001 aufgenommen wurde, ermöglichte bereits die Entwicklung von einer Art WebPart. Der Entwicklungsprozess wurde von den gängigen Werkzeugen aber nicht unterstützt. Durch den Zukauf des MS Content Management Server positionierte MS zudem ein Konkurrenzprodukt zu SharePoint.

<sup>24</sup>Teper [2009]

<sup>25</sup>Uytgeerts [2009]

Mit der nachfolgenden Version änderte MS seine Philosophie von getrennten Produkten. Es entstanden die Windows SharePoint Services 2.0 (WSS 2.0), welche die Basis für den SharePoint Portal Server 2003 bildeten. Die WSS 2.0 waren die erste Version, die in einer Windows Server Lizenz schon enthalten waren. Das bisher verwendete Web Storage System wurde durch den MS SQL Server ersetzt, wodurch die Skalierbarkeit verbessert wurde. Dieses wurde aber aus Kompatibilitätsgründen weiter unterstützt. Durch die Umstellung auf ASP.NET konnten Benutzer WebParts entwickeln und diese auf verschiedenen WebPart Zonen positionieren. Mit der Version SharePoint Portal Server 2003 wurde erstmals das Konzept der MySite implementiert. Eine große Einschränkung stellte die Suche dar. Sie lieferte zwar dank Volltextsuche des SQL Servers gute Ergebnisse, zeigte aber nur Ergebnisse der Seite an, auf der sich der Benutzer gerade befand. Zudem fehlten laut Uytgeerts [2009] eingebettete Workflows. Die inkonsistente Navigation und das Fehlen von Content Type erschwerte die Bedienung. Die Administration dieser und der vorherigen Version erfolgte direkt über die Internet Information Services (IIS) oder die Registry.

Mit den darauf folgenden Versionen, den WSS 3.0 und dem Microsoft Office SharePoint Server 2007 (MOSS 2007), hat MS den Funktionsumfang stark erweitert. Dabei standen das Content Management, die Suche und die Business Intelligence Funktionalitäten im Zentrum der Entwicklung. Aber auch die bekannten Konzepte des Web 2.0 wurden integriert. Durch die volle Unterstützung des .NET Framework ist erstmals die MS Workflow Foundation verfügbar. Die Möglichkeiten der Administration wurden durch die Central Administration und dem Kommandozeilenprogramm STSADM deutlich verbessert. Nach wie vor problematisch ist die schlechte Browser Unterstützung.

Joining Dots stellt in seinem Diagramm (Abbildung A.2) anschaulich die Geschichte von SharePoint bis zum Jahr 2007 dar <sup>26</sup>. Für die aktuelle Version von SharePoint wurden die Konzepte des Enterprise 2.0 konsequent weiterentwickelt. Der Funktionsumfang hat sich nicht wesentlich erweitert. Deutlich vorangeschritten ist allerdings die Reife der Konzepte. Laut Ovcaak [2010] gleicht die Weiterentwicklung eher einer Evolution als einer Revolution. MS fasst im SharePoint Software Development Kit (SDK) <sup>27</sup> die aktuellen Versionen und wesentlichen Funktionen zusammen (Anhang A.1).

---

<sup>26</sup>Joining Dots [2006]

<sup>27</sup>MSDN [2010g]

### 3 SharePoint als Applikationsplattform

SharePoint ist kein monolithisches Stück Software, sondern zeichnet sich durch eine hohe Modularität aus. Es nutzt bekannte Technologien wie etwa ASP.NET, IIS oder den SQL Server. Dank des umfangreichen Objektmodells kann ein Entwickler SharePoint nahezu überall erweitern und anpassen. Das System kann durchaus als Framework betrachtet werden, aus dem man sich, wie aus einem Baukasten, bedienen kann <sup>28</sup>.

Die im Folgenden aufgezeigten Vor- und Nachteile sowie Einsatzmöglichkeiten beziehen sich auf die Firma Rowa Automatisierungssysteme GmbH. Das Unternehmen setzt seit einigen Jahren MS Office SharePoint Server 2007 ein, welcher von den Endbenutzern und Entwicklern allerdings nur als reines Portal und nur bedingt als Integrationsplattform angesehen wird. Im Vorfeld dieser Arbeit wurde untersucht, welche Versuche bislang unternommen wurden, Geschäftsprozesse mit SharePoint umzusetzen, wie erfolgreich diese umgesetzt werden konnten oder warum diese gescheitert sind. Zu den wichtigsten IT Systemen, die in SharePoint integriert werden sollen, gehören

- MS Dynamics CRM 4: Customer Relationship Management System als Vertriebs- und Servicesystem
- Sage Office Line: Enterprise Ressource Planning (ERP) System zur Produktionsplanung und -kontrolle sowie Finanzbuchhaltung
- Rowa Planer: eigen entwickeltes Planungswerkzeug zur Projekt- und Ressourcenverwaltung
- Zeiterfassungssystem
- Data Warehouse System: Controllinginstrument auf Basis einer MS SQL Datenbank und den MS SQL Analysis Services
- MS BizTalk Server: Integrations- und Kommunikationsplattform für IT Systeme zur Bildung von technischen Schnittstellen

Diese Erkenntnisse wurden anschließend auf einem SharePoint Foundation 2010 und einem SharePoint Server 2010 Enterprise Testsystem angewendet und überprüft. Mit den hier gewonnenen Erkenntnissen wurden einige Anwendungsfälle für die ausgewählten Technologien ermittelt.

---

<sup>28</sup>Steinke [2010]

## 3.1 Warum SharePoint als Applikationsplattform

### 3.1.1 Gründe

Mit ASP.NET lassen sich zahlreiche Anforderungen an eine webbasierte Applikationsplattform bedienen. SharePoint basiert auf ASP.NET und erweitert die Möglichkeiten. Es kann als Alternative zu ASP.NET angesehen werden <sup>29</sup>.

### 3.1.2 Vorteile

Zahlreichen Funktionen, die bereits in der SharePoint Foundation 2010 geboten werden, sind vorhanden. Besonders Infrastrukturanforderungen, wie etwa ein Sicherheitsmanagement mit Benutzer- und Rechteverwaltung oder die Anbindung an Domain Controller und Verzeichnisdiensten, können sofort genutzt werden <sup>30</sup>. SharePoint bietet eine hohe Skalierbarkeit. In der Minimalkonfiguration können alle Komponenten auf einem Server installiert werden. Mit wachsenden Strukturen und Anforderungen können einzelne Komponenten auf andere Server ausgelagert werden. Der Einsatz von mehreren IIS als Web Frontend Server zur Lastverteilung lässt sich leicht realisieren.

Drittsysteme können mittels Web Services und Representational State Transfer Architektur (REST) Daten aus SharePoint auslesen. Dank des Client Objekt Modell können Drittsysteme auch direkt mit SharePoint Objekten arbeiten. Mit Hilfe von Event Receivern kann auf viele Ereignisse im Zusammenhang mit den wichtigsten SharePoint Objekten, wie etwa Webseitensammlungen, Webseiten, Listen und Elementen, reagiert werden <sup>31</sup>. Zur Entwicklung neuer Lösungen stehen zwei wichtige Werkzeuge zur Verfügung. Mit dem kostenlos erhältlichen SharePoint Designer lassen sich viele Anforderungen wie Workflows oder die Anbindung externer Systeme schnell realisieren. Mit Visual Studio 2010 können alle Entwicklungsmöglichkeiten genutzt werden <sup>32</sup>.

Zur Administration steht eine eigene Oberfläche, die Central Administration (CA), zur Verfügung. Da es sich hierbei auch um eine SharePoint Instanz handelt, kann auch die CA erweitert werden <sup>33</sup>. Es gibt eine Vielzahl von administrativen Seiten, welche nur von einem Administrator oder berechtigten Personen aufgerufen werden können. Auch diese Seiten können angepasst bzw. neue administrative

---

<sup>29</sup>Krause [2010]

<sup>30</sup>Krause [2010]

<sup>31</sup>Krause [2010]

<sup>32</sup>Steinke [2010]

<sup>33</sup>Krause [2010]

Seiten erstellt werden. Zur Überwachung der Seitenaufrufe, Suchanfragen usw. können die mitgelieferten Berichte und Statistiken genutzt werden.

Die Vorteile aus der Benutzersicht sind die vorhandenen Web 2.0 Funktionen und die Suchfunktionen. Ein Endbenutzer kann bei entsprechender Berechtigung Objekte nach seinen Wünschen anpassen und so den Entwickler entlasten <sup>34</sup>. SharePoint ist nicht auf die Nutzung bestimmter Browser angewiesen, sondern unterstützt die meisten aktuellen Browser. Es können mehrere Sprachen verwendet werden. Die Mehrsprachigkeit ist auch im Nachhinein einsetzbar <sup>35</sup>.

#### 3.1.3 Nachteile

Obwohl für den Einsatz der SharePoint Foundation 2010 keine Lizenz benötigt wird, ist die Nutzung von SharePoint nicht kostenlos, da pro verwendeter Server eine Windows Server 2008 Lizenz und mindestens eine SQL Server Lizenz notwendig sind. Für den Einsatz der SharePoint Server Versionen ist zusätzlich eine Office 2010 Lizenz für den Server nötig. SharePoint gibt es nur als 64bit Version <sup>36</sup>. Der volle Funktionsumfang von SharePoint kann nur in Verbindung mit Office 2010 genutzt werden <sup>37</sup>. Bei der Entwicklung von Lösungen ist darauf zu achten, dass nur das .NET Framework 3.5 unterstützt wird. Dies bedeutet auch, dass ASP.NET 2.0 verwendet wird. Die neuen Funktionen des .NET Frameworks 4.0 und von ASP.NET 4.0 stehen nicht zur Verfügung <sup>38</sup>.

Darüber hinaus treten folgende Nachteile bei der praktischen Arbeit auf. Während der Entwicklungsphase müssen Lösungen oft getestet und der Quellcode mittels Debugger untersucht werden. Dazu muss für jeden Test die Lösung bereitgestellt werden, was je nach Bereitstellungsart zu einem Neustart des jeweiligen Application Pool führt. Dies fordert vom Entwicklungsteam viel Geduld.

Zudem geht die Entscheidung für SharePoint mit einem erheblichen Schulungsaufwand einher. Die Oberfläche ist oftmals selbsterklärend. Doch durch die vielen Funktionen müssen Endbenutzer intensiv geschult werden, damit sie alle Möglichkeiten kennen und nutzen. Auch die Verwendung der Ribbon UI ist für den Endanwender oft eine Umstellung. Ein Entwickler muss ebenfalls intensiv geschult werden, selbst wenn bereits Kenntnisse in ASP.NET, eXtensible Markup Language (XML) und Master Pages vorhanden sind. Es werden viele bereits bekannte

---

<sup>34</sup>Moritz [2010]

<sup>35</sup>Hey [2010]

<sup>36</sup>Krause [2010]

<sup>37</sup>Heck [2010]

<sup>38</sup>Krause [2010]

Begriffe verwendet. Allerdings haben sie hier eine andere Bedeutung. Außerdem wird mit SharePoint eine komplexe Architektur mit zahlreichen Erweiterungsmöglichkeiten eingeführt, deren Kenntnis unabdingbar ist.

#### **3.1.4 Wofür ist es geeignet**

Zu den klassischen Anwendungsfällen, in denen SharePoint als Applikationsplattform eingesetzt werden, gehören Programmfunktionen für Mitarbeiter (Urlaub beantragen, Büromaterial bestellen, Buchung von Ressourcen wie Räume oder Equipment), die in anderen Applikationen des Unternehmens, etwa einem ERP-System, nicht eingebaut werden können. Die Implementierung, etwa in einem ERP-System, ist damit verbunden, dass wesentlich mehr Personen (z.B. Produktionsmitarbeiter) Zugriff auf das System benötigen. Solche Anforderungen lassen sich in einem Unternehmensportal wesentlich besser integrieren. Kunden, Partner und Lieferanten wird ebenfalls oft ein Portal zur Kommunikation mit dem Unternehmen zur Verfügung gestellt. Durch die Erweiterung des Portals mit Anwendungen entsteht ein Mehrwert für die Partner.

In vielen Unternehmen werden Data Warehouse Systeme zur Aggregation von Unternehmensdaten eingesetzt, auf die nur wenige Personen Zugriff haben (z.B. Controlling Mitarbeiter). Die Informationen eines Data Warehouse Systems werden oft von vielen Mitarbeitern, manchmal von allen, benötigt. SharePoint bietet sich hier als Benutzeroberfläche an, um Daten zielgruppengerecht darzustellen. Werden Informationen aus anderen Applikationen, wie ERP-Systemen, von Mitarbeitern benötigt, die keinen Zugriff auf das ERP-System haben, kann SharePoint als Benutzeroberfläche zum Einsatz kommen. Ein schreibender Zugriff auf solche Systeme lässt sich durch Web Service Aufrufe realisieren. Dies alles reduziert die Anzahl der verschiedenen Anwendungen für den Benutzer.

#### **3.1.5 Wofür ist es nicht geeignet**

Mit SharePoint kann eine alternative Benutzeroberfläche für Drittsysteme erstellt werden. Dies sollte aber nicht angestrebt werden, da der Entwicklungsaufwand für Schnittstellen und Oberfläche zu groß wäre. Die Interaktion mit Drittsystemen sollte, mit wenigen Ausnahmen, nur lesend realisiert werden. Bietet ein Drittsystem Schnittstellen wie Web Services an um Daten zu manipulieren, können diese leicht in SharePoint integriert werden. Von einem direkten schreibenden Zugriff in Datenbanken ist abzusehen, da die gesamte Geschäftslogik in SharePoint nachgebildet werden müsste.



SharePoint bietet mit der MS Workflow Foundation eine vollständige Workflow Umgebung an. Hiermit lassen sich Workflows in Drittsystemen aber kaum abbilden. Obwohl diese in einem Workflow eingebunden werden können, kann eine vollständige Geschäftslogik, welche überwiegend auf dem Drittsystem basiert, kaum realisiert werden.

SharePoint ist eng mit Office 2010 verbunden. Wird in einem Unternehmen überwiegend Software anderer Hersteller, etwa Open Office, Lotus Notes oder CAD Systeme eingesetzt, muss geprüft werden, ob andere Portallösungen nicht besser geeignet sind <sup>39</sup>.

---

<sup>39</sup>Jung [2010]

## 3.2 Möglichkeiten der Lösungsbereitstellung

Es gibt zahlreiche Möglichkeiten, SharePoint an die Bedürfnisse des Unternehmens anzupassen und mit selbst entwickelten Funktionen zu erweitern. Im Folgenden werden Aspekte der Lösungsbereitstellung behandelt, die für Lösungen zutreffen, welche mit Visual Studio 2010 erstellt wurden. Die Bereitstellung der Anpassungen erfolgt als SharePoint Solution Package (Dateiendung WSP). Ein solches Lösungspaket ist eine Cabinet (cab) Datei, in der alle Programm- und Konfigurationsdateien einer Lösung enthalten sind. Ein Paket wird automatisch in Visual Studio 2010 erstellt, wenn ein SharePoint Projekt als Projektvorlage verwendet wird <sup>40</sup>. Es gibt zwei Arten von SharePoint Lösungen, die Farm Lösung und die Sandbox Lösung.

### 3.2.1 Grundlagen

SharePoint verfügt über ein umfassendes Objektmodell. Zum besseren Verständnis der folgenden Kapitel werden zunächst die wichtigsten Ebenen der Architektur und deren Bedeutung grob umrissen <sup>41</sup> (Abbildung 3.1).

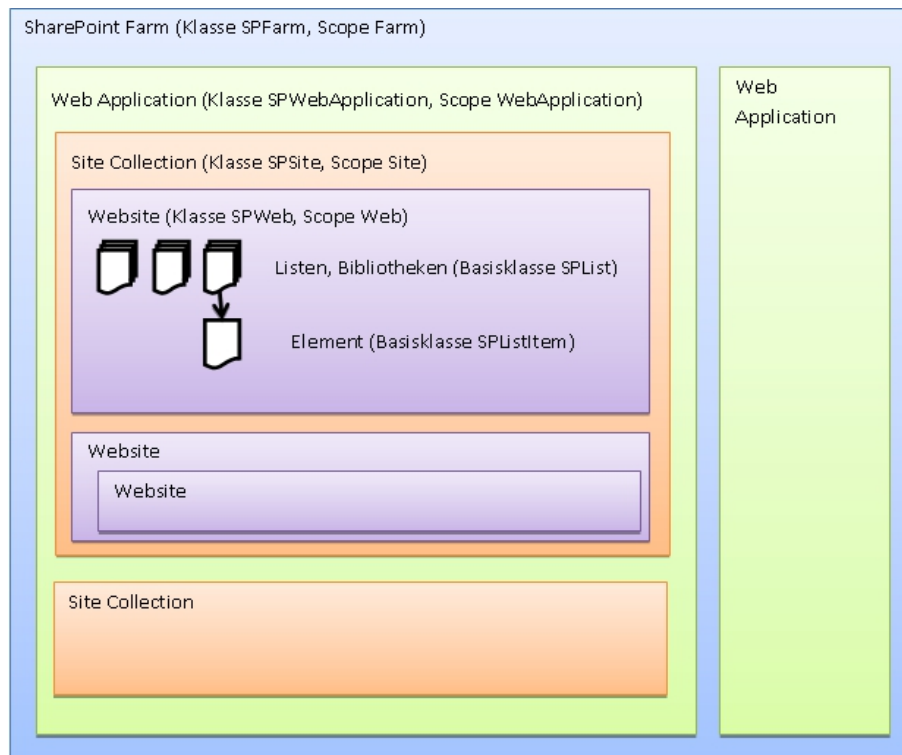


Abbildung 3.1: Hierarchieebenen in SharePoint (eigene Darstellung)

<sup>40</sup>MSDN [2011c]

<sup>41</sup>Krause u. a. [2010a]

- **Farm:** Die SharePoint Farm ist die oberste Ebene. Sie umfasst alle globalen Einstellungen, Server, Dienste und Daten einer SharePoint Installation.
- **Web Application:** Eine Web Application entspricht einer Website des IIS und dem dazugehörigen Anwendungspool. Eine Farm kann aus mehreren Web Applications bestehen. Jede Web Application hat ihre eigene URL und kann eigene Sicherheitseinstellungen, wie etwa die Art der Authentifizierung enthalten.
- **Site Collection:** Eine Site Collection ist eine Sammlung von logisch zusammengehörenden Websites. In jeder Site Collection gibt es mindestens eine Website, die Root Website. Eine Web Application kann mehrere Site Collections beinhalten. Jede Site Collection hat eine eigene Content Datenbank.
- **Website:** Eine Website ist eine Sammlung von Listen und Dokumentenbibliotheken mit deren Inhalt. In der Bibliothek Seiten (Pages) befinden sich alle Webpart- und sonstige Seiten der Website. Jede Website kann eine beliebige Anzahl an Listen und Bibliotheken aufnehmen. Websites können auch andere Websites beinhalten (Sub-Website). Auf dieser Ebene werden auch die Benutzerberechtigungen vergeben.

### 3.2.2 Farm Lösung

Eine Farm Lösung hat umfassenden Zugriff auf alle Ressourcen und Funktionen von SharePoint. Die Assemblies werden entweder im Global Assembly Cache (GAC) oder im bin Verzeichnis der Web Application angelegt. Wird die Lösung im GAC bereitgestellt, gelten keinerlei Sicherheitsbeschränkungen für den Code. Wird die Lösung im bin Verzeichnis der Web Application bereitgestellt, gelten die Code Access Security (CAS) Sicherheitsbeschränkungen, die in der web.config definiert sind. Der Code muss in jeder Web Application einzeln bereitgestellt werden <sup>42</sup>. Um eine solche Lösung bereitstellen zu können, muss der Benutzer über Farm Administrator Rechte verfügen. Nach der Bereitstellung wird automatisch ein Neustart des Webserver durchgeführt. Dies ist erforderlich, damit Änderungen an der web.config und den Assemblies wirksam werden <sup>43</sup>.

Aus diesen Aussagen lässt sich ableiten, dass Farm Lösungen die meisten Entwicklungsmöglichkeiten bieten. Sie können aber die Stabilität einer SharePoint Farm bzw. eines Web Frontend Server Servers gefährden, da die Lösungen im

---

<sup>42</sup>TechNet [2010b]

<sup>43</sup>MSDN [2010a]

Ressourcenverbrauch nicht beschränkt sind und den vollen Funktionsumfang des .NET Frameworks genießen.

### 3.2.3 Sandbox Lösung

Anders als Farm Lösungen unterliegen Sandbox Lösungen umfassenden Restriktionen. Der Code wird in einem eigenen Prozess (SPUCWorkerProcess) ausgeführt und ist von den restlichen Prozessen isoliert. Es stehen nur bestimmte Teile der SharePoint API und des .NET Frameworks zur Verfügung. Auch der Ressourcenverbrauch (CPU Zeit, Arbeitsspeicher, Datenbankzugriffe) ist beschränkt und kann vom Farm Administrator vorgegeben werden. Verletzt eine Sandbox Lösung die Restriktionen oder verbraucht sie zu viele Ressourcen, wird sie automatisch deaktiviert. Sandbox Lösungen werden in einer speziellen Bibliothek, der Solution Gallery gespeichert. Jede Site Collection verfügt über diese Bibliothek. Entsprechend berechnete Benutzer können SharePoint Solution Pakete in die Solution Gallery hochladen und hier verwalten. Es ist kein Neustart des IIS erforderlich. Müssen Bestandteile der Lösung auf dem Dateisystem des Servers abgelegt werden, kann das Projekt nicht als Sandbox Lösung erstellt werden <sup>44</sup>. Eine Sandbox Lösung kann nur auf Objekte unterhalb der aktuellen Site Collection zugreifen. Auf externe Ressourcen, wie andere Site Collections, Dateisystem des Servers, Datenbanken oder Web Services, kann nicht zugegriffen werden <sup>45</sup>. Die Restriktionen einer Sandbox Lösung lassen sich umgehen, indem eine Full Trust Proxy Klasse erstellt wird. Dies ist eine Klasse, die als Farm Lösung bereitgestellt wird und von der Basisklasse SPProxyOperation erbt. Die Proxy Klasse wird von einem eigenen Prozess (SPUCWorkerProcessProxy) ausgeführt <sup>46</sup>. Sandbox Lösungen sind in SharePoint Online Umgebungen oftmals die einzige Möglichkeit, um eigene Funktionalitäten bereit zu stellen.

Aus diesen Aussagen lässt sich ableiten, dass sich Sandbox Lösungen auch bei einer lokalen SharePoint Installation hervorragend eignen, um schnell Lösungen entwickeln und testen zu können. Dies begründet sich damit, dass der Webserver beim Bereitstellungsprozess nicht wie bei einer Farm Lösung neugestartet werden muss. Die Bereitstellung in einem Produktivsystem kann jederzeit erfolgen. Es bedarf keinem Wartungsfenster, da die Benutzer ohne Unterbrechung weiterarbeiten können. Die Effizienz des eigenen Codes lässt sich mit dem Ressourcenmonitor des Sandbox Frameworks überprüfen.

---

<sup>44</sup>MSDN [2011e]

<sup>45</sup>MSDN [2011d]

<sup>46</sup>MSDN [2011e]

Für komplexe Projekte, bei denen Interaktionen mit anderen Softwarekomponenten erforderlich sind, eignen sich Sandbox Lösungen nicht. Zwar lassen sich die Restriktionen mittels Proxy Klassen umgehen, diese müssen aber als Farm Lösung bereitgestellt werden, was in SharePoint Online Umgebungen meist nicht gestattet ist. Falls in einer gehosteten Umgebung die Bereitstellung von Farm Lösungen erlaubt ist, stellen Proxy Klassen Aufgrund des hohen Implementierungsaufwandes keine Alternative zu Farm Lösungen dar.

Bei MSDN [2011d] wird darauf hingewiesen, dass Sandbox Lösungen nicht auf externe Ressourcen, etwa Datenbanken oder Web Services zugreifen können. Nach Prüfung dieser Restriktion kann bestätigt werden, dass ein direkter Zugriff auf eine Datenbank oder ein Web Service nicht gestattet ist, der Zugriff auf eine externe Liste (vgl. Kapitel 3.3.1) hingegen erlaubt ist. Mit den Business Connectivity Services ist ein indirekter Zugriff also möglich.

#### 3.2.4 Features

Features sind Bestandteile eines SharePoint Lösungspaketes. Mit ihnen lassen sich Funktionen in unterschiedlichen Gültigkeitsbereichen einfach aktivieren und deaktivieren. Die Definition eines Features ist in der zugehörigen Feature.xml enthalten. Hier sind alle Basiseigenschaften, wie der Name, der Gültigkeitsbereich, Abhängigkeiten von anderen Features aber auch alle zugehörigen Elementdateien beschrieben <sup>47</sup>. Mittels FeatureReceiver kann auf Ereignisse des Features reagiert werden. Ein FeatureReceiver ist eine Klasse, die von Microsoft.SharePoint.SPFeatureReceiver abgeleitet wird. Die Klasse wird ebenfalls in der Feature.xml referenziert. Durch überschreiben der entsprechenden Methoden kann auf verschiedene Ereignisse reagiert werden <sup>48</sup>.

Features lassen sich per Timer Dienst, Konsole, dem Objektmodell und über die Benutzeroberfläche verwalten <sup>49</sup>. Ist ein FeatureReceiver für ein Feature registriert, ist der ausführende Benutzer Aufgrund der vielen Verwaltungsmöglichkeiten nicht klar. Es sollte daher auf die Prüfung von Benutzerberechtigungen verzichtet werden. Ebenfalls ist nicht immer gewährleistet, dass auf ein SharePoint Context Objekt zugegriffen werden kann. Ein solches Objekt ist nur bei der Aktivierung via Benutzeroberfläche verfügbar <sup>50</sup>.

---

<sup>47</sup>TechNet [2010a]

<sup>48</sup>MSDN [2010e]

<sup>49</sup>TechNet [2010a]

<sup>50</sup>Wheeler [2010]

**Gültigkeitsbereiche von Features** Der Gültigkeitsbereich (Scope) gibt an, auf welcher Ebene in der Architektur von SharePoint die Funktionen verfügbar sind, die in dem Feature definiert wurden. Er beeinflusst auch die Verwaltung des Features sowie die Anzahl der potenziellen Aufrufe der FeatureReceiver <sup>51</sup>. Eine Übersicht der einzelnen Ebenen und der Gültigkeitsbereiche befindet sich in Abbildung 3.1. Der Gültigkeitsbereich eines Features wird auch von Elementen, also den Funktionen, die bereitgestellt werden sollen, bestimmt. Eine Übersicht, welche Elemente in welchem Gültigkeitsbereich zulässig sind, findet sich bei MSDN [2011b].

**Farm Feature (Scope Farm)** Ein Farm Feature wird auf der obersten Hierarchieebene einer SharePoint Installation aktiviert. Die Funktion ist anschließend überall in der Farm verfügbar <sup>52</sup>. Ein Farm Feature kann nur über die Central Administration verwaltet werden.

**Web Application Feature (Scope WebApplication)** In einer SharePoint Farm können mehrere Web Applications enthalten sein. Ein Web Application Feature muss für jede Web Application aktiviert werden, welche die Funktion des Features nutzen soll. Die Funktion steht anschließend jeder Site Collection und Website der Web Application zur Verfügung <sup>53</sup>. Die Verwaltung des Features erfolgt ebenfalls über die Central Administration.

**Site Collection Feature (Scope Site)** Eine Site Collection enthält eine oder mehrere Websites. Ein Site Collection Feature muss in jeder Site Collection aktiviert werden, welche die Funktion des Features nutzen soll. Die Funktion steht anschließend allen Websites der Site Collection zur Verfügung. Die Verwaltung des Features erfolgt über Benutzeroberfläche der Site Collection. Dazu muss der Benutzer über Site Collection Administrationsrechte verfügen.

**Website Feature (Scope Web)** Ein Website Feature wird auf Ebene einer bestimmten Website aktiviert. Die Funktion des Features steht anschließend nur dieser Website zur Verfügung. Wird die Funktion in anderen Websites benötigt, muss das Feature in den betroffenen Websites aktiviert werden. Die Verwaltung des Features erfolgt über die Benutzeroberfläche der Website. Dazu muss der

---

<sup>51</sup>MSDN [2010b]

<sup>52</sup>Krause u. a. [2010b]

<sup>53</sup>Krause u. a. [2010b]

Benutzer mindestens über ManageWeb Berechtigungen verfügen.

Bei der praktischen Arbeit mit Website Features muss eine Besonderheit beachtet werden. Wird innerhalb einer Website ein solches Feature aktiviert und anschließend eine neue Sub-Website angelegt, ist das Feature in dieser neuen Website nicht aktiv. Es erfolgt also keine Vererbung der Feature Einstellungen.

### 3.3 Möglichkeiten der Entwicklung

Im folgenden werden verschiedene Technologien und Entwicklungsmöglichkeiten von SharePoint vorgestellt. Zu jeder Technologie werden im Anschluss an die Beschreibung einige Anwendungsfälle dargestellt.

#### 3.3.1 Business Connectivity Services

Mit den Business Connectivity Service (BCS) kann auf Geschäftsdaten von Drittsystemen lesend und schreibend zugegriffen werden, ohne dass der Benutzer SharePoint verlassen muss. Zur Definition der externen Daten werden External Content Types verwendet <sup>54</sup>.

Ein Content Type beschreibt das Schema von Daten, also die Attribute, Eigenschaften und Datentypen. Ein External Content Type erweitert diese Angaben mit Informationen darüber, wo die Daten herkommen und wie darauf zugegriffen werden kann. Zur Beschreibung eines External Content Type (ECT) wird XML verwendet <sup>55</sup>. Laut Krause u. a. [2010c] bestehen die BCS im Wesentlichen aus folgenden drei Komponenten

- Business Data Catalog (BDC) Metadata Store beinhaltet die Definitionen der BDC Modelle und der darin definierten ECT.
- BDC Server Runtime führt die Kommunikation mit dem Line-of-Business System (LOB System).
- BDC Client Runtime wird auf dem Arbeitsplatzrechner des Benutzers ausgeführt. Mit ihr können Office Anwendungen direkt mit dem LOB System kommunizieren und die Daten mittels Cache auch offline nutzen. Die Client Runtime ist nicht in der SharePoint Foundation 2010 verfügbar.

Darüber hinaus kann der Secure Store Service zusammen mit den BCS genutzt werden. Er dient zur verschlüsselten Speicherung von Anmeldeinformationen und kann in vielen SharePoint Komponenten genutzt werden. Diese Funktion ist in der SharePoint Foundation 2010 nicht verfügbar.

Die Daten des LOB System werden nicht in der Content Datenbank von SharePoint gespeichert. Mit BCS lassen sich Daten aus externen Systemen SharePoint

---

<sup>54</sup>Stevenson [2009]

<sup>55</sup>Krause u. a. [2010c]



integrieren und darstellen, ohne dass auf gewohnte Listenfunktionen (sortieren, filtern) verzichtet werden muss. Ein Entwickler kann auf externe Listen mit den Teilen des Objekt Modells zugreifen, die auch bei SharePoint Listen verwendet werden. SharePoint Server 2010 bietet darüber hinaus noch BCS WebParts, eine Integration in die Suchfunktion und die Office Integration mit der BDC Client Runtime <sup>56</sup>. Zwischen einzelnen ECT lassen sich Beziehungen erstellen, welche von den BCS WebParts zur Navigation genutzt werden können. Die Beziehungen werden als Auswahlmöglichkeit in Lookup Feldern genutzt, was das Risiko von Fehleingaben bei der Neuanlage von Datensätzen reduziert <sup>57</sup>.

Nach Outcalt [2010] können folgende vier Verfahren zur Authentifizierung am LOB System genutzt werden:

- User's Identity gibt die aktuelle Anmeldung des Benutzers weiter.
- Impersonated Windows ID nutzt den Secure Store Service. Beim Aufruf des Zielsystems wird zunächst die benötigte Benutzerkennung aus dem Secure Store Service ausgelesen und als Windows Benutzerkennung an das Zielsystem weitergereicht. Diese Methode ist nicht in SharePoint Foundation 2010 verfügbar.
- Impersonate Custom ID nutzt ebenfalls den Secure Store Service. Die Benutzerkennung wird ausgelesen und als RdbCredentials weitergereicht. Auch diese Methode ist nicht in der SharePoint Foundation 2010 verfügbar.
- Bei Revert to Self wird das Benutzerkonto des Anwendungspools vom IIS zur Anmeldung an das Zielsystem genutzt. Diese Methode der Authentifizierung kann nur mittels Powershell Kommando verfügbar gemacht werden.

Neben diesen Authentifizierungsverfahren kann pro BCS Modell, LOB System und ECT eine eigene Berechtigung vergeben werden. Nur wenn ein Benutzer mindestens über Execute Rechte auf allen drei Ebenen verfügt, kann er den ECT nutzen.

Bei der Prüfung der einzelnen Authentifizierungsverfahren ist eine Besonderheit bei der Nutzung des Secure Store Service aufgefallen. Wird eine im Secure Store Service hinterlegte Benutzerkennung mittels einer Authentifizierung per Impersonated Windows ID oder Impersonated Custom ID genutzt um mehrere externe Systeme anzubinden, muss bei der Definition der Benutzerkennung darauf geachtet werden, dass als Typ „Group“ angegeben wird. Andernfalls kann die Benutzerkennung nur für ein einziges System genutzt werden. Eine Mehrfachnut-

---

<sup>56</sup>MSDN [2010m]

<sup>57</sup>Baginski [2010]

zung führt zu einer Fehlermeldung bei der Nutzung des ECT.

Auch wenn externe Listen auf den ersten Blick wie SharePoint Listen wirken, gibt es doch deutliche Unterschiede <sup>58</sup>. Einige werden bei Baginski [2010] beschrieben:

- Kein Event Handling. Mit EventHandler kann auf verschiedene Ereignisse einer Liste reagiert werden. Es können keine EventHandler für ECT angelegt werden.
- Keine RSS Feeds, Workflows und Alerts. Da externe Listen keine EventHandler haben, können diese Funktionen nicht genutzt werden.
- Keine REST Schnittstelle.
- Keine Versionierung der Datensätze <sup>59</sup>

Vergleicht man darüber hinaus eine klassische SharePoint Liste mit einer externen Liste, können weitere Unterschiede festgestellt werden:

- Keine DataGrid Editierung. Die Datensätze von SharePoint Listen können in einer Excel-ähnlichen Ansicht dargestellt und editiert werden. Zwar können die Datensätze eines ECT auch geschrieben und aktualisiert werden, dies aber nicht in einer DataGrid Ansicht.
- Kein Datenexport nach Excel möglich.
- Keine Item Level Permissions. Die Benutzerberechtigungen in SharePoint lassen sich sehr granular vergeben. Die Berechtigungen auf ein Listenelement sind die kleinste administrierbare Berechtigungseinheit. Bei der Nutzung einer externen Liste bilden die Zugriffsrechte der externen Liste die kleinste Berechtigungseinheit.
- Keine Web 2.0 Funktionen. Benutzer können Bewertungen und Kommentare zu Datensätzen in einer SharePoint Liste abgeben. Da keine externen Daten in der Content Datenbank gespeichert werden, können solche Daten nicht bei externen Listen erfasst werden.

Wie bei Krause u. a. [2010c] beschrieben, verfügt die BDC Client Runtime über einen Cache. Hier wird aber nicht erwähnt, ob es einen serverseitigen Cache

---

<sup>58</sup>MSDN [2010m]

<sup>59</sup>MSDN [2010m]

gibt. Mit dem MS SQL Profiler wurde die Existenz eines serverseitigen Caches überprüft. Mit dem Profiler lassen sich alle Zugriffe auf eine Datenbank protokollieren und analysieren. Da die BDC Server Runtime über keinen Cache verfügt und die Daten auch nicht in der Content Datenbank gespeichert werden, erfolgt bei jeder Nutzung eines ECT ein direkter Zugriff auf das externe System (Abbildung 3.2). Werden zahlreiche Daten abgerufen und erfolgen die Abrufe innerhalb

EventClass	TextData	ApplicationName	NTUserName	LoginName	CPU	Reads	Writes	Duration	ClientProcessID	SPID	StartTime
Trace Start											2011-01-11 10:50:5
ExistingConnection	-- network protocol: TCP/IP set qu...	.Net SqlClie...		bdc					2884	93	2011-01-11 10:44:3
Audit Login	-- network protocol: TCP/IP set qu...	.Net SqlClie...		bdc					2884	67	2011-01-11 10:51:0
SQL:BatchStarting	SELECT [PLZGebiet] , [Laendercode] ...	.Net SqlClie...		bdc					2884	67	2011-01-11 10:51:0
SQL:BatchCompleted	SELECT [PLZGebiet] , [Laendercode] ...	.Net SqlClie...		bdc	16	10	0	0	2884	67	2011-01-11 10:51:0
Audit Logout		.Net SqlClie...		bdc	16	10	0	83	2884	67	2011-01-11 10:51:0
Audit Login	-- network protocol: TCP/IP set qu...	.Net SqlClie...		bdc					2884	67	2011-01-11 10:51:0
SQL:BatchStarting	SELECT [PLZGebiet] , [Laendercode] ...	.Net SqlClie...		bdc					2884	67	2011-01-11 10:51:0
SQL:BatchCompleted	SELECT [PLZGebiet] , [Laendercode] ...	.Net SqlClie...		bdc	0	10	0	0	2884	67	2011-01-11 10:51:0
Audit Logout		.Net SqlClie...		bdc	0	10	0	83	2884	67	2011-01-11 10:51:0
Trace Stop											2011-01-11 10:51:1

Abbildung 3.2: MS SQL Profiler zeigt, dass der Datenzugriff sofort stattfindet (eigene Darstellung)

kurzer Zeit, z.B. durch zahlreiche Benutzer, kann das LOB System deutlich beeinträchtigt werden.

Für die Definition eines ECT kann der SharePoint Designer (Anhang A.3) und Visual Studio 2010 genutzt werden. Mit dem SharePoint Designer können ECT dank eines Wizard schnell erstellt werden. Als mögliche Quellsysteme werden nur MS SQL Datenbanken, Windows Communication Foundation Services und .NET Verbindungsassemblies unterstützt. Verbindungen zu anderen Datenbanken können nur mit Visual Studio 2010 erstellt werden<sup>60</sup>. Zudem kann für jeden ECT ein InfoPath Formular erstellt werden (nur bei SharePoint Server 2010 Enterprise).

Bei der Entwicklung eines BCS Modells mit dem SharePoint Designer wurde eine Besonderheit festgestellt. Zu Beginn des Entwicklungsprozesses muss der Anwender sich einmalig mit dem externen System verbinden. Diese temporäre Verbindung wird im BDC Metadata Store erstellt und verbleibt hier auch nach Abschluss der Entwicklung. Es entsteht eine Vielzahl von Verbindungen zu externen Systemen. Es empfiehlt sich, die temporär erstellten externen Systeme aus dem BDC Metadata Store zu entfernen. Die Verbindungen sind zum einen über den

<sup>60</sup>Krause u. a. [2010c]

automatisch generierten Namen, zum anderen durch das Fehlen von ECT zu erkennen, die diese Verbindung nutzen (Anhang A.4).

Um ein BDC Modell auf ein anderes System zu übertragen, kann das Modell über den SharePoint Designer oder mittels Central Administration exportiert werden. Das so erstellte XML kann anschließend auch in Visual Studio 2010 eingelesen und weiter entwickelt werden. Nur über diesen Weg lassen sich andere Datenbankanbieter oder ODBC Quellen als externes System nutzen, indem die Verbindungszeichenfolge direkt in das XML eingegeben wird <sup>61</sup>. Visual Studio 2010 verfügt über eine Projektvorlage um BCS Modelle zu erstellen, so dass nahezu jede Quelle als externes System nutzbar wird. Die Projektvorlage bietet einen Designer, mit dem sich die Klasse zum Abrufen der Daten leicht entwickeln lässt <sup>62</sup>. Alternativ kann auch ein .NET Verbindungsassembly oder ein Web Service in Visual Studio 2010 entwickelt werden. Nur in Visual Studio 2010 kann auch direkt eine Datentransformation vorgenommen werden.

Bei der Entwicklung von Web Services oder Verbindungsassemblies muss darauf geachtet werden, dass diese eine BCS-konforme Schnittstelle bieten <sup>63</sup>.

Wie bei Baginski [2010] erwähnt, können ECT miteinander verbunden werden und so eine Relation bilden. Die Relation zwischen zwei ECT wird im BCS Modell definiert. Da mittels BCS auch schreibend auf Daten zugegriffen werden kann, wurde untersucht, wie sich Relationen auf Schreib- und Löschoperationen auswirken. Dabei wurden einige Besonderheiten entdeckt:

- Werden Tabellen als ECT bereitgestellt, die mit anderen Tabellen in einer Beziehung stehen, gilt die Foreign-Key Beschränkung der Datenbank.
- Wird ein Eintrag in der Primärschlüsseltabelle gelöscht und eine Foreign-Key Beschränkung verletzt, wird eine entsprechende Fehlermeldung ausgegeben (Anhang A.5).
- Wenn die Fremdschlüsselbeziehung das Löschen eines primären Elementes unterstützt (durch kaskadierende Löschoperationen oder durch Aktualisierung der Fremdschlüssel mit Null Werten), können Datensätze auch mit den Business Connectivity Service gelöscht werden. Werden ERP Systeme oder andere, komplexe Datenbanken als ECT bereitgestellt, muss sorgfältig darauf geachtet werden, dass das Quellsystem nicht durch Lösch- oder Ak-

---

<sup>61</sup>MSDN [2010i]

<sup>62</sup>Microsoft SharePoint Team Blog [2009]

<sup>63</sup>MSDN [2010c]

tualisierungsoperationen zerstört wird. Dies gilt besonders dann, wenn die Relationen der Entitäten in der Geschäftslogik des ERP Systems und nicht in der Datenbank definiert sind.

- Werden zwei ECT miteinander verbunden, aber nur für den primären ECT eine Löschoperation definiert, kann eine Löschoperation auch dann erfolgreich sein, wenn der verbundene ECT nur Leseoperationen unterstützt. In diesem Fall gelten ebenfalls die Beschränkungen der Datenbank.
- ECT lassen sich auch dann miteinander verbinden, wenn sie aus verschiedenen LOB Systemen stammen. Somit können die Daten verschiedener Systeme zusammengeführt und dem Benutzer präsentiert werden.
- Mittels ECT können Entitäten miteinander verknüpft werden, obwohl für sie keine Beziehung in der Datenbank definiert sind. Wird nun ein primärer Datensatz gelöscht, wird keine Löschoperation im verbundenen ECT ausgelöst.
- Wird eine Entität über einen zusammengesetzten Primärschlüssel identifiziert und dienen Schlüsselfelder zeitgleich als Fremdschlüssel, kann diese Entität im BCS Modell keine Beziehung eingehen.

Eine Übersicht über die hier beschriebenen Fälle findet sich in Anhang A.6.

In den folgenden Abschnitten werden einige Anwendungsfälle skizziert und Hinweise zu deren Implementation gegeben.

**Anwendungsfall 1: Aufbau eines Vertriebsdashboard** Die Mitarbeiter des Vertriebs sollen über die aktuelle Markt- und Unternehmenssituation informiert werden. Dies soll über eine eigene SharePoint Webseite geschehen, auf der verschiedene Key Performance Indikatoren und Diagramme angezeigt werden. Als Datenquelle soll die Datenbank des Data Warehouse System genutzt werden. Um diese Anforderung umzusetzen, müssen zunächst in einem BCS Modell die ECT definiert werden. Da es sich um eine MS SQL Datenbank handelt, kann dies leicht mit dem SharePoint Designer realisiert werden. Der Aufwand zur Anbindung der Daten ist sehr gering.

Zur Visualisierung der Daten können die vorhandenen BCS WebParts genutzt werden, sofern SharePoint Server 2010 eingesetzt wird. Wird die SharePoint Foundation 2010 genutzt, müssen eigene WebParts entwickelt werden, welche die Daten anzeigen. Der Aufwand erhöht sich um die Implementierungszeit des

WebPart.

**Anwendungsfall 2: Equipment- und Ressourcenverwaltung** Das Unternehmen stellt seinen Mitarbeitern verschiedene Ressourcen und Equipment für Besprechungen und Kundenbesuche zur Verfügung. Die Ressourcen werden in dem hauseigenen Planungswerkzeug erfasst und verwaltet. Die Mitarbeiter sollen mit einer Webseite über die Ressourcen und deren Verfügbarkeit informiert werden. Die Verplanung der Ressourcen übernimmt dabei ein Mitarbeiter zentral. Dieser ist über Vormerkungen und Reservierungen informiert.

Zur Realisierung müssen zunächst in einem BCS Modell die External Content Types zur Darstellung der Ressourcen und deren Verplanungen erstellt werden. Da es sich hierbei um eine MS SQL Datenbank handelt, kann dies leicht mit dem SharePoint Designer umgesetzt werden. Es wird eine Beziehung zwischen dem ECT der Ressourcen und dem ECT mit deren Verplanungen erstellt und für beide Lese- und Schreiboperationen definiert. Anschließend werden in der Central Administration die erforderlichen Berechtigungen vergeben. Wird SharePoint Server 2010 genutzt, kann das vorhandene BCS List WebPart und BCS related List WebPart genutzt werden. Auf einer neuen Webseite werden beide WebParts positioniert und miteinander verbunden. Nun kann jeder Mitarbeiter sehen, welche Ressourcen verfügbar sind. Durch Auswahl einer Ressource wird in dem zweiten WebPart angezeigt, wann die Ressource verplant ist. Der Mitarbeiter, der für die Verplanung zuständig ist, kann neue Ressourcen anlegen und bestehende löschen oder aktualisieren. Zudem kann er neue Reservierungen für vorhandene Ressourcen eingeben. Dabei wird automatisch per Lookup Feld sichergestellt, dass nur für vorhandene Ressourcen Termine eingegeben werden können.

Wird die SharePoint Foundation 2010 verwendet, können die externen Daten nur als einfache Listen dargestellt werden. Auch hiermit können sich die Mitarbeiter über verfügbare und verplante Ressourcen informieren bzw. neue Datensätze anlegen. Den Komfort der Navigation, den die BCS WebParts bieten, muss neu entwickelt werden. Der Aufwand steigt deutlich.

**Anwendungsfall 3: Benachrichtigung des Vertriebsmanagements** Sobald ein neuer Kundenauftrag gewonnen wird, soll das Management des Vertriebs darüber per E-Mail informiert werden. Gewonnene Aufträge werden im CRM System und in der Finanzbuchhaltung abgebildet. Durch Anbindung der entsprechenden Tabellen mittels BCS lässt sich die Anforderung nicht erfüllen, da keine Event

Handler, Benachrichtigungssysteme oder Workflows unterstützt werden. Es ist zu prüfen, ob sich die Anforderung nicht in den betroffenen Systemen umsetzen lässt.

**Alternative zu BCS: Datensynchronisation per Timer Job** Mit dem Verzicht auf die BCS steigt der Implementierungsaufwand erheblich. Es ergeben sich aber neue Möglichkeiten, da die Restriktionen der BCS umgangen werden.

In der Zeiterfassung und dem ERP-System des Unternehmens werden Daten von Mitarbeitern gespeichert. Wird ein Mitarbeiter eingestellt oder verlässt ein Mitarbeiter das Unternehmen, müssen zahlreiche Stellen informiert werden. Bisher wurden die beteiligten Stellen per Telefon oder E-Mail informiert. Die betroffenen Stellen sollen zukünftig automatisch informiert werden. Wie bereits aus dem Anwendungsfall 3 hervor geht, lässt sich eine automatische Benachrichtigung mittels BCS nicht realisieren. Statt dessen wird ein neuer SharePoint Timer Job definiert. Dieser wird von einem Hintergrundprozess automatisch nach einem definierten Zeitplan ausgeführt. Aufgabe des Jobs ist es, die bestehenden Personaldaten auszulesen und mit einer SharePoint Liste abzugleichen. Wird ein neuer Eintrag im Personalstamm angelegt oder ein bestehender geändert, werden die aktuellen Daten in der SharePoint Liste hinzugefügt bzw. geändert. Zwar liegen die Daten nun redundant in verschiedenen Systemen vor, Workflows oder Benachrichtigungen lassen sich aber so realisieren.

In einer neuen Klasse, die von SPJobDefinition erbt, wird in der überschriebenen Methode Execute der Job implementiert. Nachdem die Daten aus der Personal-tabelle ausgelesen wurden, wird jeder Datensatz mit der Zielliste abgeglichen. Der vollständige Quelltext des Jobs befindet sich im Listing L.1. Der neue Job soll als Feature bereitgestellt werden. Dazu wird eine weitere Klasse erstellt, die von SPFeatureReceiver erbt. Die Basisklasse bietet verschiedene Methoden an, mit denen auf die Ereignisse des Features reagiert werden kann. In der Methode FeatureActivated wird zunächst die SharePoint Liste erstellt, die als Ziel der Synchronisation dient. Danach wird geprüft, ob der Job bereits in der Central Administration enthalten ist. Existiert eine Job Instanz, wird diese gelöscht. Nun wird der neue Job erstellt. In der Methode FeatureDeactivating wird die Instanz des Timer Jobs gelöscht. Anschließend wird die SharePoint Liste gelöscht, die als Ziel der Synchronisation gedient hat. Der vollständige Quelltext des FeatureReceivers befindet sich im Listing L.2.

**Empfehlungen** Der Business Connectivity Service ist der zu bevorzugende Weg beim Datenzugriff auf externe Systeme, da die Daten nicht nur mit den vorhandenen BCS WebParts verarbeitet werden können, sondern auch in jeder anderen Eigenentwicklung genutzt werden können. Mit den vorhandenen BCS WebParts lassen sich komfortable Dashboards auf Basis von Geschäftsdaten erstellen. Durch Filter- und Chart WebParts können die Daten zielgruppengerecht und ansprechend präsentiert werden. Die folgenden Tabellen dienen als Entscheidungshilfe für zukünftige Projekte. In Tabelle 1 wird dargestellt, welche Möglichkeiten der BCS in den einzelnen SharePoint Versionen bietet. Tabelle 2 beinhaltet einen Vergleich der Entwicklungswerkzeuge. Tabelle 3 beinhaltet Empfehlungen, wann und wie der BCS verwendet werden soll.

<b>Funktion</b>	<b>SharePoint Foundation 2010</b>	<b>SharePoint Server 2010</b>	<b>SharePoint Server 2010 Enterprise</b>
Lesender und schreibender Zugriff	•	•	•
Authentifizierung mittels Secure Store Service	-	•	•
Darstellung als Liste	•	•	•
BCS WebParts	-	•	•
Client Runtime	-	-	•

- wird unterstützt
- wird mit Einschränkung unterstützt
- wird nicht unterstützt

Tabelle 1: BCS Funktionen nach SharePoint Versionen (eigene Darstellung)



Funktion	SharePoint Designer	Visual Studio
Zugriff auf MS SQL Datenbank	•	•
Zugriff auf Web Service	○ (nur wenn Dienst BCS konform ist)	•
Zugriff auf Verbindungsassembly	•	•
Erstellen von Verbindungsassembly	-	•
Zugriff auf ODBC Quellen	-	•
Zugriff auf sonstige Quellen	-	•

- wird unterstützt
- wird mit Einschränkung unterstützt
- wird nicht unterstützt

Tabelle 2: Vergleich der Entwicklungswerkzeuge für BCS (eigene Darstellung)

Anwendungsfall	Empfehlung	Alternative
Lesender Zugriff auf Datenbanken	BCS verwenden, wenn auf Cache verzichtet werden kann	Eigener Dienst erstellen, der Cache bereitstellt
Lesender Zugriff auf Web Service	BCS nur verwenden, wenn <ul style="list-style-type: none"> <li>• Authentifizierung per Secure Store Service (SSS)</li> <li>• Darstellung als externe Liste</li> </ul> gefordert sind	Dienst direkt aufrufen
Schreibender Zugriff auf Datenbanken	BCS nur verwenden bei <ul style="list-style-type: none"> <li>• einzelnen Tabellen</li> <li>• einfachen Datenmodellen</li> </ul> nicht verwenden bei komplexen Systemen (z.B. ERP System)	evtl. vorhandener oder eigenen Dienst verwenden
Nutzung von Listenereignisse und Workflows	kein BCS verwenden	Daten per Dienst bereitstellen oder in SharePoint Liste synchronisieren
REST Zugriff	kein BCS verwenden	Daten per Dienst bereitstellen oder in SharePoint Liste synchronisieren

Exportieren von Daten	kein BCS verwenden	WebPart erstellen, der BCS Daten anzeigt und Export Funktion anbietet
Verwendung von SSS	nur verwenden, wenn Benutzer keine Rechte am Quellsystem haben soll	Benutzer am Quellsystem berechtigen

Tabelle 3: Empfehlungen für die Nutzung von BCS (eigene Darstellung)

### 3.3.2 WebPart

Ein WebPart ist ein ASP.NET Control, welches auf dem Web Frontend Server ausgeführt wird. Es ist ein lose gekoppeltes Control, was bedeutet, dass es von einem Benutzer auf verschiedenen WebPart Seiten positioniert werden kann. Eine WebPart Seite ist eine ASP.NET Seite, die eine oder mehrere WebPart Zonen beinhaltet. Jeder WebPart kann in jeder WebPart Zone beliebig oft positioniert werden. Der Benutzer erhält so die Möglichkeit, den Inhalt und das Aussehen einer Seite anzupassen (Abbildung 3.3) <sup>64</sup>. Jeder WebPart verfügt über Eigen-

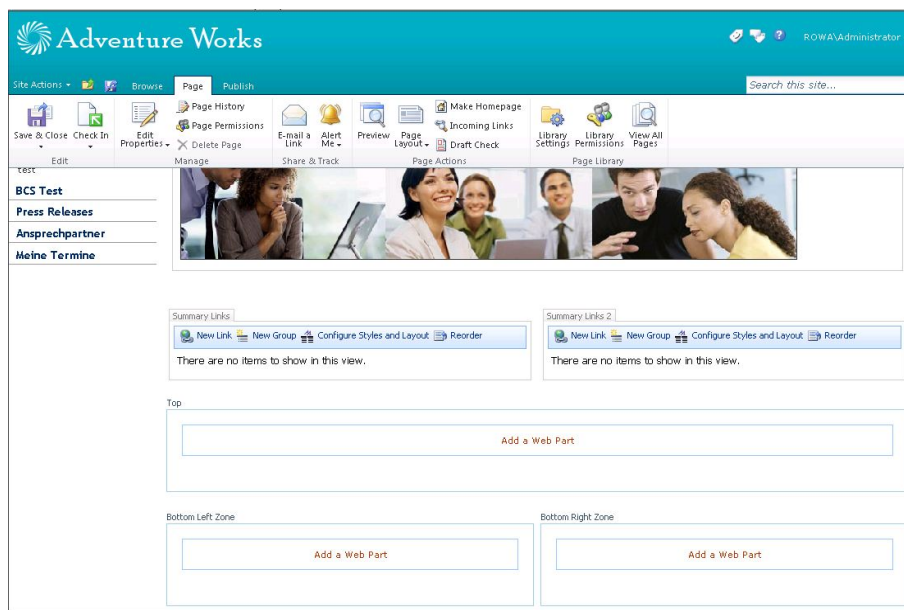


Abbildung 3.3: WebPart Zonen (eigene Darstellung)

schaften, die vom Benutzer angepasst werden können, wie etwa Titel oder Beschreibung. Ein Entwickler kann eigene Eigenschaften hinzufügen, die der Benutzer nach seinen Bedürfnissen anpassen kann. Die Anpassungen können pro Benutzer individuell erfolgen oder für alle Benutzer gültig sein. WebParts bieten eine einfache Möglichkeit, Daten und Inhalte benutzergerecht darzustellen <sup>65</sup>.

Während sich nach MSDN [2010I] zwei Arten von WebParts unterscheiden lassen:

- ASP.NET WebParts: Können in jeder ASP.NET WebPart Seite genutzt werden. Sie sind nicht auf SharePoint beschränkt, solange keine SharePoint Objekte verwendet werden.

<sup>64</sup>Krause u. a. [2010d]

<sup>65</sup>MSDN [2011a]

- SharePoint WebParts: Sind eine Spezialisierung von ASP.NET WebPart. Mit ihnen lassen sich zwei WebParts verbinden, die in unterschiedlichen WebPart Seiten liegen. Klassische ASP.NET WebParts können nur verbunden werden, wenn sie sich auf der selben WebPart Seite befinden. Diese Art von WebParts kann nur in SharePoint eingesetzt werden.

kann man in der praktischen Arbeit diese zusätzlichen Arten differenzieren:

- Visual WebPart: Ist ein ASP.NET oder SharePoint WebPart, dass zur Laufzeit ein ASP.NET UserControl lädt. Der eigentliche Code mit den gewünschten Programmfunktionen befinden sich im UserControl. Diese Art von WebPart lässt sich nicht als Sandbox Lösung bereitstellen.
- Sandbox WebPart: Ist ein WebPart, welches als Sandbox Lösung bereitgestellt wird. Hierbei handelt es sich um ein ASP.NET oder SharePoint WebPart, welches mit den Restriktionen und Sicherheitsstufen einer Sandbox Solution ausgeführt wird.

In SharePoint können alle verfügbaren ASP.NET WebParts genutzt werden <sup>66</sup>. Wie bereits bei den BCS WebParts beschrieben, können WebParts untereinander verbunden werden. Hiermit kann ein WebPart als Filter oder Auswahl für einen anderen WebPart dienen.

Beginnt man mit der Entwicklung eines neuen WebPart fällt auf, dass Visual Studio 2010 lediglich eine Projektvorlage für ein Visual WebPart bietet. Sollen andere Arten entwickelt werden, muss ein leeres SharePoint Projekt erstellt werden. Hier kann anschließend ein neues Element vom Typ WebPart hinzugefügt werden. Visual Studio 2010 bietet keinen Designer zum Erstellen eines WebPart an. Alle Bedienelemente und Ausgaben (z.B. HTML) müssen im Quellcode des WebPart definiert werden. Es erfolgt keine Trennung zwischen Code und Layout. Um dennoch den Code für das Layout von der eigentlichen Programmlogik zu trennen, sollte der Code des Layouts in eine eigene Methode, Klasse oder partielle Klasse ausgelagert werden. Die Lesbarkeit und Wartbarkeit des Programmcodes wird somit deutlich erhöht. Wird ein Visual WebPart entwickelt, steht für den eigentlichen WebPart ebenfalls kein Designer zur Verfügung. Da hier aber zusätzlich ein ASP.NET UserControl geladen wird und für diesen ein Designer existiert, kann dieser auch genutzt werden. Durch den Einsatz des Usercontrols erfolgt eine Trennung zwischen Layout und Code. Visual WebParts können nicht als Sandbox Lösung bereitgestellt werden, da das UserControl auf dem Dateisystem des Webserverns liegt.

---

<sup>66</sup>MSDN [2010]

Wie bereits erwähnt, verfügen WebParts über Attribute und Eigenschaften, die von einem Benutzer geändert werden können. Dazu müssen in der Klasse des WebParts entsprechende Attribute als Feld gekapselt werden. Direkt über der Definition des Feldes werden in eckigen Klammern Optionen angegeben, die das Verhalten und Aussehen des Feldes in der Tool Pane des WebPart bestimmen<sup>67</sup>. Dies wird in Anhang A.7 dargestellt.

Bei der Prüfung der Möglichkeiten, die sich mit der Definition von Attributen und Eigenschaften ergeben, ist folgendes Problem aufgetreten. Wird ein Visual WebPart entwickelt, kann das UserControl nicht auf die Eigenschaften zugreifen, da sich die eigentliche Programmfunktion im Programcode des ASP.NET UserControl befindet. Die Ursache für dieses Verhalten findet sich in der Vorlage, die von Visual Studio 2010 genutzt wird, um den WebPart zu erstellen. Hier wird zwar das UserControl geladen, erhält aber keine Referenz auf den übergeordneten WebPart. Durch Anpassung der WebPart- und der UserControl Klasse kann das Problem behoben werden. Zunächst wird in der Klasse des Usercontrols ein neues, öffentliches Attribut vom Typ des WebParts (im Beispiel Attribut myWebPart) erstellt (Listing L.1). Anschließend wird in der Klasse des WebParts ein neues At-

```
1  /// <summary>
2  /// Definition des UserControls
3  /// </summary>
4  public partial class VisualUserControl1 : UserControl
5  {
6      /// <summary>
7      /// Definition der Referenz auf das übergeordnete WebPart.
8      /// Dies ist erforderlich , um auf die Eigenschaften des
9      /// WebParts zugreifen zu können
10     /// </summary>
11     private VisualWebPart1 myWebPart = null;
12
13     /// <summary>
14     /// Gets or sets WebPart
15     /// </summary>
16     public VisualWebPart1 WebPart
17     {
18         get { return this.myWebPart; }
19         set { this.myWebPart = value; }
20     }
21     ...
22 }
```

Listing L.1: Usercontrol mit Zugriffsmöglichkeit auf die WebPart Eigenschaften

tribut vom Typ des UserControl (im Beispiel Attribut mainControl) definiert. In der

---

<sup>67</sup>MSDN [2010]

Methode CreateChildControls der WebPart Klasse wird diesem Attribut eine Instanz des Usercontrols zugewiesen. Anschließend wird dem Attribut myWebPart des Usercontrols die aktuelle WebPart Instanz zugewiesen. Im Programmcode des Usercontrols sind nun alle Eigenschaftswerte des WebParts erreichbar (Listing L.2).

```
1  /// <summary>
2  /// Definition des WebPart
3  /// </summary>
4  public class VisualWebPart1 : WebPart
5  {
6      /// <summary>
7      /// Pfad zum UserControl
8      /// </summary>
9      private const string ASCXPath = @"~/_CONTROLTEMPLATES/
10         CustomControl/VisualWebPart1/VisualUserControl1.ascx";
11
12     /// <summary>
13     /// Attribut vom Typ UserControl. Dieses Attribut ist
14     /// erforderlich, um die Attributswerte der Klasse
15     /// VisualUserControl1 zu erreichen
16     /// </summary>
17     private VisualUserControl1 mainControl = null;
18
19     /// <summary>
20     /// Laden der Controls
21     /// </summary>
22     protected override void CreateChildControls()
23     {
24         // Instanz des UserControl erstellen
25         this.mainControl =
26             (VisualUserControl1) this.Page.LoadControl(ASCXPath);
27
28         // Füllen des Attributs WebPart der UserControl
29         // Instanz. Das UserControl erhält so die
30         // Referenz auf den übergeordneten WebPart
31         this.mainControl.WebPart = this;
32         Controls.Add(this.mainControl);
33     }
34 }
```

Listing L.2: Visual WebPart mit notwendigen Anpassungen

Bei der Entwicklung von WebParts muss der Lebenszyklus des WebPart beachtet werden. Rüsche u. Quix [2008] beschreibt einige Methoden und deren Aufrufreihenfolge. Mittels Programmdebugger wurde die Aufrufreihenfolge der Methoden überprüft und um die Methoden OnInit und OnUnLoad ergänzt. Daraus ergeben sich folgende wichtige Methoden und deren Reihenfolge, die überschrieben werden können, um eigene Funktionen zu implementieren.

1. OnInit: In dieser Methode erfolgt die Instantiierung der erforderlichen Objekte wie etwa Steuerelemente.
2. CreateChildControls: In dieser Methode werden alle Steuerelemente dem WebPart hinzugefügt und Standardwerte gesetzt. Hier werden auch die Ereignisse der Steuerelemente registriert <sup>68</sup>.
3. OnLoad: In dieser Methode sind bereits alle Steuerelemente verfügbar. Hier können Verbindungen zu Datenquellen geöffnet werden. Beim erstmaligen Aufruf des WebParts wird diese Methode vor der Methode CreateChildControls aufgerufen. Handelt es sich um einen Postback, wird die Methode nach CreateChildControls aufgerufen <sup>69</sup>.
4. Events der Steuerelemente: Hier werden alle Events der Steuerelemente, wie etwa Klickereignisse von Buttons, ausgeführt <sup>70</sup>.
5. OnPreRender: Diese Methode wird unmittelbar vor der Ausgabe des WebParts aufgerufen. Es sind alle Benutzereingaben der Steuerelemente verfügbar, alle Events wurden verarbeitet. Hier können in Abhängigkeit der Benutzereingaben Daten aus den Datenquellen abgerufen oder sonstige Operationen durchgeführt werden, die Benutzereingaben benötigen <sup>71</sup>.
6. Render: Diese Methode erzeugt die Ausgabe der Steuerelemente und sonstige Inhalte des WebParts. Hier können eigene Ausgaben, etwa HTML Anweisungen oder CSS Klassen, hinzugefügt werden <sup>72</sup>.
7. OnUnLoad: Hier werden alle Steuerelemente zerstört. Offene Datenverbindungen oder sonstige eigene Objekte können hier zerstört werden.

In den folgenden Abschnitten werden einige Anwendungsfälle skizziert und Hinweise zu deren Implementierung gegeben.

---

<sup>68</sup>Rüsche u. Quix [2008]

<sup>69</sup>Rüsche u. Quix [2008]

<sup>70</sup>Rüsche u. Quix [2008]

<sup>71</sup>Rüsche u. Quix [2008]

<sup>72</sup>Rüsche u. Quix [2008]

**Anwendungsfall 1: Kontaktdaten anzeigen** Auf vielen Seiten der Unternehmensportale sollen Kontaktdaten der Personen, die für den jeweiligen Inhalt verantwortlich sind, angezeigt werden. Die Redakteure der Inhalte sollen selbst bestimmen können, wer für den Inhalt verantwortlich ist und ob diese Informationen angezeigt werden sollen.

Zur Realisierung der Anforderung bietet sich ein WebPart an. Jeder Seite in SharePoint ist ein Content Type zugewiesen, der die Metadaten der Seite festlegt. Zur Aufnahme des verantwortlichen Mitarbeiters wird der Content Type der Seiten angepasst und um ein Feld vom Typ Benutzer erweitert. In den Metadaten der Seite kann der verantwortliche Mitarbeiter eingetragen werden. In einem leeren SharePoint Projekt wird ein WebPart hinzugefügt. Alle Steuerelemente zur Anzeige der Kontaktdaten werden in der Methode OnInit instanziiert und in der Methode CreateChildControls hinzugefügt. In der Methode OnPreRender werden die Metadaten der Seite ausgelesen. Mit dem SharePoint Objektmodell werden nun die Daten des verantwortlichen Mitarbeiters gelesen.

**Anwendungsfall 2: Visualisieren von Service-Aktivitäten eines Kunden** Das Unternehmen bietet seinen Kunden in einem Kundenportal zahlreiche Informationen, wie Handbücher, Vertragsinformationen oder Rechnungen. Störungen können die Kunden über eine eigene Servicehotline melden. Um die Transparenz bei der Bearbeitung der Serviceanfrage zu erhöhen, sollen im Kundenportal alle Anfragen und die daraus resultierenden Schritte dargestellt werden. Die erforderlichen Daten liegen im CRM System vor, auf das die Kunden keinen Zugriff haben. Zur Darstellung der Daten bieten sich zwei Methoden an. Zum einen können die Daten per BCS gelesen und als Liste dargestellt werden. Alternativ lassen sich die Daten mit einem WebPart visualisieren. Die Informationen sollen ansprechend ausgegeben werden. Die Umsetzung der Anforderung mit einem WebPart bietet sich an.

Zur Realisierung der Anforderung wird zunächst ein BCS Modell erstellt, welches die Daten aus dem CRM System repräsentiert. Anschließend wird ein neues Visual WebPart Projekt erstellt. Zur Darstellung der Daten wird ein Grid verwendet, welches die Daten gruppiert pro Serviceanfrage ausgibt. Das Grid wird in dem UserControl mit Unterstützung des Designers platziert. Anschließend werden weitere Steuerelemente zur Gestaltung des Layouts platziert (HTML Tabellen, CSS Klassen etc.) In der Methode OnLoad werden die Daten des Kunden aus der BCS Quelle in eine Liste gelesen, die dem Grid als Datenquelle dient.



**Anwendungsfall 3: Organisationsbrowser** Mitarbeiter, Kunden und Partner sollen die Möglichkeit erhalten, Kollegen und Ansprechpartner zu finden. Dazu sollen sie zunächst über eine Baumstruktur einen Mitarbeiter auswählen. Die Mitarbeiter werden nach Abteilungszugehörigkeit gruppiert angezeigt. Durch Auswahl eines Mitarbeiter sollen die Kontaktdaten des Mitarbeiters, eine Beschreibung seiner Tätigkeit sowie ein Foto angezeigt werden. Die Fotos der Mitarbeiter sind in einer SharePoint Bildbibliothek abgelegt. Die Beschreibung der Tätigkeit sowie die Kontaktdaten sind im Benutzerprofil des Mitarbeiters abgelegt. Die Hierarchie der Mitarbeiter ergibt sich durch den Vorgesetzten des Mitarbeiters. Dieser wird im Active Directory gepflegt und ist ebenfalls Bestandteil des Benutzerprofils. Zur Realisierung können zwei WebParts genutzt werden, welche über eine Schnittstelle miteinander verbunden werden. Der Provider WebPart zeigt die Hierarchie der Mitarbeiter an. Der Consumer WebPart zeigt das Bild sowie die Daten des Mitarbeiters an. Zur Realisierung muss zusätzlich noch eine Schnittstelle implementiert werden. Die Bereitstellung als Sandbox Lösung kommt nicht in Frage, da diese nicht mit anderen WebParts kommunizieren können. Nachdem ein leeres SharePoint Projekt erstellt wurde, wird ein neues Element vom Typ Interface hinzugefügt. Hier wird die Schnittstelle zwischen den beiden WebParts definiert. Es soll der Name des Mitarbeiters an den Consumer WebPart übergeben werden. Als nächstes wird der Provider WebPart als neues Element dem Projekt hinzugefügt. Die wesentlichen Codeelemente werden in Anhang A.10 dargestellt. Der Consumer WebPart muss lediglich ein Attribut vom Typ der Schnittstelle sowie eine Methode zum Abrufen der Schnittstelle implementieren. Anschließend kann auf alle Member der Schnittstelle direkt zugegriffen werden. Nachdem beide Webparts bereitgestellt wurden, müssen sie auf eine SharePoint WebPart Seite abgelegt werden. Anschließend kann im Editiermodus der Seite die Verbindung zwischen den beiden WebParts hergestellt werden.

**Alternative zu Visual WebPart: Silverlight** Visual WebParts bieten durch die Kombination eines klassischen WebParts mit einem ASP.NET UserControl eine einfache Möglichkeit, wiederverwendbare Anwendungen mit Unterstützung eines Designers zu erstellen. Jedoch können sie nicht als Sandbox Lösung bereitgestellt werden.

Alternativ kann eine Silverlight Anwendung erstellt werden, welche mit einem bereits in SharePoint vorhandenen WebPart an den Benutzer gesendet wird. Silverlight Anwendungen werden nicht auf dem Server ausgeführt, sondern auf dem Client. Durch das Einbetten einer solchen Anwendung in eine Webseite lädt der Browser des Benutzers den Programmcode herunter und führt diesen anschließend aus. Die Laufzeitumgebung von Silverlight wird durch ein Browser Plugin bereitgestellt. Dem Entwickler steht in Visual Studio 2010 ein grafischer Designer zur Verfügung (Anhang A.11). Da die Anwendung auf dem Client ausgeführt wird, unterliegt sie nicht den Restriktionen einer Sandbox Lösung. SharePoint Objekte werden mit dem SharePoint Client Objektmodell erreicht.

Im folgenden Beispiel sollen in einer Kartenanwendung alle offenen Serviceanfragen dargestellt werden. Die Servicetechniker erhalten so einen Überblick über ihre Aufgaben und können ihre Routen optimieren. In einem neuen Silverlight Projekt werden zunächst alle benötigten Steuerelemente positioniert. Im Programmcode der Anwendung wird anschließend der aktuell am System angemeldete Benutzer ermittelt und alle offenen Serviceanfragen des Benutzers aus einer Liste gelesen. Die Liste wird mittels BCS aus dem CRM System gefüllt. Alle Anfragen des Client an das SharePoint System werden durch das Client Objektmodell als Webservice Anfrage gestellt. Mit dem ClientContext Objekt der Anwendung wird zunächst definiert, welche Objekte aus dem SharePoint System gelesen werden sollen. Sind alle definiert, erfolgt eine asynchrone Anfrage an den Server. Durch einen Event Handler wird die Antwort des Servers verarbeitet. Es erfolgt keine Blockierung der Seite während des Ladevorgangs. Sind alle Elemente gelesen, wird für jeden Datensatz ein neuer PushPin auf der Karte angelegt. Sind alle Markierungen platziert, wird der Kartenausschnitt und der Zoom der Karte angepasst, um alle Kunden anzeigen zu können (Abbildung 3.4). Der Quellcode der Anwendung befindet sich in Anhang L.7. Um die Anwendung als SharePoint Modul bereitstellen zu können, wird der Lösung ein leeres SharePoint Projekt hinzugefügt. In diesem Projekt wird ein neues Element vom Typ SharePoint Modul erstellt. In den Eigenschaften des Moduls wird ein neuer Projektausgabeverweis hinzugefügt. Somit ist das Silverlight Projekt Bestandteil des Moduls. Zum Schluss wird in der Elements.xml des Moduls ein File Element eingefügt. Dieses gibt an, welche Teile des Projektes auf welcher Adresse bereitgestellt werden sollen. Hier wird die Zielbibliothek der Anwendung angegeben (Anhang L.9). Diese Adresse wird im Silverlight WebPart als Quelladresse der Silverlight Anwendung

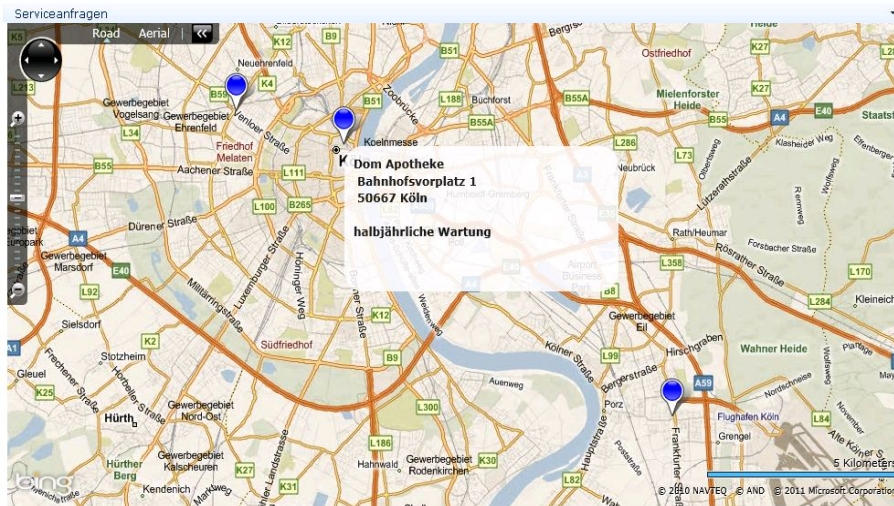


Abbildung 3.4: Silverlight Anwendung (eigene Darstellung)

angegeben.

**Empfehlungen** WebParts eignen sich immer dann, wenn Benutzer die Inhalte auf einer Seite nach ihren Bedürfnissen anpassen sollen. Da das vollständige SharePoint Objektmodell zur Verfügung steht, eignen sich WebParts auch, um den Benutzerzugriff zu kontrollieren. Zudem bieten WebParts eine einfache Möglichkeit, Benutzern Daten aus einer SharePoint Liste oder sonstigen Quellen zu präsentieren, ohne dass die Benutzer Berechtigungen auf die Datenquelle haben. Der Datenzugriff lässt sich sehr leicht kontrollieren. Mit WebParts können ansprechende Benutzeroberflächen für die Interaktion mit Listen oder Workflows erstellt werden. Des weiteren lassen sich Inhalte wie CSS, JavaScript oder Silverlight Anwendungen an den Browser des Benutzers senden, ohne die Master Page von SharePoint anpassen zu müssen. Die folgenden Tabellen dienen als Entscheidungsgrundlage für zukünftige Projekte. Tabelle 4 beinhaltet Empfehlungen, wann welche Art von WebPart verwendet werden soll. Tabelle 5 beinhaltet allgemeine Handlungsempfehlungen beim Entwickeln von WebParts.

WebPart Arten	Empfehlung	Alternative
ASP.NET WebPart	wenn <ul style="list-style-type: none"> <li>• Verwendung in Sandbox nicht ausgeschlossen werden soll</li> <li>• viele Ressourcen benötigt werden</li> </ul>	Silverlight Anwendung bei Intranet, anderer WebPart Typ

Sandbox WebPart	<p>verwenden wenn</p> <ul style="list-style-type: none"> <li>• schnelle Bereitstellung gefordert ist</li> <li>• Monitoring des Ressourcenverbrauch gefordert ist</li> <li>• kein Wartungsfenster für produktive Bereitstellung existiert</li> <li>• wenige Ressourcen verbraucht werden</li> <li>• in SharePoint Online Umgebungen</li> </ul> <p>nicht verwenden wenn</p> <ul style="list-style-type: none"> <li>• verbundene WebParts erstellt</li> <li>• externe Systeme benötigt werden</li> </ul>	Silverlight Anwendung bei Intranet
SharePoint WebPart	<p>wenn</p> <ul style="list-style-type: none"> <li>• seitenübergreifende Verbindung gefordert wird</li> <li>• viele Ressourcen benötigt werden</li> </ul> <p>nicht verwenden, wenn WebPart in anderer ASP.NET Applikation wiederverwendet werden soll</p>	Silverlight Anwendung bei Intranet, anderer WebPart Typ
Visual WebPart	<p>wenn</p> <ul style="list-style-type: none"> <li>• auf Sandbox verzichtet werden kann</li> <li>• Wartungsfenster zur Bereitstellung existieren</li> <li>• viele Ressourcen benötigt werden</li> <li>• grafischer Designer benötigt wird</li> </ul>	Silverlight Anwendung bei Intranet, anderer WebPart Typ

Tabelle 4: WebPart Arten (eigene Darstellung)

Anforderung	Empfehlung	Alternative
Zugriff auf externe Daten	wenn möglich BCS verwenden oder Web Service	direkter Zugriff auf externe Datenbank
WebPart Eigenschaften in der Tool Pane	wenn <ul style="list-style-type: none"> <li>• jede WebPart Instanz über verschiedene Parameterwerte verfügen soll</li> <li>• Personalisierung gefordert ist</li> </ul>	globale Konfigurationsdatei, wenn alle Instanzen gleiche Parameterwerte haben sollen
Zugriff auf Web Service	Endpunkt in separater web. config Datei ablegen	Endpunkt in globaler web. config oder im Quellcode des WebParts definieren
verbundene Web-Parts	erstellen wenn <ul style="list-style-type: none"> <li>• jedes WebPart auch einzeln genutzt werden kann</li> <li>• ein WebPart mehrere andere WebParts versorgen soll</li> <li>• WebParts frei positioniert werden sollen</li> </ul>	Funktionen in einem Web-Part realisieren

Tabelle 5: Allgemeine Empfehlungen bei der Entwicklung von WebParts (eigene Darstellung)

### 3.3.3 Workflow Foundation

Workflows sind Programme zur Abbildung und Automatisierung von Geschäftsprozessen. Ein Workflow folgt einer definierten Reihenfolge von Aktivitäten, die Teile eines Geschäftsprozesses repräsentieren <sup>73</sup>. Eine Aktivität ist ein wiederverwendbares Stück Code, welches wie ein Baustein im Workflow eingesetzt werden kann. Werden während des Ablaufs Informationen von Benutzern benötigt, stoppt die Programmausführung, bis die Informationen vorliegen. Um während dieser Wartezeit keine Ressourcen zu belegen, wird die Workflow Instanz serialisiert und gespeichert (dehydriert), bis die Informationen vorliegen <sup>74</sup>. Zur Realisierung von Workflows wurde mit dem .NET Framework 3.0 die Windows Workflow Foundation (WF) eingeführt <sup>75</sup>.

MSDN [2010o] unterscheidet zwei Arten von Workflows. Ein Sequential Workflow ist einem Flussdiagramm oder einer Prozesskette sehr ähnlich. Die Aktivitäten ergeben eine eindeutige Reihenfolge, die Verzweigungen und Wiederholungen zulässt, aber keine Rücksprünge unterstützt (Abbildung 3.5). Ein State Ma-

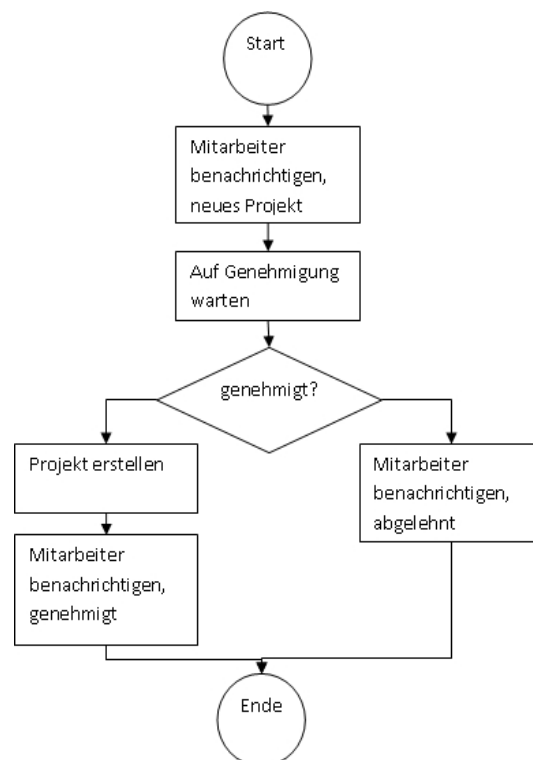


Abbildung 3.5: Sequentieller Workflow (eigene Darstellung)

chine Workflow besteht aus mehreren States, die den Zustand eines Prozesses

<sup>73</sup>TechNet [2010e]

<sup>74</sup>Krause u. a. [2010e]

<sup>75</sup>TechNet [2010e]

darstellen. Jeder Zustand (State) definiert eine Abfolge von Aktivitäten. Entsprechend dem aktuellen Zustand einer Workflow Instanz werden die zugehörigen Aktivitäten ausgeführt. Zu jedem Zeitpunkt in der Sequenz kann der Zustand der Workflow Instanz neu gesetzt werden.

Aus diesen Definitionen lässt sich ableiten, dass ein State Machine Workflow sich immer dann empfiehlt, wenn eine Realisierung als Sequential Workflow zu zahlreichen Verzweigungen führen würde oder wenn Rücksprünge im Ablauf des Geschäftsprozesses notwendig sind (Abbildung 3.6).

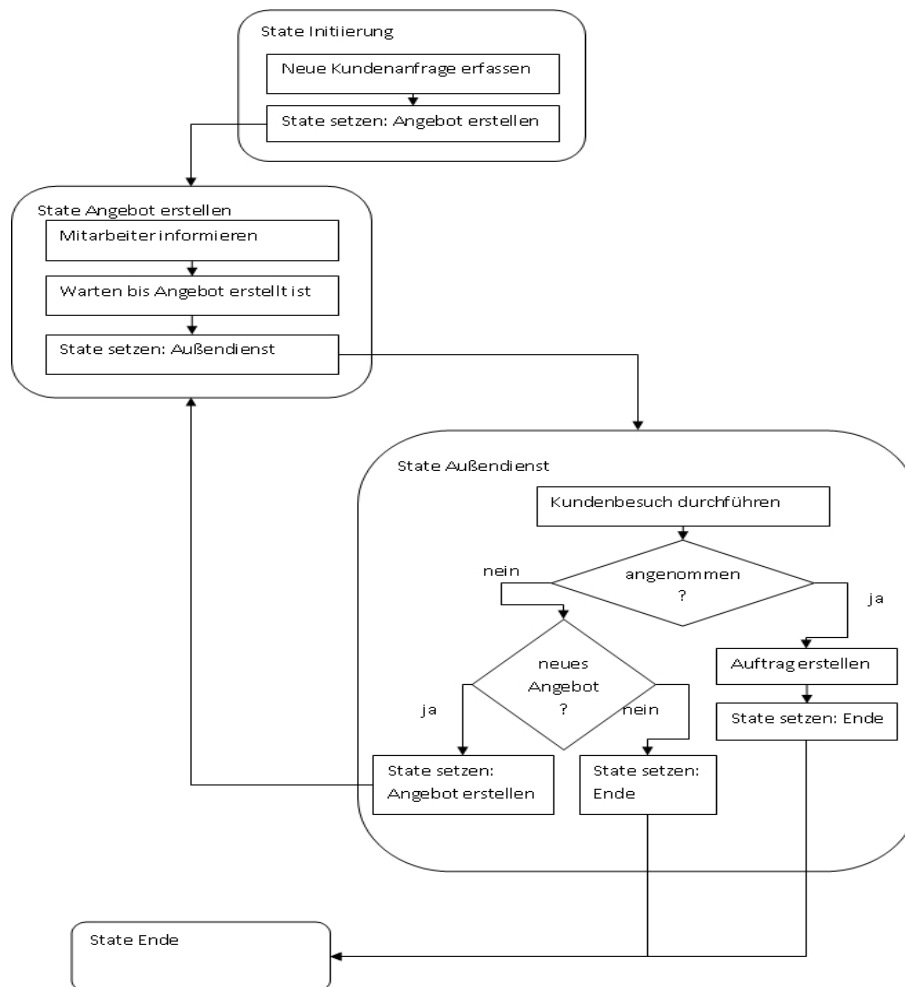


Abbildung 3.6: State Machine Workflow (eigene Darstellung)

Neben diesen beiden Workflowarten lassen sich auch zwei Arbeitsbereiche (Scopes) unterscheiden. Ein Listenworkflow wird einer bestimmten Liste, Bibliothek oder einem Content Type zugewiesen. Die Instanzen des Workflows sind einem bestimmten Element aus der Liste oder der Bibliothek zugewiesen. Der Workflow kann entweder manuell, automatisch beim Erstellen oder automatisch beim Ändern eines Elementes gestartet werden <sup>76</sup>.

<sup>76</sup>Andrew [2009]

Ein Site Workflow wird einer Website (Objekt SPWeb) zugewiesen. Sie können nur manuell gestartet werden <sup>77</sup>. Site Workflows sind nur über das Site Settings Menü einer Website erreichbar (Anhang A.13). Da das Site Settings Menü nur von Website-Administratoren aufgerufen werden kann und Site Workflows manuell gestartet werden müssen, ergibt sich als Hauptanwendungsbereich die Durchführung administrativer Aufgaben.

Die Zuordnung einer Workflowvorlage zu einer Liste oder Seite erfolgt mittels eines Workflow Association Objektes. In diesem Objekt werden alle Parameter der Workflowzuordnung (etwa die Startbedingungen) gespeichert <sup>78</sup>.

Bei näherer Betrachtung des Workflow Objektmodells ist aufgefallen, dass jedes Objekt, dem ein Workflow zugeordnet werden kann, über eine Sammlung von Workflow Association Objekten verfügt. Mehrere Workflowvorlagen können also einer Liste hinzugefügt werden. Es kann auch mehrmals die selbe Workflowvorlage einer Liste hinzugefügt werden. Dies bietet sich an, wenn ein Workflow mit unterschiedlichen Parametern ausgeführt werden soll. Es kann aber stets nur eine Instanz einer Workflowzuordnung pro Listenelement ausgeführt werden. Eine neue Instanz lässt sich erst erzeugen, wenn die vorherige Instanz beendet wurde.

Der Lebenszyklus eines Workflows besteht im Wesentlichen aus vier Schritten, welche über entsprechende Formulare parametrisiert werden können <sup>79 80</sup>.

1. Association: Wird ein Workflow erstmalig zugewiesen, wird ein neues Association Objekt erstellt. Hier werden die Startbedingungen, der Name und die Aufgaben- sowie Verlaufsliste des Workflows definiert. Über ein Association Form lassen sich weitere Parameter erfassen. Das Association Form wird auch angezeigt, wenn eine bestehende Workflowzuordnung geändert wird.
2. Initiation: Wird eine neue Workflowinstanz manuell erzeugt, wird ein Initiation Form angezeigt. In diesem Formular kann der Benutzer, der den Workflow starten will, weitere Parameter angeben oder die Standardwerte aus dem Association Objekt anpassen.
3. Modification: Wird eine bestehende Workflow Instanz verändert, wird ein Modification Form angezeigt.
4. Task: Werden während der Ausführung des Workflows weitere Benutzereingaben benötigt, muss eine Aufgabe (Task) erstellt und einem Benutzer zugewiesen werden. Wenn der Benutzer diese Aufgabe bearbeitet, wird ein

---

<sup>77</sup>Andrew [2009]

<sup>78</sup>MSDN [2010n]

<sup>79</sup>TechNet [2010d]

<sup>80</sup>Krause u. a. [2010e]



Task Form angezeigt, in dem er die benötigten Daten erfassen kann.

Zum Entwickeln von Workflows stehen drei Werkzeuge zur Verfügung.

Mit Visio 2010 Premium können SharePoint Workflowmodelle definiert werden. Das Modell wird aus einzelnen Shapes zusammengesetzt. Es kann hier nur das Skelett des Modells und kein vollständiger Workflow erstellt werden. Das Modell muss anschließend mit dem SharePoint Designer weiter bearbeitet und mit Logik gefüllt werden. Es werden nur sequenzielle Workflows unterstützt<sup>81</sup>.

Mit dem SharePoint Designer lassen sich deklarative Workflows erstellen. Diese werden entweder direkt einer Liste oder Seite hinzugefügt. Sie können aber auch als sogenannter Reusable Workflow einem Content Type zugewiesen werden. Nur diese lassen sich exportieren oder als Vorlage speichern. Mit dem SharePoint Designer können nur sequenzielle Workflows erstellt werden<sup>82</sup>. Deklarative Workflows bestehen aus Aktionen und Bedingungen, welche ohne Programmcode erstellt werden. Welche Aktivitäten verfügbar sind, kann der Administrator in der web.config der Web Application festlegen. Es können eigene Aktivitäten mit Visual Studio 2010 erstellt werden<sup>83</sup>. Mit dem SharePoint Designer lassen sich nur Initiation- und Task Forms erstellen. Andere Formulare werden nicht unterstützt. Ein deklarativer Workflow wird immer mit den Berechtigungen des Benutzers ausgeführt, der die Instanz gestartet hat. Durch Impersonation innerhalb des Workflows können die Rechte des Workflowerstellers genutzt werden<sup>84</sup>.

Beim Entwickeln eines deklarativen Workflows mit dem SharePoint Designer fällt zunächst auf, dass zur Darstellung der Ablauffolge kein Flussdiagramm, sondern eingerückte Sätze verwendet werden. Bei der Prüfung der verfügbaren Workflow Aktivitäten wurde festgestellt, dass es keine gibt, mit der externe Systeme (z.B. mittels WebServices) angebunden werden können. Der Zugriff auf externe Daten ist nur dann möglich, wenn mittels Business Connectivity Service eine Liste mit externen Daten erstellt wird. Darüber hinaus ist es nicht möglich, mit der Copy List Item Aktivität ein Listenelement zu kopieren, wenn sich die Zielliste in einer anderen Site Collection befindet. All diese Einschränkungen lassen sich damit erklären, dass deklarative Workflows in einer SharePoint Online Umgebung oft die einzige Möglichkeit sind, um Workflows zu erstellen. Werden weitere Funktionen benötigt, können die Anforderungen nur mit Visual Studio 2010 realisiert werden.

Visual Studio 2010 bietet den größten Funktionsumfang bei der Workflow Entwicklung. Es können sequenzielle und State Machine Workflows erstellt werden.

---

<sup>81</sup>Posey [2010]

<sup>82</sup>Krause u. a. [2010f]

<sup>83</sup>TechNet [2010d]

<sup>84</sup>Krause u. a. [2010f]

Sie können nur als Farm Lösung bereitgestellt werden. In einer SharePoint Online Umgebung können so erstellte Workflows in der Regel nicht verwendet werden. Visual Studio 2010 verfügt über Vorlagen für Association und Initiation Formulare. Mit diesen Vorlagen lassen sich entsprechende ASPX Seiten erstellen. Sollen Task oder Modification Formulare genutzt werden, müssen diese selbst entwickelt werden. Alternativ kann man mit InfoPath alle Arten von Workflow Formularen erstellen. Zum Datentransport dienen XML Dateien, die in InfoPath als sekundäre Datenquelle eingebunden werden <sup>85</sup>. Ein mit Visual Studio 2010 erstellter Workflow wird stets mit der Identität des SharePoint Applikationspool ausgeführt.

Bei der Zuordnung eines Workflows an eine Liste oder Seite muss angegeben werden, ob vorhandene Aufgaben- und Verlaufslisten verwendet oder ob die Listen neu erstellt werden sollen. In der Aufgabenliste werden alle von den Workflow Instanzen erzeugten Aufgaben abgelegt. In der Verlaufsliste können Meldungen des Workflows gespeichert werden.

Zur genaueren Analyse dieser Listen wurde zunächst ermittelt, wo diese abgelegt sind. Sie befinden sich direkt auf der Ebene der Website (Objekt SPWeb) und werden zusammen mit allen anderen Listen der Website angezeigt. Bei genauerer Betrachtung der hier verwendeten Listen stellt man fest, dass die Berechtigungen der übergeordneten Website vererbt werden. Benutzer mit Schreibrechten können die Datensätze, wie bei anderen Listen, manipulieren. Sie sind somit als Überwachungspfad ungeeignet. Es muss darauf geachtet werden, keine vertraulichen Informationen in den verwendeten Listen zu speichern. Andernfalls können Benutzer mit Leserechten für die Aufgaben- oder Verlaufsliste an sensible Daten gelangen. Diese Gefahr besteht sowohl für deklarative als auch für programmierte Workflows. Sie lässt sich nur begrenzen, indem keine vertraulichen Daten in den Listen gespeichert werden oder indem neue Aufgaben- und Verlaufslisten erstellt werden. Die Berechtigungen auf die neuen Listen können anschließend angepasst werden.

In den folgenden Abschnitten werden einige Anwendungsfälle skizziert und Hinweise zu deren Implementierung gegeben.

**Anwendungsfall 1: Benachrichtigung bei Änderung eines Feldwertes** In einer Dokumentenbibliothek werden Publikationen des Unternehmens gesammelt. Jedes Dokument verfügt über einen Veröffentlichungsstatus, welcher die Werte Entwurf, Öffentlich und Archiv annehmen kann. Ein Mitarbeiter der Öffentlichkeitsarbeit möchte darüber informiert werden, wenn der Status eines Dokumentes ge-

---

<sup>85</sup>Krause u. a. [2010g]

ändert wurde. Die Alert Funktion der Dokumentenbibliothek kann die Anforderung nicht bedienen, da der Benutzer informiert wird, sobald sich irgendein Feld oder das Dokument ändert.

Zunächst wird in der Dokumentenbibliothek eine neue Spalte erzeugt, die den alten Status des Dokumentes beinhaltet. Dies ist erforderlich, da im SharePoint Designer nur auf die aktuellen Feldwerte zugegriffen werden kann und der Workflow nach der Änderung startet. Anschließend wird die Webseite mit der Dokumentenbibliothek im SharePoint Designer geöffnet und ein neuer Workflow vom Typ Listenworkflow erstellt. In der Design Ansicht werden nun die einzelnen Aktivitäten in Form von Sätzen abgelegt. Zunächst wird geprüft, ob sich die Feldwerte Veröffentlichungsstatus und Veröffentlichungsstatus\_alt unterscheiden. Zudem wird geprüft, ob der Veröffentlichungsstatus ungleich Entwurf ist. Bei Bedarf wird der Benutzer mit einer E-Mail informiert. Anschließend wird das Feld Veröffentlichungsstatus\_alt auf den Wert von Veröffentlichungsstatus geändert. In der Verlaufsliste des Workflows wird protokolliert, dass der Benutzer informiert wurde. Vor der Veröffentlichung des Workflows, werden die Startbedingungen in den Workfloweinstellungen angepasst, damit der Workflow gestartet wird, wenn sich ein Element ändert (Abbildung 3.7).

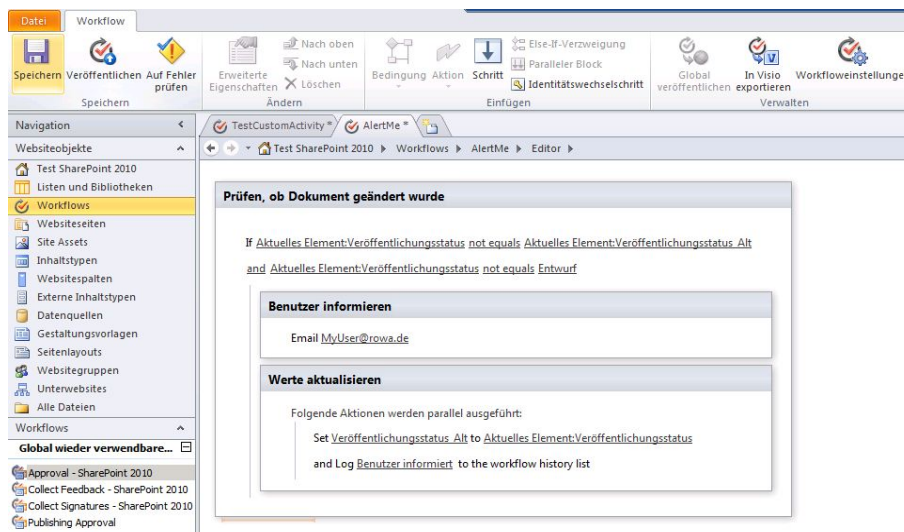


Abbildung 3.7: Workflow Definition im SharePoint Designer (eigene Darstellung)

**Anwendungsfall 2: Verbesserungsvorschlag** Die Mitarbeiter des Unternehmens sollen zukünftig ihre Verbesserungsvorschläge über das Unternehmensportal einreichen können. Die eingegangenen Vorschläge werden je nach Inhalt des Vorschlags an verschiedene Mitarbeiter zur Prüfung übergeben. Diese Mitarbeiter müssen entscheiden, ob die Vorschläge umgesetzt werden und ob der Mitarbeiter eine Prämie erhält.

Um dies zu realisieren, wird eine neue SharePoint Liste mit allen notwendigen Spalten angelegt, in der die Verbesserungsvorschläge gespeichert werden. Anschließend wird die Webseite mit der Liste im SharePoint Designer geöffnet und ein neuer Listenworkflow erzeugt. In der Design Ansicht werden alle notwendigen Workflowschritte definiert. Zunächst wird eine neue Aufgabe erstellt, mit der der Mitarbeiter bestimmt werden kann, der den Vorschlag bewerten soll. Die Entscheidung, welcher Mitarbeiter den Vorschlag bewerten soll, wird von einem Mitarbeiter der Personalabteilung getroffen. Die vom Benutzer einzugebenden Daten (Kommentar, Umsetzen, Prämie) werden mit dem Task Wizard definiert (Anhang A.14). Um die eingegebenen Daten weiter verarbeiten zu können, müssen sie aus der Aufgabenliste ausgelesen werden. Hat der Benutzer die Aufgabe abgeschlossen, wird geprüft, ob der Vorschlag umgesetzt werden soll. Wird er umgesetzt, wird weiter geprüft, ob eine Prämie gezahlt wird. Der vorschlagende Mitarbeiter wird über das Ergebnis der Bewertung per E-Mail informiert (Abbildung 3.8). Der Workflow startet, sobald ein neues Element der Liste hinzugefügt

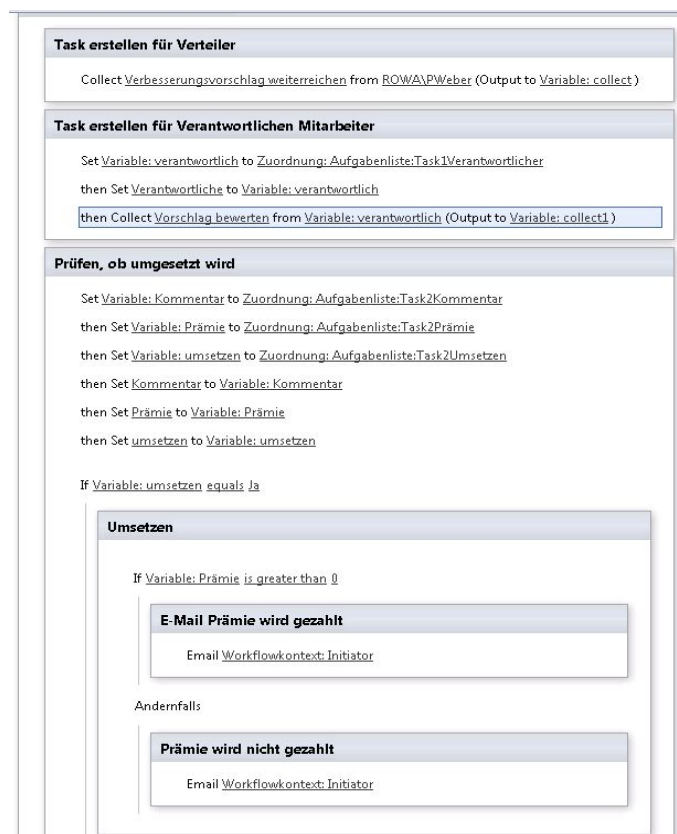


Abbildung 3.8: Definition des Workflows zum Einreichen von Verbesserungsvorschlägen (eigene Darstellung)

wird. Nun können die automatisch generierten Task Formulare aus dem SharePoint Designer heraus mit InfoPath bearbeitet werden. Damit der vorschlagende Benutzer die Felder „Prämie“ und „umsetzen“ nicht verändern kann, werden die Standardformulare zum Anzeigen und Editieren der Liste geändert und diese Fel-

der ausgeblendet. Sie sind somit nur noch über den Workflow anpassbar, werden in der Liste aber nach wie vor angezeigt.

**Anwendungsfall 3: Personalbeschaffung als State Machine** Das Unternehmen ist häufig auf der Suche nach neuen Mitarbeitern. Wenn Personalbedarf besteht, teilt dies der zuständige Abteilungsleiter der Personalabteilung mit. Diese unterrichtet darauf hin die Marketing Abteilung, dass Stellenanzeigen geschaltet werden müssen. Die Personalabteilung prüft nun zusammen mit dem Abteilungsleiter die eingegangenen Bewerbungen. Ist kein geeigneter Bewerber gefunden, werden erneut Stellenanzeigen geschaltet. Der Prozess wiederholt sich, bis ein neuer, geeigneter Mitarbeiter eingestellt wurde oder die Suche aufgegeben wird. Der Prozess soll in SharePoint durch einen Workflow abgebildet werden. Die Personalanforderung wird in einer SharePoint Liste gespeichert. In dieser Liste werden auch die Beschreibung der Stelle und das Anforderungsprofil definiert. Mit einem Association Form des Workflows werden die E-Mail Adressen der Ansprechpartner der Personal- und Marketingabteilung bestimmt. Es werden zwei Aufgaben definiert. Die erste Aufgabe wird von der Personalabteilung bearbeitet. Hier kann die Anforderung einer Stellenanzeige erstellt oder der Workflow beendet werden (Anhang A.12). Die zweite Aufgabe wird von der Marketing Abteilung bearbeitet und dient zur Rückmeldung bei geschalteter Stellenanzeige. Beide Formulare werden als InfoPath Formulare erstellt. In einem leeren SharePoint Projekt wird ein Element vom Typ Zustandsautomaten Workflow (State Machine) hinzugefügt. Der Workflow wird als Listenworkflow erstellt. Es werden insgesamt vier Zustände definiert. Neben einem Start- und einem Endzustand wird für die Personalabteilung und das Marketing je ein eigener Zustand definiert (Anhang A.13). Der Workflow beginnt immer mit dem Initialen Zustand. Hier ist bereits eine Aktivität vom Typ OnWorkflowActivated definiert. In der dazugehörigen Methode werden die Daten des Association Form gelesen. Das Formular kann in Visual Studio 2010 über eine Vorlage erstellt werden. Es wird eine neue ASPX Seite erstellt, in der alle notwendigen Schritte zur Assoziation des Workflows an die Liste enthalten sind. Der Entwickler muss nur die Methode GetAssociationData implementieren, in der die Formulardaten ausgelesen und als XML String zurückgegeben werden. Der Quellcode des Association Form befindet sich in Listing A.16. Sind alle Daten des Association Form eingelesen, wird der Zustand des Workflows auf den Zustand der Personalabteilung geändert (Anhang A.14). In diesem Zustand wird eine neue Aufgabe erstellt. Damit die Aufgabe eindeutig identifiziert werden kann, wird ein neues Correlation Token definiert. In der Methode der CreateTask Aktivität wird festgelegt, dass der Mitarbeiter der Personalabteilung die Aufgabe bearbeiten soll. Außerdem werden die Werte für das Task Formu-

lar gefüllt. Über das Attribut `TaskType` wird angegeben, welches Formular für die Aufgabe verwendet werden soll. Hat der Mitarbeiter die Aufgabe abgeschlossen, wird geprüft, ob eine neue Stellenanzeige geschaltet werden soll. Ist dies der Fall, ändert sich der Zustand des Workflows auf den Zustand der Marketingabteilung, sonst auf den finalen Zustand (Anhang A.15). Für den Mitarbeiter der Marketingabteilung wird ebenfalls eine Aufgabe erstellt. Hat er diesen bearbeitet, ändert sich der Zustand des Workflows wieder auf den Zustand der Personalabteilung. Damit die Werte des Workflows im Formular angezeigt werden, wird eine `ItemMetadata.xml` für jedes Formular erstellt. In dieser XML Datei sind alle Feldnamen der Parameter der Aufgabe enthalten. Wichtig ist hier, dass die Feldnamen mit `ows_` beginnen (Listing L.3).

```
1 <z:row xmlns:z="#RowsetSchema"
2   ows_Abteilung=" "
3   ows_Stelle=" "
4   ows_Stellenbeschreibung=" "
5   ows_Anforderungen=" " />
```

Listing L.3: ItemMetadata.xml

Die XML Datei wird als sekundäre Datenquelle dem Formular hinzugefügt. Um die InfoPath Formulare zusammen mit dem Workflow bereitzustellen, werden diese dem Visual Studio 2010 Projekt hinzugefügt. Der Bereitstellungstyp wird auf `ElementFile` festgelegt. Zum Abschluss wird die `Elements.xml` des Workflows angepasst. Hier werden die Uniform Resource Name (URN) der Formulare angegeben und der Inhaltstyp der Aufgabenliste angepasst (Listing A.17).

Sollen die Task Formulare als ASPX Seiten erstellt werden, steigt der Implementierungsaufwand erheblich. Zunächst wird ein neuer Content Type erstellt. In diesem werden alle Felder der Aufgabe sowie die Formularseiten definiert. Hier ist darauf zu achten, dass die ID des neuen Content Types mit `0x01080100` beginnt und somit von dem Workflow Task Content Type erbt. Dieser Content Type muss der Aufgabenliste des Workflows hinzugefügt werden. Eine Definition des Content Type befindet sich im Listing A.18. Die Verbindung zu der aktuellen Workflowinstanz wird dabei durch URL Parameter gesichert. Nach Auslesen der Parameter kann die zur Workflowinstanz gehörende Aufgabenliste, das Element der Aufgabenliste sowie die Workflowinstanz selbst ermittelt werden.

Um Benutzereingaben zu speichern, wird eine neue Hashtabelle erstellt. Diese Tabelle wird mit allen Werten gefüllt, die der Benutzer erfasst hat. Im Code des Workflows können diese Werte aus den `ExtendedProperties` der Aufgabe ausgelesen werden. Sind alle Daten gesichert, wird mit der statischen Methode `AlterTask` der Klasse `SPWorkflowTask` die zugehörige Aufgabe geändert. Danach wird

der Benutzer zu der ursprünglichen Seite weiter geleitet. Eine vollständige Definition des Task Formulars für den Mitarbeiter der Personalabteilung befindet sich im Listing A.19. Im Code des Workflows wird die Aktivität `CreateTaskWithContentType` genutzt, um eine neue Aufgabe mit der Id des Content Type zu erstellen, der zuvor definiert wurde. In der zugehörigen Methode muss zunächst geprüft werden, ob die Aufgabenliste des Workflows Content Types unterstützt. Ist dies sichergestellt, wird geprüft, ob der Content Type bereits der Liste hinzugefügt wurde. Bei Bedarf wird er der Aufgabenliste angefügt (Listing A.20).

**Alternative zu Code Activity: Custom Activity als Sandbox** Bei der Entwicklung von deklarativen Workflows stehen nur die vorgegebenen Workflow Aktivitäten zur Verfügung. Reichen diese nicht aus, kann der Workflow exportiert und mit Visual Studio 2010 weiter bearbeitet werden. Der Entwickler kann hier die Code Activity nutzen, um einen beliebigen Programmcode auszuführen. Ein so erstellter Workflow kann nur als Farm Lösung bereitgestellt werden, was die Verwendung in einer SharePoint Online Umgebung ausschließt. Die erstellte Code Activity kann auch nicht in deklarativen Workflows genutzt werden.

Eigene, wiederverwendbare Workflow Funktionen lassen sich als Custom Activities realisieren, welche als Sandbox bereitgestellt werden können. Zunächst wird eine neue Klasse erstellt, welche die gewünschte Funktion als öffentliche Methode enthält. Diese Methode muss als Parameter ein Objekt vom Typ `SPUserCodeWorkflowContext` erwarten. Weitere Parameter können hinzugefügt werden. Der Rückgabeparameter muss vom Typ `Hashtable` sein.

Im folgenden Beispiel soll eine Custom Workflow Activity erstellt werden, welche mit einem Collaborative Application Markup Language (CAML) Ausdruck bestimmte Elemente einer Liste ausliest und die ID des ersten Elements zurück gibt. Mit dieser Aktivität können aus einem deklarativen Workflow heraus komplexe Abfragen auf eine Liste angewendet werden. Mit der zurückgegebenen ID kann anschließend im Workflow auf das Element zugegriffen werden. Im Code wird zunächst mit dem `SPUserCodeWorkflowContext` Objekt auf die übergebene SharePoint Liste zugegriffen. Anschließend wird der CAML Ausdruck auf die Liste angewendet. Wurden Elemente gefunden, wird die ID des ersten Elements zurückgegeben, andernfalls -1. Der vollständige Quellcode der Custom Activity befindet sich im Anhang A.22.

Damit die Klasse als Custom Activity genutzt werden kann, wird dem Projekt ein SharePoint Modul hinzugefügt. In der `Elements.xml` des Moduls wird die Schnittstelle zur Aktivität definiert. Alle Aktivitäten der Klasse werden in einem eigenem `<Action>` Element unterhalb des `<WorkflowActions>` Element festgelegt. Im Element `RuleDesigner` wird das Aussehen der Aktivität im SharePoint Designer defi-

niert. In dem <Parameters> Element werden alle Parameter der Aktivität definiert. Wichtig ist hier, dass als erster Parameter das Kontext Objekt angegeben wird. Dieses muss zwingend den Namen \_\_Context tragen. Die vollständige Schnittstellendefinition befindet sich im Anhang A.21. Nachdem die Aktivität bereitgestellt wurde, kann diese im SharePoint Designer unter den Aktivitäten ausgewählt werden (Abbildung 3.9).

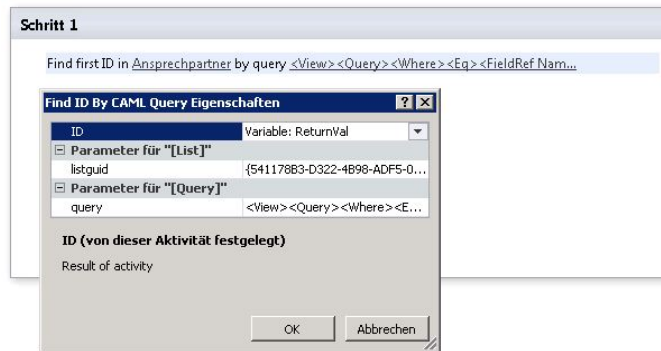


Abbildung 3.9: Eigenschaften der CustomActivity (eigene Darstellung)

**Empfehlungen** Workflows sind der bevorzugte Weg zur Abbildung von Geschäftsprozessen. Dies gilt auch dann, wenn keine Benutzerinteraktion erforderlich ist. Die folgenden Tabellen dienen als Entscheidungsgrundlage für zukünftige Projekte. In Tabelle 6 wird dargestellt, welche Möglichkeiten Workflows in den einzelnen SharePoint Versionen bieten. Tabelle 7 beinhaltet einen Vergleich der Entwicklungswerkzeuge. Tabelle 8 enthält allgemeine Handlungsempfehlungen beim Entwickeln von Workflows

Funktion	SharePoint Foundation 2010	SharePoint Server 2010	SharePoint Server 2010 Enterprise
Nutzung von Visio Services	-	-	•
Nutzung von InfoPath	-	-	•
Sequenzieller Workflow	•	•	•
State Machine Workflow	•	•	•

- wird unterstützt
- wird mit Einschränkung unterstützt
- wird nicht unterstützt

Tabelle 6: Workflow Funktionen nach SharePoint Versionen (eigene Darstellung)



Funktion	SharePoint Designer	Visual Studio
Sequenzieller Workflow	•	•
State Machine Workflow	•	•
Custom Activity	○ (nur Nutzung, keine Entwicklung)	•
Association Formular	-	•
Initiation Formular	•	•
Modification Formular	-	•
Task Formular	•	•
Impersonation (Wechsel des Benutzerkontext)	○  (nur Autor und Initiator)	•
Anbindung externer Systeme	○ (nur mit BCS als externe Liste)	•

- wird unterstützt
- wird mit Einschränkung unterstützt
- wird nicht unterstützt

Tabelle 7: Vergleich der Entwicklungswerkzeuge für Workflows (eigene Darstellung)

Anforderung	Empfehlung	Alternative
Sequenzieller Workflow	verwenden bei <ul style="list-style-type: none"> <li>• einfachen Modellen</li> <li>• wenigen/keine Wiederholungen</li> <li>• wenige/keine Verzweigungen</li> <li>• SharePoint Online Umgebungen</li> </ul>	State Machine Workflow oder Funktion in Custom Activity zusammenfassen
State Machine Workflow	verwenden bei <ul style="list-style-type: none"> <li>• komplexen Modellen</li> <li>• vielen Wiederholungen</li> <li>• vielen Verzweigungen</li> <li>• lokaler SharePoint Installation</li> </ul>	Sequenzieller Workflow oder Funktion in Custom Activity zusammenfassen

Listenworkflows	wenn <ul style="list-style-type: none"><li>• Workflow von Listenelementen abhängig ist</li><li>• Benutzer den Workflow starten sollen</li><li>• automatischer Start benötigt wird</li></ul>	Event Receiver für Liste
Site Workflows	nur wenn Administrator den Workflow starten soll	Timer Job oder Anwendung als geplanter Task
Custom Activity	wenn <ul style="list-style-type: none"><li>• Aktivitäten des SharePoint Designers nicht ausreichen</li><li>• Code wiederverwendet werden soll</li></ul>	Mit Visual Studio erstellter Workflow mit Code Activity

Tabelle 8: Handlungsempfehlungen beim Entwickeln von Workflows (eigene Darstellung)

### 3.3.4 Windows Communication Services

Die Windows Communication Foundation ist eine serviceorientierte Plattform zur Kommunikation von verteilten Anwendungen. Alle bekannten Microsoft Technologien zum Erstellen verteilter Anwendungen, wie etwa .NET Remoting, Microsoft Message Queue (MSMQ) oder ASP.NET Web Services, stehen unter einem einheitlichen Programmiermodell zur Verfügung. Ein Service ist eine Sammlung von Operationen, welche eine funktionale Einheit bilden. Er kann von verschiedenen Anwendungen konsumiert werden <sup>86</sup>. Ein WCF Dienst kann von beliebigen Applikationen gehostet werden <sup>87</sup>. Im Folgenden werden nur die Besonderheiten im Zusammenhang mit einem Hosting durch SharePoint betrachtet.

SharePoint bietet bereits eine umfassende Sammlung an Windows Communication Foundation (WCF) Diensten. Zudem sind ASP.NET Web Services der vorherigen SharePoint Version weiterhin verfügbar. Alle verfügbaren Dienste befinden sich im ISAPI Verzeichnis der SharePoint Installation. Dieses kann über die URL einer beliebigen SharePoint Seite als Verzeichnis `_vti_bin` erreicht werden <sup>88</sup>. Über die URL `http://<My SharePoint Site>/_vti_bin/Lists.asmx` kann beispielsweise ein Dienst zum Verwalten und Lesen von Listen erreicht werden. Neben den bereits verfügbaren Diensten können auch eigene WCF Dienste erstellt und von SharePoint gehostet werden <sup>89</sup>. Bevor ein Service genutzt werden kann, muss zunächst ein Endpunkt definiert werden. Über den Endpunkt kann eine Applikation mit dem Service kommunizieren. Ein Endpunkt besteht aus drei Teilen <sup>90</sup>:

- Address: Die Adresse ist die URI des Dienstes.
- Binding: Das Binding gibt an, welche Protokolle, Kodierungen und Sicherheitseinstellungen zur Kommunikation genutzt werden.
- Contract: Im Contract wird definiert, über welche Funktionen ein Dienst verfügt. Benutzerdefinierte Datentypen für Parameter oder Rückgabewerte bilden den DataContract. Die bereitgestellten Funktionen bilden den ServiceContract.

Bei der Entwicklung von eigenen Diensten, die von SharePoint gehostet werden sollen, muss der Entwickler keine Endpunkte definieren. SharePoint liefert eine Service Factory, die alle notwendigen Endpunkte zur Laufzeit erzeugt <sup>91</sup>. Share-

---

<sup>86</sup>Kotz u. Hölzl [2009]

<sup>87</sup>Peiris u. Mudler [2007]

<sup>88</sup>MSDN [2010k]

<sup>89</sup>MSDN [2010k]

<sup>90</sup>Kotz u. Hölzl [2009]

<sup>91</sup>MSDN [2010k]

Point liefert bereits eine Vielzahl an Diensten und Methoden, welche die wichtigsten Operationen bereitstellen. Dies zeigt auch das Client Objekt Modell von SharePoint. Hierbei handelt es sich ebenfalls um einen WCF Dienst (client.svc)<sup>92</sup>. Bevor ein neuer WCF Dienst erstellt wird, sollte geprüft werden, ob die benötigte Funktion nicht mit vorhandenen Diensten realisiert werden kann. WCF Dienste lassen sich nur als Farm Lösung bereitstellen, die Bereitstellung in SharePoint Online Umgebungen ist in der Regel nicht möglich.

Um einen Dienst aus einer .NET Anwendung heraus nutzen zu können, wird in Visual Studio 2010 ein Dienstverweis hinzugefügt. Hiermit wird automatisch eine Proxy Klasse mithilfe der Metadaten des Dienstes erstellt. Der Dienst lässt sich anschließend wie eine lokale Klasse bedienen. Um ihn von einer anderen Plattform wie Java aufrufen zu können, wird eine einfache HTTP Anfrage mit der Nachricht als XML an die Adresse des Dienstes geschickt.

Bei der Verwendung der SharePoint Dienste ist eine Besonderheit aufgefallen. Soll ein Dienst von einem WebPart aufgerufen werden, erstellt Visual Studio 2010 beim Hinzufügen des Dienstverweises automatisch eine app.config Datei, in der die Endpunkte des Dienstes definiert sind. Nach dem Bereitstellen des WebParts kann dieser aber nicht den Dienst nutzen. Statt dessen muss der Endpunkt in der web.config von SharePoint abgelegt werden. Dies führt schnell zu einer umfangreichen Konfigurationsdatei. Hier empfiehlt es sich, eine eigene web.config für den WebPart zu erstellen. Der IIS fasst zur Laufzeit alle Konfigurationsdateien zu einer globalen Konfiguration zusammen. Eine direkte Anpassung der SharePoint Konfigurationsdatei kann also umgangen werden. Die hier beschriebene Vorgehensweise gilt für alle Eigenentwicklungen, die direkt auf dem Server ausgeführt werden (z.B. Workflows, ASPX Seiten, EventReceiver etc.).

Die SharePoint Foundation 2010 liefert auch einen REST Dienst, mit dem sich Listenelemente direkt per URL adressieren und manipulieren lassen. Durch die Übergabe von URL Parametern können komplexe Abfragen erstellt werden. Der Entwickler muss keine SOAP Nachrichten erzeugen. Die Rückgabe erfolgt als Atom Publishing Protocol (AtomPub) oder als JavaScript Object Notation (JSON). Zum Testen einer REST Abfrage genügt ein einfacher Browser<sup>93</sup>.

Bei der Prüfung der Möglichkeiten, welche die REST Schnittstellen bieten, ist ein Problem im Zusammenhang mit dem REST Listendienst listdata.svc aufgefallen. Wird der REST Dienst als Dienstverweis in Visual Studio 2010 hinzugefügt, wird

---

<sup>92</sup>Driscoll u. a. [2010]

<sup>93</sup>MSDN [2010h]

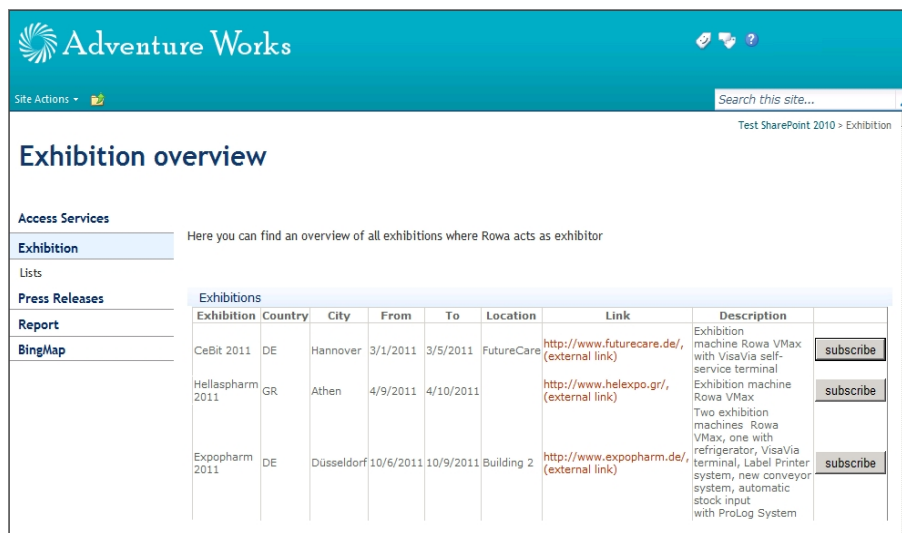
automatisch eine Proxy Klasse erstellt, welche eine Definition für alle in der Website enthaltenen Listen und Spalten enthält. Dies schlägt allerdings fehl, wenn in irgendeiner Liste der Website im Listennamen oder in den Spaltenbezeichnungen Umlaute enthalten sind. Der Dienstverweis lässt sich auch nach Korrektur der betroffenen Namen nicht aktualisieren oder löschen. Um ihn zu entfernen, muss das Projekt geschlossen und das Service Reference Verzeichnis innerhalb des Projektorders manuell gelöscht werden. Da die Proxy Klasse alle Listen und Spalten der Website enthält, wird eine Portierung auf andere Systeme erschwert. Diese gelingt nur, wenn auf dem Zielsystem die gleichen Listen und Spalten angelegt sind. Wenn also nicht sichergestellt ist, dass das Zielsystem die gleiche Struktur aufweist, sollte in einer .NET Anwendung ein anderer Dienst zum Bearbeiten von Listen genutzt werden oder die REST Abfrage als einfache http Anfrage gesendet werden.

Bei der Entwicklung eines WCF Dienstes für SharePoint fällt zudem auf, dass Visual Studio 2010 hierfür keine Projektvorlage zur Verfügung stellt. Um den Dienst als WSP Paket bereitstellen zu können, muss dieser in einem leeren SharePoint Projekt implementiert werden.

In den folgenden Abschnitten werden einige Anwendungsfälle skizziert und Hinweise zu deren Implementierung gegeben.

**Anwendungsfall 1: Datenmanipulation per REST Schnittstelle** Das Unternehmen präsentiert seine Produkte regelmäßig auf internationalen und nationalen Fachmessen. Die Planung der Messen und des Equipments erfolgt über die Marketingabteilung mit einer SharePoint Liste im Intranet. Bislang wurde die Information, auf welchen Messen das Unternehmen mit welchen Produkten anwesend ist, auf Anfrage an die Vertriebspartner weitergereicht. Dies führt dazu, dass die Partner veraltete Informationen nutzen. Zukünftig sollen sich die Partner im Extranet über alle Messeauftritte informieren. Der Vertriebspartner soll über einen Button weitere Informationen zur Messe anfordern können. Alle Anfragen sollen in einer Liste gesammelt werden und vor der Messe zur Terminabsprache dienen. Um diese Anforderung bedienen zu können, soll die REST Schnittstelle von SharePoint genutzt werden. Mit ihr werden die Messetermine direkt aus dem Intranet gelesen und im Extranet präsentiert. Die Daten werden in einem neuen Visual WebPart in einem GridView Control ausgegeben. Um die Daten per REST auslesen zu können, wird im Projekt ein Verweis auf den listdata.svc Dienst hinzugefügt. Visual Studio 2010 erstellt daraufhin eine Proxy Klasse für die REST Schnittstelle. In dieser Klasse sind alle Listen und deren Attribute der Website

enthalten. In der Methode OnInit des WebParts wird zunächst der DataContext des Dienstes erstellt. Über ihn sind alle Tabellen und deren Felder erreichbar. Anschließend werden in der Methode OnLoad mittels Linq Ausdruck alle Termine, die im Extranet angezeigt werden sollen, ausgelesen. Die ausgelesenen Daten werden anschließend als Datenquelle dem Grid übergeben. Im Grid wird pro Datensatz ein Button ausgegeben. Der Benutzer kann hiermit weitere Informationen anfordern. Alle Anfragen werden in einer neuen SharePoint Liste gespeichert. Dazu wird zunächst ein neues Listenelement über die REST Schnittstelle erzeugt. Danach können die Felder des Elementes gefüllt werden. Die Mitarbeiter des Unternehmens werden über neue Einträge informiert und treten mit dem Partner in Kontakt, um die Art der benötigten Informationen zu bestimmen (Abbildung 3.10). Da die Mitarbeiter des Partners keinen Zugriff auf das Intranet haben, werden die



Exhibition	Country	City	From	To	Location	Link	Description	
CeBit 2011	DE	Hannover	3/1/2011	3/5/2011	FutureCare	<a href="http://www.futurecare.de/">http://www.futurecare.de/</a> (external link)	Exhibition machine Rowa VMax with VisaVia self-service terminal	subscribe
Hellaspharm 2011	GR	Athen	4/9/2011	4/10/2011		<a href="http://www.helexpo.gr/">http://www.helexpo.gr/</a> (external link)	Exhibition machine Rowa VMax	subscribe
Expopharm 2011	DE	Düsseldorf	10/6/2011	10/9/2011	Building 2	<a href="http://www.expopharm.de/">http://www.expopharm.de/</a> (external link)	Two exhibition machines Rowa VMax, one with refrigerator, VisaVia terminal, Label Printer system, new conveyor system, automatic stock input with ProLog System	subscribe

Abbildung 3.10: WebPart stellt Daten per REST API dar (eigene Darstellung)

Daten im Kontext des Systemaccounts ausgelesen. Der Quellcode der Anwendung befindet sich im Anhang A.23.

**Anwendungsfall 2: Anlegen neuer Websites** Das Unternehmen stellt seinen Vertriebspartnern ein Extranet zur Verfügung. Über diesen Kanal werden allgemeine und Vertriebsinformationen ausgetauscht. Zukünftig sollen auch Dokumente zu Kundenprojekten ausgetauscht werden. Dies geschieht bislang per E-Mail. Um alle Dokumente zu einem Kundenprojekt an einem Ort vorhalten zu können, soll pro Projekt eine eigene Website im Extranet angelegt werden. Zudem muss darauf geachtet werden, dass nur Mitarbeiter des jeweiligen Partners Zugriff auf das Kundenprojekt erhalten und die Berechtigungsstufen je nach Mitarbeiter unterschiedlich sein können. Zu einem Kunden können mehrere Projekte gehören. Daher soll für jeden Kunden eine eigene Website erstellt werden, in der

pro Projekt eine Sub Website angelegt wird. Der Pfad zu einem Projekt soll Land – > Kunde – > Projekt sein. Die Anforderung für eine neue Website soll über das CRM System und des Rowa Planers gestellt werden können.

Um die Anforderungen erfüllen zu können, wird ein neuer WCF Dienst erstellt, der von SharePoint gehostet wird. SharePoint liefert zwar einen Dienst, mit dem Websites erstellt werden können, Berechtigungen lassen sich aber hiermit nicht setzen. Bei der Anlage des Projektes muss zudem geprüft werden, ob für das Land und den Kunden bereits eine Website angelegt ist oder diese angelegt werden muss. Zunächst wird ein neues leeres SharePoint Projekt erstellt. Anschließend wird eine neue Klasse und eine neue Schnittstellendefinition hinzugefügt. In der Schnittstelle werden die Methode und der Contract des Dienstes beschrieben. Die Implementierung der Funktion erfolgt in der Dienstklasse. Zunächst wird der übergebene Pfad in die einzelnen Websites aufgeteilt. Anschließend wird pro Website geprüft, ob sie angelegt werden muss. Nach Anlage der Website werden die erforderlichen Berechtigungen anhand der übergebenen Berechtigungsliste gesetzt. Konnten die Berechtigungen nicht gesetzt werden, wird dies in dem Rückgabeobjekt vermerkt. Danach wird der Titel der Startseite auf den übergebenen Wert gesetzt. War dies nicht erfolgreich, wird es ebenfalls in dem Rückgabeobjekt vermerkt. Damit der Dienst von SharePoint gehostet werden kann, soll die Service Factory von SharePoint verwendet werden. Zunächst wird dem Projekt ein Verweis auf das ISAPI Verzeichnis der SharePoint Installation hinzugefügt. In diesem wird anschließend eine neue svc Datei angelegt (Anhang A.24). Damit die SharePoint Factory genutzt werden kann, muss im Projekt noch ein Verweis auf das Assembly Microsoft.SharePoint.Client.ServerRuntime eingefügt werden. Nach der Bereitstellung des Projektes kann der Dienst wie gewohnt verwendet werden.

**Anwendungsfall 3: Systemüberwachung** Das Unternehmen setzt zur Überwachung der IT Infrastruktur das Open Source Produkt Nagios ein. Die Portale des Unternehmens können bisher nur unzureichend überwacht werden. Lediglich die Auslastung von CPU- und Speicherressourcen sowie die Erreichbarkeit der Portale kann geprüft werden. Zukünftig sollen auch bestimmte Timer Jobs von SharePoint geprüft werden. Um die Anforderung erfüllen zu können, wird ein neuer WCF Dienst erstellt und von SharePoint gehostet. Die Methode des Dienstes erhält als Übergabeparameter den Namen des zu prüfenden Jobs sowie eine Zeitspanne. Die Methode prüft anschließend in der gesamten Farm, wie oft der Dienst in der Zeitspanne fehlgeschlagen ist. Wird der Job nicht gefunden oder tritt ein anderer Fehler ein, wird -1 zurück gegeben. Damit der Dienst von SharePoint gehostet werden kann, wird er wie im Anwendungsfall 2 beschrieben

bereitgestellt. Der Quellcode des Dienstes befindet sich in Anhang A.25.

**Alternative: WCF Dienst in eigener Web Application** Ein in SharePoint gehosteter Dienst wird immer auf dem Dateisystem des Servers bereitgestellt. Folglich ist es nicht möglich, in einer SharePoint Online Umgebung eigene Dienste bereitzustellen. Die Dienste lassen sich aber auf einem anderen (lokalen) IIS Server bereitstellen. Von hier kann aber nicht mit dem Server Objektmodell, sondern nur mit dem Client Objektmodell des SharePoint gearbeitet werden. Eine Bereitstellung eigener Dienste kann aber dennoch sinnvoll sein, um wiederkehrende Entwicklungsaufgaben zu vermeiden. Im folgendem Beispiel soll ein Dienst erstellt werden, mit dem Dokumente in eine Bibliothek hochgeladen werden können. Zudem soll pro hochgeladenes Dokument festgelegt werden, welche Benutzer welche Berechtigungen für das Dokument haben. Zunächst wird in einem neuen WCF Projekt die Schnittstelle des Dienstes beschrieben. Anschließend erfolgt die Implementierung in der Dienstklasse. Anhand der übergebenen URL wird zuerst ein ClientContext Objekt erzeugt. Mit ihm können weitere Elemente der Website erreicht werden. Danach wird der übergebenen Dokumentenbibliothek die übergebene Datei angefügt. Im Abschluss wird die Berechtigung des neuen Dokumentes angepasst. Der vollständige Quelltext der Anwendung befindet sich im Anhang A.26.

**Empfehlungen** WCF Dienste sind der bevorzugte Weg zur Interaktion mit SharePoint Objekten, wenn die Anwendung, welche mit den Objekten arbeiten möchte, nicht auf dem SharePoint Server ausgeführt wird. Sie bieten sich auch an, wenn eine portalübergreifende Kommunikation notwendig ist. Durch Entwicklung eigener Dienste können redundante Funktionen vermieden werden. Die folgenden Tabellen dienen als Entscheidungsgrundlage für zukünftige Projekte. In Tabelle 9 werden die drei Service Factorys von SharePoint verglichen. Tabelle 10 enthält allgemeine Handlungsempfehlungen beim Entwickeln und Nutzen von WCF Diensten.



Service Factory	Empfehlung	Alternative
ADO.NET Data Service Factory	nutzen wenn <ul style="list-style-type: none"> <li>• Daten per URI abgerufen werden sollen</li> <li>• Open Data Protocol (OData) gefordert ist</li> <li>• Daten als ADO.NET Entity Data Model verarbeitet werden sollen</li> <li>• andere Formate (AtomPub, JSON) gefordert sind</li> </ul>	REST Service Factory
REST Service Factory	nutzen wenn <ul style="list-style-type: none"> <li>• Daten per URI abgerufen werden sollen</li> <li>• JSON oder AtomPub gefordert ist</li> <li>• auch wenn Dienst kein Datendienst ist</li> </ul>	ADO.NET Data Service Factory oder SOAP Service Factory
SOAP Service Factory	nutzen wenn SOAP Protokoll genutzt werden soll	REST Service Factory oder eigene Factory

Tabelle 9: Handlungsempfehlungen zur Verwendung der Service Factorys (eigene Darstellung)

Anforderung	Empfehlung	Alternative
Eigener WCF Dienst bereitstellen	wenn <ul style="list-style-type: none"> <li>• vorhandene Dienste nicht ausreichen</li> <li>• mehrere Dienstaufrufe in einem Aufruf gekapselt werden sollen</li> <li>• von Clientanwendungen oder anderen Systemen zugegriffen werden soll</li> </ul> in jedem Fall sollte eine Service Factory genutzt werden	Client Objektmodell nutzen
Eigener WCF Dienst bei SharePoint Online Umgebungen	<ul style="list-style-type: none"> <li>• Dienst muss auf eigenem Server bereitgestellt werden</li> <li>• Client Objektmodell im Dienst nutzen</li> </ul>	vorhandene Dienste und Client Objektmodell direkt in Applikation nutzen
Nutzen von Diensten in WebParts, Workflows etc.	Endpunkt in eigener web.config definieren. Nicht verwenden bei <ul style="list-style-type: none"> <li>• SharePoint Online Umgebungen</li> <li>• Sandbox Lösungen</li> </ul>	Endpunkte im Code oder in globaler web.config definieren
Nutzung des vorhandenen REST Dienstes (listdata.svc)	nutzen wenn <ul style="list-style-type: none"> <li>• kein SOAP verwendet werden soll</li> <li>• aus Java oder anderen Umgebungen heraus aufgerufen wird</li> </ul> keine Service Referenzen in Visual Studio 2010 erstellen	SOAP Nachricht per Code erstellen und anderen Dienst nutzen

Tabelle 10: Handlungsempfehlungen beim Entwickeln und Nutzen von WCF Diensten für SharePoint (eigene Darstellung)

### 3.3.5 Access Services

Bei den Access Services handelt es sich um einen Dienst zum Veröffentlichen von Access Datenbanken. Die Datenbanken können anschließend direkt im Browser betrachtet und mit den Datensätzen interagiert werden, ohne dass Access auf dem Client installiert sein muss <sup>94</sup>. Sie sind nur in der SharePoint Server 2010 Enterprise Version verfügbar <sup>95</sup>. Nach Microsoft [2011a] bieten sich die Access Services an, wenn Daten und Datenbankfunktionen über Abteilungs- und Organisationsgrenzen hinweg geteilt werden sollen. Außerdem gewährleisten sie umfassenden Zugriffsschutz und Datensicherheit.

Bei näherer Betrachtung ergeben sich darüber hinaus weitere Vorteile. Dank der Access Services kann ein erfahrener Benutzer eigene Datenbankanwendungen mit Access gestalten und in SharePoint zur Verfügung stellen, ohne auf einen Entwickler angewiesen zu sein. Ein Entwickler kann die Access Services zum Erstellen einfacher datengetriebener Anwendungen nutzen, ohne Workflows, Web-Parts oder sonstigen Code erstellen zu müssen. Wird eine Access Datenbank von mehreren Clients verwendet, muss diese bei Änderungen an die entsprechenden Clients verteilt werden. Durch die Nutzung der Access Services entfällt die Notwendigkeit der Verteilung, da die gesamte Anwendung zentral in einer SharePoint Website abgelegt ist.

Eine veröffentlichte Datenbank wird als Web Datenbank bezeichnet <sup>96</sup>. Beim Veröffentlichen einer Datenbank in SharePoint wird eine neue Website erstellt. In dieser werden für jede Tabelle, die in der Datenbank angelegt wurde, eine SharePoint Liste erstellt und evtl. vorhandene Datensätze übernommen. Formulare und Berichte werden ebenfalls in der neuen Website abgelegt und können direkt im Browser betrachtet werden (Abbildung 3.11). Makros werden als deklarative Workflows bereitgestellt <sup>97</sup>. Die Datenbank kann auch wie gewohnt mit Access 2010 auf einem Client geöffnet werden. Eine zeitgleiche Verwendung des Browsers und Access 2010 auf dem Client wird unterstützt.

Damit die Veröffentlichung gelingt, muss die Datenbank mit einer Web Datenbank kompatibel sein. Zu den häufigsten Fehlern, die eine Veröffentlichung verhindern, zählen laut Microsoft [2010a]

- Ungültige Spalten- und Tabellennamen; beispielsweise reservierte Schlüs-

---

<sup>94</sup>Microsoft [2011a]

<sup>95</sup>Microsoft [2010b]

<sup>96</sup>Lewis [2009]

<sup>97</sup>Microsoft [2010a]

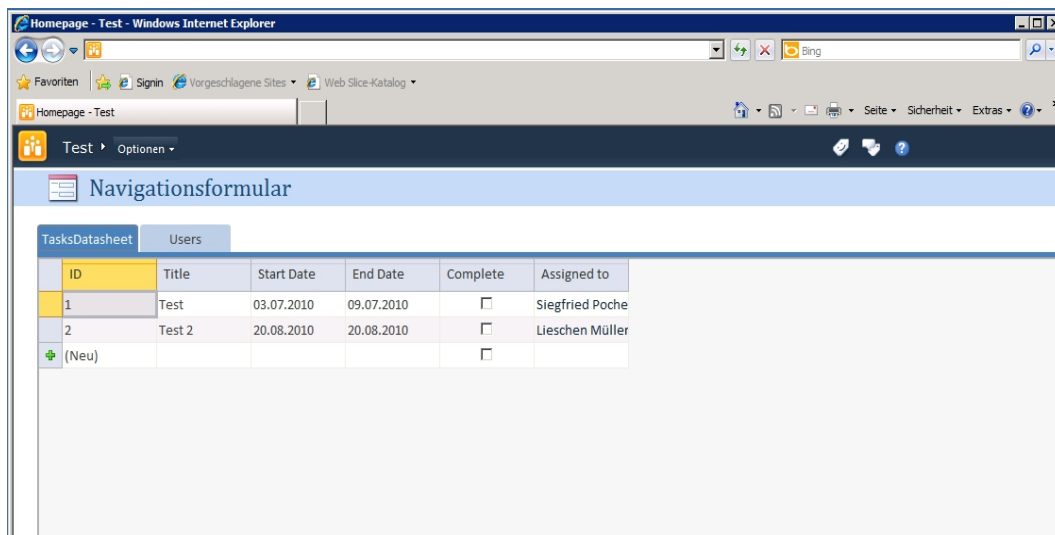


Abbildung 3.11: Web Datenbank im Browser (eigene Darstellung)

#### selwörter

- Zusammengesetzte Primärschlüssel: Es darf nur eine Spalte als Primärschlüssel definiert sein. Die Spalte muss vom Typ Long Integer sein.
- Fremdschlüsselbeziehungen: Es können keine echten Fremdschlüsselbeziehungen genutzt werden. Beziehungen können nur als Nachschlagefeld definiert werden.
- Primärschlüsselfeld hat ungültigen Datentyp: Das Primärschlüsselfeld muss vom Typ Long Integer sein.

Darüber hinaus gibt es weitere Einschränkungen, die zwar eine Veröffentlichung nicht verhindern, in der Web Datenbank aber nicht genutzt werden können<sup>98</sup>. Diese sind

- Visual Basic Skripte
- Keine verlinkten Tabellen: In Access 2010 können als Datenbasis auch Tabellen anderer Datenbanken genutzt werden. Diese können in einer Web Datenbank nicht verwendet werden.
- Union Query: Es können keine Union Abfragen ausgeführt werden<sup>99</sup>.
- Datentypen: In einer Web Datenbank können nur wenige Datentypen ge-

<sup>98</sup>Microsoft [2010a]

<sup>99</sup>Microsoft [2011a]

nutzt werden <sup>100</sup>.

In Access 2010 werden zwei verschiedene Objekttypen unterschieden, Web Objekte und Client Objekte, welche die bekannten Access Objekte, etwa Tabellen, Formulare oder Berichte repräsentieren. Nur Web Objekte können im Browser dargestellt werden <sup>101</sup>.

Dies hat bei der praktischen Arbeit weitreichende Folgen. Wird eine bestehende Datenbank veröffentlicht, werden nur die Tabellen als Web Objekte, also als SharePoint Listen erstellt. Formulare oder Berichte werden nicht in Web Objekte umgewandelt. Der Benutzer muss diese folglich als entsprechendes Web Objekt neu erstellen. Dies erschwert die Veröffentlichung bestehender, komplexer Datenbanken erheblich.

Um zwischen den einzelnen Objekten (Formulare und Berichte) einer Web Datenbank bequem navigieren zu können, kann ein spezielles Formular, das Navigationsformular, erstellt werden. Damit das Formular beim Öffnen der Web Datenbank angezeigt wird, muss es in Access 2010 unter den Optionen der Datenbank als Startformular definiert werden <sup>102</sup> (Anhang A.27). Zur Verwaltung der Zugriffsrechte werden die Berechtigungen auf die während des Veröffentlichungsvorgangs erstellten Website vergeben <sup>103</sup>.

Bei näherer Betrachtung der von den Access Services erstellten Website fällt zunächst auf, dass hier nicht die Master Page der anderen Websites verwendet wird. Eine Web Datenbank fügt sich nicht in die bestehenden SharePoint Seiten ein. Dies betrifft zum einen die Farbgestaltung der Elemente, aber auch die Möglichkeiten der Administration. Das Site Action Menü, über das üblicherweise auf administrative Seiten zugegriffen werden kann, fehlt. Auch die vertrauten Navigationselemente wie der linke Navigationsbereich oder die Bread crumb Navigation fehlen (Abbildung 3.12). Über das Optionen Menü können wenige administrative Bereiche geöffnet werden. Im Bereich Einstellungen werden alle Web Objekte der Datenbank angezeigt. Diese können hier entweder betrachtet, oder mit Access bearbeitet werden (Anhang A.28). Die gewohnten Listenfunktionen stehen nicht zur Verfügung.

Administrativen Seiten lassen sich nur indirekt erreichen. Die Seiten haben keine festgelegte URL, sondern können über eine Adresse nach dem Muster `http://<Site collection>/<Web Site>/_layouts/<administrative Seite>.aspx` aufgerufen werden. Ein Aufruf der Seite `viewlists.aspx` liefert die gewohnte Ansicht aller Website Ob-

---

<sup>100</sup>Krist [2010]

<sup>101</sup>Lewis [2009]

<sup>102</sup>Krist [2010]

<sup>103</sup>Microsoft [2011a]

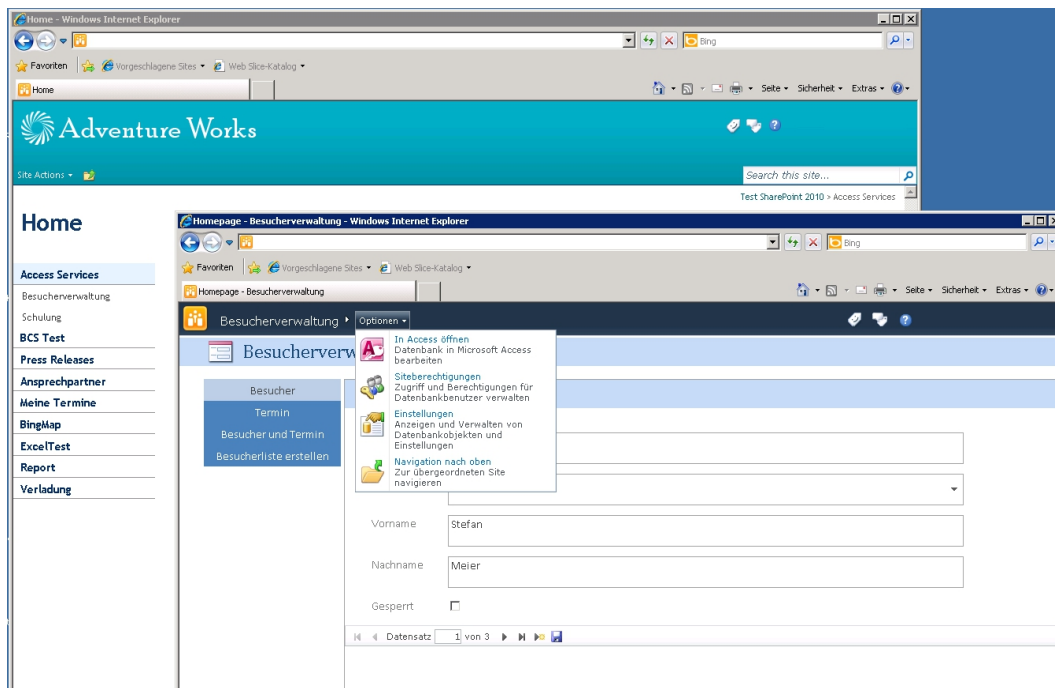


Abbildung 3.12: Vergleich des Layouts einer Website mit einer Web Datenbank (eigene Darstellung)

jekte (Anhang A.29). Hier ist deutlich zu sehen, dass alle Tabellen der Datenbank als einfache Listen angelegt wurden. Dies scheint viele Entwicklungsmöglichkeiten, etwa die Anbindung von Workflows, zu ermöglichen. Microsoft [2011b] weist jedoch ausdrücklich darauf hin, dass keine deklarativen oder programmierten Workflows bei diesen Listen unterstützt werden. Dennoch ergeben sich hier neue Möglichkeiten, die über die Oberfläche der Web Datenbank nicht realisierbar sind. Beispielsweise können Benutzer mittels Listenbenachrichtigung (Alerts) über Datenänderungen unterrichtet werden. Auch lassen sich Benutzerrechte wesentlich feiner definieren. Einem Benutzer können auf die Listen einer Web Datenbank unterschiedliche Berechtigungen gegeben werden. Versucht ein Benutzer nun, einen Eintrag in einer Tabelle zu erstellen oder zu verändern, für die er nur Leserechte hat, erhält er eine entsprechende Fehlermeldung (Anhang A.30). Die vorhandenen Web Services von SharePoint, etwa die REST Schnittstelle zum Verarbeiten von Listen, können auch bei Listen verwendet werden, die mit den Access Services erstellt wurden. Somit können Drittsysteme auf die Daten lesend und schreibend zugreifen.

**Anwendungsfall 1: Besucherverwaltung** Das Unternehmen empfängt täglich viele Besucher. Die Besucher bewegen sich im begrenzten Rahmen selbstständig auf dem Firmengelände. Zur Begrüßung und Identifizierung der Besucher sollen über die Zentrale Besucherausweise ausgegeben werden. Die Mitarbei-

ter der Zentrale sollen in der Lage sein, neue Besucher und Besuchstermine zu erfassen. Zusätzlich sollen Besuche über andere Systeme, etwa dem CRM System angemeldet werden können. Mittels Berichten sollen sich die Mitarbeiter der Zentrale über die erwarteten Besucher informieren können und Besucherausweise ausstellen. Zur Realisierung der Anforderungen werden die Access Services genutzt. In Access 2010 wird dazu eine neue Web Datenbank angelegt. Nun können die benötigten Tabellen (Besucher und deren Besuche) angelegt werden. Um eine Beziehung zwischen den Tabellen herzustellen, wird in der Tabelle Besuche ein Nachschlagefeld (Lookup Feld) erstellt, welches auf die Tabelle der Besucher verweist. Anschließend werden einige neue Web Formulare erstellt, mit denen neue Besucher und Termine angelegt werden können. Damit der Benutzer in der Applikation später zwischen den einzelnen Formularen und Berichten navigieren kann, wird ein neues Navigationsformular erstellt und alle Verknüpfungen definiert. Das Navigationsformular wird als Startformular der Web Datenbank festgelegt. Zur Ausgabe der erwarteten Besucher und der Besucherausweise werden neue Web Berichte angelegt. Die auszugebenden Daten lassen sich per Parameter definieren. Nach dem Bereitstellen der Web Datenbank können die Benutzerrechte angepasst werden. Beim Aufruf der neuen Website wird das Navigationsformular ausgegeben. Durch Auswahl eines Navigationseintrages wird das entsprechende Formular oder der Bericht erstellt (Abbildung 3.13). Da die Tabellen der Web Datenbank beim Bereitstellen in einfache SharePoint Listen umgewandelt wurden, können Datensätze mit den üblichen Web Services manipuliert werden.

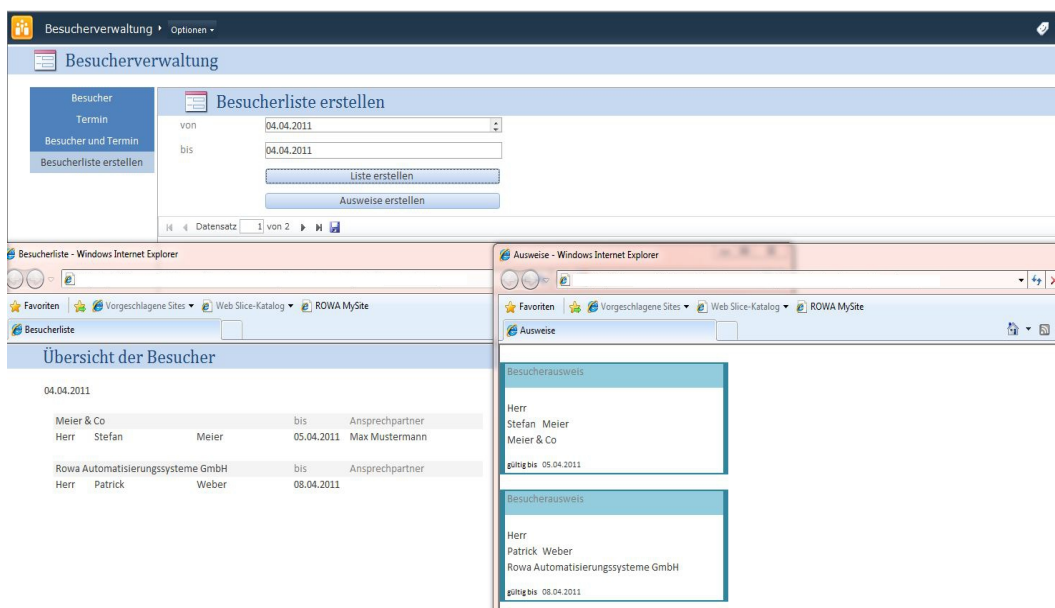


Abbildung 3.13: Besucherverwaltung mit Berichten (eigene Darstellung)

**Anwendungsfall 2: Schulungsdatenbank** Das Unternehmen bietet regelmäßig Schulungen für Mitarbeiter an. Darüber hinaus finden regelmäßig Schulungen für Kunden statt. Die Schulungen der Mitarbeiter werden über die Personalabteilung, Kundens Schulungen über die Abteilung Service koordiniert. Zur Verwaltung der Schulungen setzt die Abteilung Service eine eigene Schulungsdatenbank auf Basis von Access 2007 ein. Diese soll zukünftig auch der Personalabteilung dienen, um Mitarbeiterschulungen zu verwalten. Da bereits eine Datenbank existiert, wurde entschieden, diese weiter zu verwenden und keine neue Applikation zu entwickeln. Zur Realisierung der Anforderungen muss die vorhandene Access Datenbank mit Access 2010 geöffnet werden. Nur diese Version bietet die Möglichkeit, Access Datenbanken in SharePoint zu veröffentlichen. Bei der Prüfung der Kompatibilität wird festgestellt, dass alle vorhandenen Fremdschlüsselbeziehungen durch Lookup Felder ersetzt werden müssen (Abbildung 3.14). Dazu werden zunächst alle definierten Beziehungen entfernt. Anschließend wird

ID	Elementtyp	Elementname	Steuerelem	Steuerelem	Eigenschaft	Problemtyp	Problemtyp-II	Beschreibung	Zum Hinzufügen klicken
1	Beziehung	VERANSTALTU	Beziehung	VERANSTALTU	NGEN.LEHRER	Fehler	ACCWeb105016	Beziehungen, die nicht mit einem webkompatiblen Nachschlagefeld verknüpft sind, sind nicht mit dem Web kompatibel.	
2	Beziehung	VERANSTALTU	Beziehung	VERANSTALTU	NGEN.SCHULU	Fehler	ACCWeb105016	Beziehungen, die nicht mit einem webkompatiblen Nachschlagefeld verknüpft sind, sind nicht mit dem Web kompatibel.	
3	Beziehung	VERANSTALTU	Beziehung	VERANSTALTU	NGSTEILNEHME	Fehler	ACCWeb105016	Beziehungen, die nicht mit einem webkompatiblen Nachschlagefeld verknüpft sind, sind nicht mit dem Web kompatibel.	
4	Beziehung	VERANSTALTU	Beziehung	VERANSTALTU	NGSTEILNEHME	Fehler	ACCWeb105016	Beziehungen, die nicht mit einem webkompatiblen Nachschlagefeld verknüpft sind, sind nicht mit dem Web kompatibel.	
*	(Neu)								

Abbildung 3.14: Kompatibilitätsfehlermeldungen (eigene Darstellung)

in der Entwurfsansicht der Tabellen das Fremdschlüsselfeld mit dem Nachschlage Assistenten in ein Lookup Feld geändert. Im Assistenten kann auch ausgewählt werden, ob die Datenintegrität durch Löschweitergabe oder Verhindern von Löschen gesichert werden soll. Alle Formulare und Berichte, die in SharePoint verfügbar sein sollen, müssen als entsprechendes Web Objekt neu erstellt werden. Damit zwischen den verschiedenen Formularen und Berichten navigiert werden kann, muss eine Navigationsseite erstellt werden. Nach dem Veröffentlichen kann die neu erstellte Website aufgerufen werden (Anhang A.31).

**Anwendungsfall 3: Portierung einer umfangreichen Datenbank** Das Unternehmen verwendet zur Planung und Koordinierung von Kundenprojekten den selbst entwickelten Rowa Planer. Dabei handelt es sich um eine Access Applikation, deren Daten in einer MS SQL Datenbank abgelegt sind. Die Applikation besteht seit vielen Jahren und wurde ständig weiterentwickelt. Nach dem Öffnen



der Datenbank mit Access 2010 und einer anschließenden Web Kompatibilitätsprüfung werden zahlreiche Fehler angezeigt, die eine Portierung zu einer Web Datenbank verhindern. Zu den Gründen, die eine Portierung verhindern, zählen die zahlreichen Fremdschlüsselbeziehungen zwischen den Tabellen, die häufige Verwendung von Visual Basic Skripten sowie die Verwendung von verlinkten Tabellen. Dies wird von einer Web Datenbanken nicht unterstützt. Zur Portierung müssten alle Tabellen neu erstellt und Fremdschlüsselbeziehungen durch Nachschlagefelder ersetzt werden. Zusätzlich müssen alle VBA Skripte durch Daten Makros ausgetauscht werden. Damit Formulare und Berichte in einer Web Datenbank angezeigt werden, müssen diese als Web Formulare bzw. Web Berichte neu erstellt werden. Aufgrund der Komplexität der Applikation und den umfassenden Änderungen, die für eine Portierung notwendig wären, wird von der Portierung abgesehen.

**Alternative: SharePoint Liste mit InfoPath Formular** Mit den Access Services lassen sich datengetriebene Anwendungen schnell umsetzen. Sie stoßen allerdings an ihre Grenzen, wenn Workflowfunktionen oder die nahtlose Integration in eine bestehende Website gefordert sind. Workflowfunktionen lassen sich mit einfachen SharePoint Listen realisieren. Durch die Anpassung der vorhandenen Listenformulare mit InfoPath lassen sich zudem ansprechende Formulare gestalten. Auch eine Datenvalidierung lässt sich so umsetzen.

Im folgenden Beispiel sollen die Kunden des Unternehmens über das Kundenportal ein Feedback zu Mitarbeiterkontakten geben. Ziel ist es, den Kunden eine einfache Möglichkeit zu geben, die Mitarbeiter des Unternehmens zu bewerten. Die Daten sollen zur Steigerung der Servicequalität genutzt werden. Das Feedback soll in einer SharePoint Liste gespeichert werden. Neue Einträge werden vom Kunden mit einem Formular erfasst, welches sich in das bestehende Layout der Website einfügen soll. Sobald der Kunde einen Eintrag erfasst hat, sollen mittels Workflow, abhängig von der Beurteilung des Kunden, Aktivitäten im CRM System (beispielsweise ein Rückruf) eingeplant werden.

Zur Realisierung der Anforderung wird zunächst eine neue SharePoint Liste erstellt, die alle notwendigen Spalten (Name des Mitarbeiters, Grund des Kontaktes, Datum, Bewertungen etc.) enthält. Anschließend wird mit dem InfoPath Designer 2010 ein neues SharePoint Listenformular erzeugt. Nachdem die Vorlage ausgewählt wurde, kann die zu bearbeitende Liste ausgewählt werden. Es wird nun das vorhandene Listenformular geöffnet. Der Benutzer hat Zugriff auf alle Spalten der Liste. Ist das Formular fertig gestaltet, kann es mit der Schnellveröffentlichungsfunktion bereitgestellt werden (Anhang A.32). Damit der Kunde nicht zur Liste navigieren muss, um sein Feedback abgeben zu können, wird der

Link zu einem neuen Listenelement an einer passenden Stelle platziert. Folgt der Kunde dem Link, wird das soeben erstellte Formular geladen. Das Aussehen und Verhalten der Website bleibt erhalten (Abbildung 3.15). Zur Erfüllung der geforderten Workflowfunktion kann dieser mit dem SharePoint Designer oder mit Visual Studio 2010 erstellt werden.

Abbildung 3.15: Kundenfeedback Formular (eigene Darstellung)

**Empfehlungen** Mit den Access Services lassen sich einfache Datenbanken in SharePoint veröffentlichen und mit anderen Personen teilen. Sie bieten sich an, wenn Endbenutzer eigene Anwendungen ohne Hilfe von Entwicklern erstellen sollen. In einer SharePoint Online Umgebung sind sie oftmals die einzige Möglichkeit, umfangreiche datengetriebene Anwendungen bereitzustellen. Tabelle 11 enthält allgemeine Handlungsempfehlungen bei der Verwendung der Access Services und dient als Entscheidungsgrundlage für zukünftige Projekte.

Anforderung	Empfehlung	textbfAlternative
Workflows	keine Access Services verwenden	<ul style="list-style-type: none"> <li>• Listen manuell anlegen</li> <li>• Listenformulare anpassen oder</li> <li>• WebPart bereitstellen</li> </ul>

Migration bestehender Access Datenbanken	nur kleine Datenbanken mit wenigen Objekten migrieren	Datenbank neu erstellen
Zugriff auf externe Daten	keine Access Services verwenden	InfoPath Formulare für externe Daten erstellen
Integration in Seitenlayout	keine Access Services verwenden	Listenformulare mit InfoPath anpassen oder WebPart erstellen
Zugriff von Extern	SharePoint Dienste nutzen	Client- oder Serverobjektmodell nutzen
Benachrichtigung bei Datenänderung	Benachrichtigungsfunktion der Listen verwenden	

Tabelle 11: Handlungsempfehlungen bei der Verwendung von Access Services  
(eigene Darstellung)

### 3.3.6 Excel Services

Bei den Excel Services handelt es sich um einen Dienst zum Veröffentlichen von Excel Arbeitsmappen. Die Arbeitsmappen können direkt im Browser betrachtet werden. Ohne das Excel auf dem Client installiert sein muss, kann mit den Daten interagiert werden (Abbildung 3.16). Darüber hinaus bieten die Excel Services einen REST Dienst für den Zugriff auf Inhalte der Arbeitsblätter per URL an <sup>104</sup>. Sie sind nur in der SharePoint Server 2010 Enterprise Version verfügbar <sup>105</sup>. Die

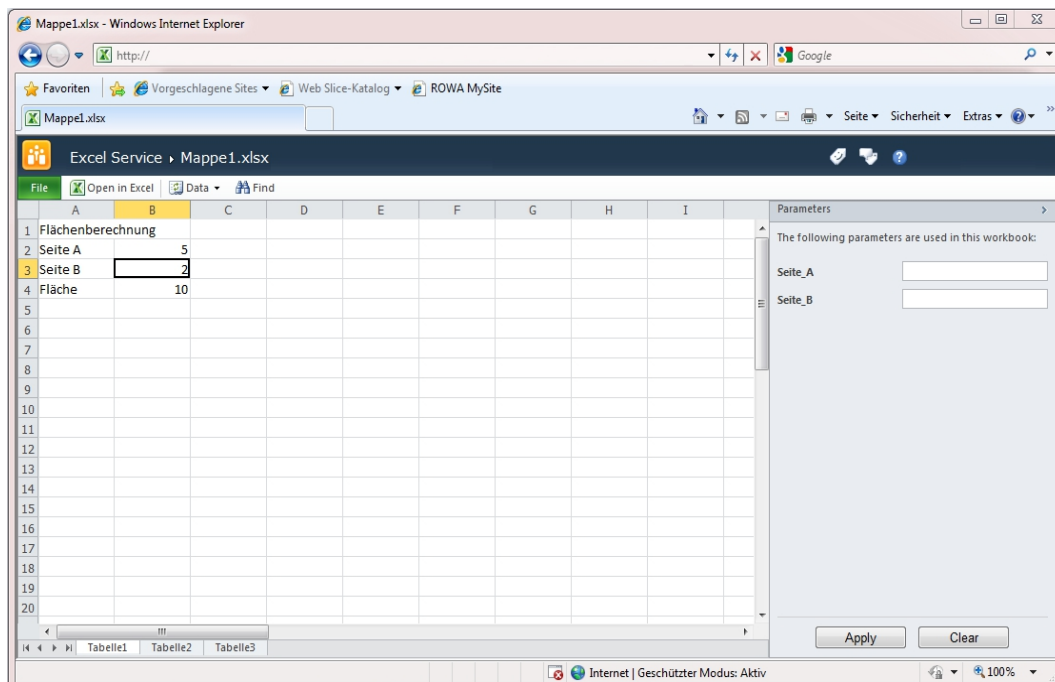


Abbildung 3.16: Excel Arbeitsmappe im Browser (eigene Darstellung)

veröffentlichten Arbeitsmappen können mit dem Excel Web Access WebPart auf beliebigen Seiten angezeigt werden. Enthalten die Mappen eine Verbindung zu externen Datenquellen, beispielsweise zu einem SQL Server, können die Daten direkt im Browser aktualisiert werden. Werden in der Arbeitsmappe benannte Bereiche verwendet, können diese als Parameter im Excel Web Access WebPart angezeigt werden. Mit den Excel Services können keine neuen Dokumente erstellt oder bestehende verändert werden <sup>106</sup>. Die Administration der Excel Services erfolgt mit der Central Administration. Bei der Überprüfung der Standardeinstellungen ist aufgefallen, dass nur ein Ort als vertrauenswürdiger Speicherort eingerichtet ist (Anhang A.33). Als Adresse ist die URL `http://` eingetragen, was bedeutet, dass alle Dokumentenbibliotheken in der gesamten Farm als vertrauenswürdige Speicherorte gelten. Es können sowohl eingebettete als auch

<sup>104</sup>TechNet [2010c]

<sup>105</sup>Microsoft [2010b]

<sup>106</sup>MSDN [2010d]

vertrauenswürdige Datenverbindungen genutzt werden. Dies kann geprüft werden, indem Excel Arbeitsmappen in beliebigen Dokumentenbibliotheken abgelegt werden. Durch Anklicken einer Arbeitsmappe wird diese im Browser angezeigt. Sollen nur bestimmte Dokumentenbibliotheken für die Excel Services erreichbar sein, muss ein neuer vertrauenswürdiger Speicherort eingetragen und der vorhandene Eintrag der Farm gelöscht werden. Dies kann erforderlich sein, wenn nur wenige Personen Arbeitsblätter veröffentlichen sollen oder nur bestimmte Datenquellen verwendet werden dürfen. Werden externe Daten in der Arbeitsmappe abgerufen, muss in der Datenverbindung angegeben werden, welche Authentifizierung die Excel Services nutzen sollen. Es können folgende Authentifizierungstypen ausgewählt werden:

- Windows Authentifizierung: Die Windows-Anmeldeinformationen des in SharePoint angemeldeten Benutzers werden zur Anmeldung an der Datenquelle verwendet.
- SSS-ID: Es wird eine Benutzeranmeldung verwendet, die im Secure Store Service hinterlegt ist.
- Keine: die Authentifizierung erfolgt mit einer Benutzeranmeldung, die im Secure Store Service hinterlegt ist. Welche Benutzeranmeldung verwendet werden soll, wird in den Global Settings der Excel Services unter Unattended Service Account festgelegt (Abbildung A.34).

Bei der Überprüfung der Authentifizierungsverfahren ist aufgefallen, dass die Excel Services in der Standardinstallation nicht auf Benutzeranmeldungen zugreifen können, die im Secure Store Service hinterlegt sind. Die Authentifizierungsarten SSS und Keine können somit nicht genutzt werden. Die Arbeitsmappen werden zwar im Browser angezeigt, die Daten können aber nicht aktualisiert werden (Abbildung 3.17). Es werden veraltete Daten angezeigt. Damit Benutzeranmel-

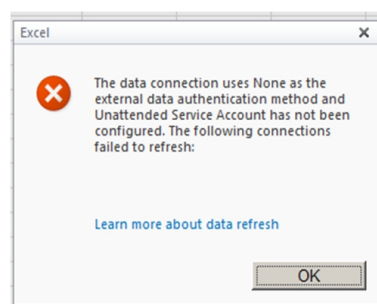
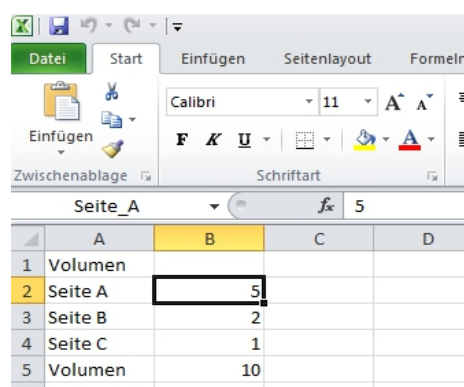


Abbildung 3.17: Fehlermeldung der Excel Services bei Verwendung der Secure Store Services (eigene Darstellung)

dungen, die im Secure Store Service hinterlegt sind, verwendet werden können, muss der Benutzer, der die Anmeldung nutzen möchte, dazu berechtigt sein. Welche Benutzer eine bestimmte Anmeldung nutzen dürfen, wird in der Central Administration festgelegt. Hier können nur Benutzer aus dem Active Directory oder aus anderen Authentifizierungsprovidern angegeben werden. Bei der Installation von SharePoint werden einige Web Applications angelegt. Unter anderem wird eine eigene Web Application für diverse SharePoint Dienste wie Business Connectivity Service, Secure Store Service oder Excel Services, eingerichtet. Jede Web Application wird mit einer bestimmten Identität ausgeführt. Die Identität kann in der Central Administration im Bereich Security geprüft und geändert werden. Die Web Application für SharePoint Dienste wird in der Standardinstallation mit der Identität Local Service ausgeführt. Da diese Identität nicht als berechtigter Benutzer im Secure Store Service hinterlegt werden kann, können die Excel Services keine hier hinterlegten Anmeldeinformationen nutzen (Anhang A.35). Erst nach Änderung der Identität der Web Application auf einen existierenden Benutzer und Berechtigung des Benutzers auf die Anmeldeinformation im Secure Store Service können die Authentifizierungsarten SSS und Keine der Excel Services genutzt werden. Zur Anzeige der Excel Arbeitsblätter kann der Excel Web Access WebPart verwendet werden. Wie bei MSDN [2010d] beschrieben, verfügt der WebPart über eine Parameter Pane. Dies ist ein Bereich, mit dem der Benutzer bestimmte Parameter an die Arbeitsmappe senden kann um Eingabewerte für Berechnungen oder Datenfilter zu setzen. Damit bestimmte Zellen in der Parameter Pane dargestellt werden, müssen die Zellen mit einem Namen versehen (Abbildung 3.18) und zusätzlich beim Speichern der Arbeitsmappe angegeben werden, welche Elemente verfügbar sein sollen (Abbildung A.36). Beim Prüfen



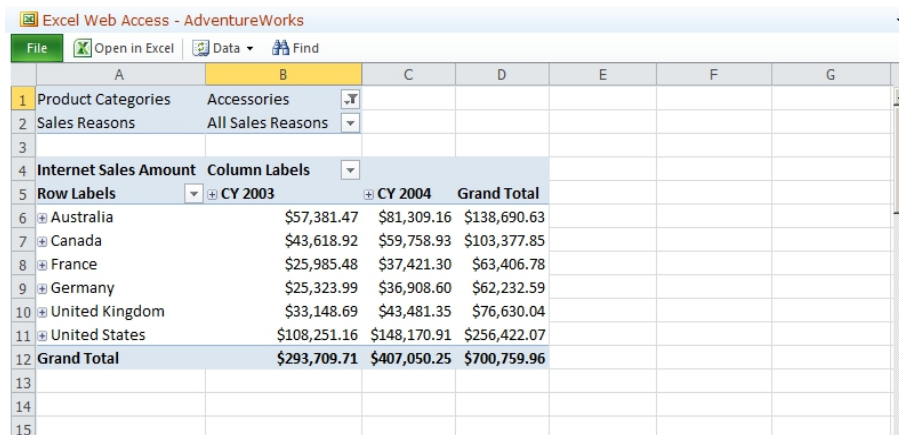
	A	B	C	D
1	Volumen			
2	Seite A	5		
3	Seite B	2		
4	Seite C	1		
5	Volumen	10		

Abbildung 3.18: Benannte Zelle in Excel (eigene Darstellung)

der benannten Parameter bei externen Datenquellen ist eine Besonderheit bei Verbindungen zu einem MS SQL Server Analysis Cube aufgefallen. Excel stellt die Daten als Pivot Tabelle dar. Werden in den Berichtsfiler Dimensionsattribute des Cubes abgelegt und die so entstandenen Zellen als Parameter bereitgestellt,

werden diese nicht als Parameter Pane dargestellt. Die Parameter können nur mit einem verbundenen Filter WebPart gefüllt werden. Der Filter muss den vollständigen Multidimensional Expression (MDX) Ausdruck an den Excel Web Access WebPart senden, sonst werden alle Elemente in der Pivot Tabelle angezeigt (Abbildung 3.19). Ansonsten werden alle Elemente in der Pivot Tabelle angezeigt. Da der Endbenutzer den korrekten Multidimensional Expressions Ausdruck nicht kennt, müssen bestimmte Filter WebParts, wie etwa der Choice Filter WebPart, verwendet werden. Der Excel Web Access WebPart verfügt über zahlreiche Ein-

Product Category  
[Product].[Product Categories].[Category].&[4]



Product Categories	Accessories					
Sales Reasons	All Sales Reasons					
Internet Sales Amount	Column Labels					
Row Labels		± CY 2003	± CY 2004	Grand Total		
Australia		\$57,381.47	\$81,309.16	\$138,690.63		
Canada		\$43,618.92	\$59,758.93	\$103,377.85		
France		\$25,985.48	\$37,421.30	\$63,406.78		
Germany		\$25,323.99	\$36,908.60	\$62,232.59		
United Kingdom		\$33,148.69	\$43,481.35	\$76,630.04		
United States		\$108,251.16	\$148,170.91	\$256,422.07		
<b>Grand Total</b>		<b>\$293,709.71</b>	<b>\$407,050.25</b>	<b>\$700,759.96</b>		

Abbildung 3.19: Excel Arbeitsmappe mit Filter WebPart (eigene Darstellung)

stellungen, mit denen das Verhalten und die Möglichkeiten der Interaktion beeinflusst werden kann. So kann bestimmt werden, ob ein Benutzer Daten Filtern oder Sortieren darf. Auch Pivot Funktionen wie das Expandieren von Gruppen können beeinflusst werden. Zudem kann dem Benutzer die Möglichkeit genommen werden, die Excel Arbeitsmappe lokal zu öffnen bzw. einen Snapshot davon zu erstellen. Arbeitsmappen können mit personenbezogenen Parametern ausgestattet werden. Mittels verstecktem Filter WebPart können diese beim Anzeigen der Seite automatisch mit Benutzerdaten (etwa der Personalnummer) versorgt werden. Werden die Interaktionsmöglichkeiten eingeschränkt, hat der Benutzer keine Möglichkeit, die Filter im WebPart zu ändern und Daten eines Kollegen einzusehen. Dies scheint eine geeignete Möglichkeit zu sein, dem Benutzer sensible Daten zu präsentieren, ohne dass er die vorgegebenen Filtereinstellungen verändern kann. Klickt der Benutzer aber auf den Titel des WebParts, gelangt er direkt zur Arbeitsmappe. Diese wieder zwar im Browser dargestellt, der Benutzer hat aber nun alle Möglichkeiten der Interaktion. Dies kann nur verhindert werden, indem der WebPart Titel ausgeblendet wird. Da der Benutzer mindestens Leserechte für die Arbeitsmappe hat, kann kein Schutz von sensiblen Daten gewährleistet werden.

**Anwendungsfall 1: Bereitstellung statischer Informationen** Um einen reibungslosen Ablauf bei der Be- und Endladung von LKW zu gewährleisten, erstellen die Verantwortlichen des Lagers jede Woche einen Belegungsplan der Laderampen. Auf diesem Plan ist vermerkt, wann welche Spedition erwartet wird, welche Laderampe reserviert wurde und was verladen werden soll. Der Belegungsplan wird bisher in Excel erstellt und als Ausdruck verteilt. Da sich der Belegungsplan im Laufe einer Woche ändern kann und der Ausdruck im Lageralltag stark leidet, bittet das Lagerpersonal bei den Verantwortlichen häufig um ein neues Exemplar. Um diese zu entlasten soll der Belegungsplan auf einer SharePoint Seite angezeigt werden. Der Plan soll weiterhin als Excel Arbeitsmappe erstellt werden, um den Schulungsaufwand gering zu halten. Die Mitarbeiter des Lagers sollen die in der Halle befindlichen Informationsterminals verwenden, um die Seite aufzurufen. Zunächst wird der Belegungsplan in einer Dokumentenbibliothek abgelegt. Änderungen des Plans werden zukünftig immer in diesem Dokument eingetragen. Dieses wird auf einer neuen Seite mittels Excel Web Access WebPart angezeigt (Abbildung 3.20). Die Mitarbeiter des Lagers haben so die Möglichkeit, selbst einen aktuellen Plan auszudrucken.

Hier finden Sie die Belegungspläne der Laderampen für die aktuelle Kalenderwoche. Die Druckfunktion befindet sich im Aktionsmenü.  
Bei Fragen wenden Sie sich an Ihren Vorarbeiter oder den Verlademeister.

**Belegungsplan**

File	Open in Excel	Data	Find
1	KW:	12 2011	
2	Mittwoch	Laderampe	
3	Verladeuhrzeit	1	2
4			3
5	8:00	Proj. 4711	Proj. 4321, Sped. Schenker
6	8:30	Adler Apotheke, München	General Hospital, London
7	9:00	Sped. Meier	Ladung 1 von 2
8	9:30		
9	10:00		
10	10:30	Eingang, Best. 08-154712	
11	11:00	Sped. Müller	
12	11:30	LKW 1	Eingang, Best. 08-154712
13	12:00		Sped. Müller, LKW 2
14	12:30	Pause	Pause
15	13:00	Proj. 5513, 5514	Proj. 2262, Sped. Meier
16	13:30	5515, Seecontainer	Marktapotheke, Köln
17	14:00	Sped. P&O	Sped. Müller
18	14:30		Messeanlage für
19	15:00		Expopharm
20	15:30		Proj. 4321, Sped. Schenker
21			General Hospital, London
22	Verladung		Ladung 2 von 2
23	Wareneingang		
24	Verladung Seefracht		
25	Störung		
26	Pause		

**Wichtiger Hinweis:**  
Beachten Sie unbedingt die Regeln zur ordnungsgemäßen Ladungssicherung. Die Verladung in Seecontainern muss unbedingt vom Lademeister überwacht werden.

Abbildung 3.20: Belegungsplan der Verladerampen (eigene Darstellung)



**Anwendungsfall 2: Erstellen eines Berichtsdashboards** Die Mitarbeiter des Service sollen über aktuelle Fehlerquoten und Störungshäufungen der Produkte des Unternehmens informiert werden. Zu diesem Zweck werden einmal täglich elektronische Statusberichte der Anlagen abgerufen und in einem Data Warehouse aufbereitet. Damit die Mitarbeiter einen schnellen Überblick erhalten, sollen auf einer neuen SharePoint Seite einige Diagramme und Tabellen angezeigt werden. Da diese häufig durch andere Darstellungen ersetzt werden müssen, um situationsgerechte Informationen darzustellen, wird bewusst auf den Einsatz der Business Connectivity Service verzichtet. Statt dessen erstellen die Mitarbeiter mit Excel die benötigten Auswertungen und legen diese in einer Dokumentenbibliothek ab. Die Arbeitsmappen beziehen ihre Daten vom Data Warehouse. Anschließend bearbeiten sie die SharePoint Seite, auf der die Tabellen und Diagramme dargestellt werden sollen. Die Mitarbeiter sind so in der Lage, neue Auswertungen ohne fremde Hilfe zu erstellen und bereitzustellen.

**Anwendungsfall 3: REST Schnittstelle** Das Unternehmen verfügt über ein Wikipedia System. Hier werden zahlreiche technische und vertriebsunterstützende Informationen in Form von Artikeln vorgehalten. In einem Artikel sollen Informationen über Wettbewerber und die aktuelle Marktverteilung angezeigt werden. Die Marktverteilung soll als Kuchendiagramm dargestellt werden und jederzeit aktuelle Daten enthalten. Um diese Anforderungen erfüllen zu können, wird zunächst eine Excel Arbeitsmappe mit der geforderten Auswertung erstellt. Diese wird anschließend in einer Dokumentenbibliothek bereitgestellt. In dem Artikel kann nun an der gewünschten Stelle ein neues Bild eingefügt werden. Als Adresse (URL) des Bildes wird ein REST Aufruf, wie in Listing L.4 dargestellt, angegeben.

```
1 
```

Listing L.4: Excel Diagramm per REST API abrufen

Bei der Verwendung der REST Schnittstelle ist zu beachten, dass der aufrufende Benutzer mindestens Leserechte für die Arbeitsmappe hat. Verfügt der Benutzer über keine oder unzureichende Berechtigungen, erhält er eine Fehlermeldung. Der Benutzer muss sich folglich zwingend am SharePoint System authentifizieren. Eine direkte Verwendung der REST Schnittstelle auf Seiten, die einen anonymen Zugriff erlauben (etwa die Internetpräsenz des Unternehmens), ist somit ausgeschlossen.

**Alternative: Reporting Services** Die MS SQL Server Reporting Services sind eine Sammlung von Diensten des SQL Servers, mit dem Berichte erstellt und ausgeführt werden können. Die Reporting Services lassen sich mittels dem Reporting Services Add-In direkt aus SharePoint heraus nutzen. Mit den hier enthaltenen Report Viewer WebPart können Berichte auf beliebigen Seiten dargestellt werden. Zusätzlich lassen sich Berichte per Abonnement automatisch ausführen und per E-Mail versenden oder in Dokumentenbibliotheken ablegen. Das Add-In kann kostenlos im Microsoft Download Center heruntergeladen werden. Es kann sowohl in der SharePoint Foundation 2010 als auch im SharePoint Server 2010 verwendet werden. Nähere Informationen zur Installation und Konfiguration der Reporting Services für SharePoint findet sich bei MSDN [2010f]. Ein Bericht wird entweder mit Visual Studio 2010 oder mit dem SQL Server Report Builder erstellt. Die Vorgehensweise ist in beiden Fällen identisch. Zunächst wird in einer neuen Datenquelle die Verbindung zur Datenbank definiert. Dies ist vergleichbar mit den Datenverbindungen in Excel Arbeitsmappen. Es werden verschiedene Arten von Datenquellen, etwa MS SQL Server, MS SQL Server Analysis Services oder ODBC Verbindungen, unterstützt. Anschließend kann in einem Dataset die Abfrage definiert werden, welche die Daten des Berichts liefert. Hier können auch Parameter festgelegt werden, mit denen die Daten eingegrenzt werden können. Zur Gestaltung des Berichts stehen in der Toolbox einige Elemente bereit. Die Daten lassen sich beispielsweise als Tabelle, Matrix oder Diagramm darstellen. Eine Kombination der Elemente ist ebenfalls möglich (Anhang A.37). Damit die Berichte in SharePoint angezeigt werden können, müssen sie bereitgestellt werden. Die Berichte und freigegebenen Datenquellen werden in getrennten Dokumentenbibliotheken abgelegt. Diese müssen bei Bedarf neu erstellt werden. In den Projekteigenschaften des Berichtsprojektes muss der Server und die Dokumentenbibliothek angegeben werden, die das Bereitstellungsziel bilden (Anhang A.38). Wurde im Bericht eine freigegebene Datenquelle definiert, muss diese in der Dokumentenbibliothek zur Verwendung genehmigt werden. Der Bericht kann ähnlich wie Excel Arbeitsblätter entweder in einem WebPart oder in einer eigenen Seite dargestellt werden. Wird der Report Viewer WebPart verwendet, kann wie beim Excel Web Access WebPart definiert werden, welche Aktionen (Parameter ändern, Exportieren etc.) der Benutzer ausführen darf. Parameter können auch mittels Filter WebPart gefüllt werden. Die Parameter lassen sich ausblenden, so dass der Benutzer diese nicht ändern kann (Abbildung 3.21). Klickt der Benutzer auf den Titel des Report Viewer WebPart, wird der Bericht in einer neuen Seite dargestellt. Hier hat er Zugriff auf die Parameter. Sensible Daten lassen sich also auch mit den Reporting Services nicht schützen.

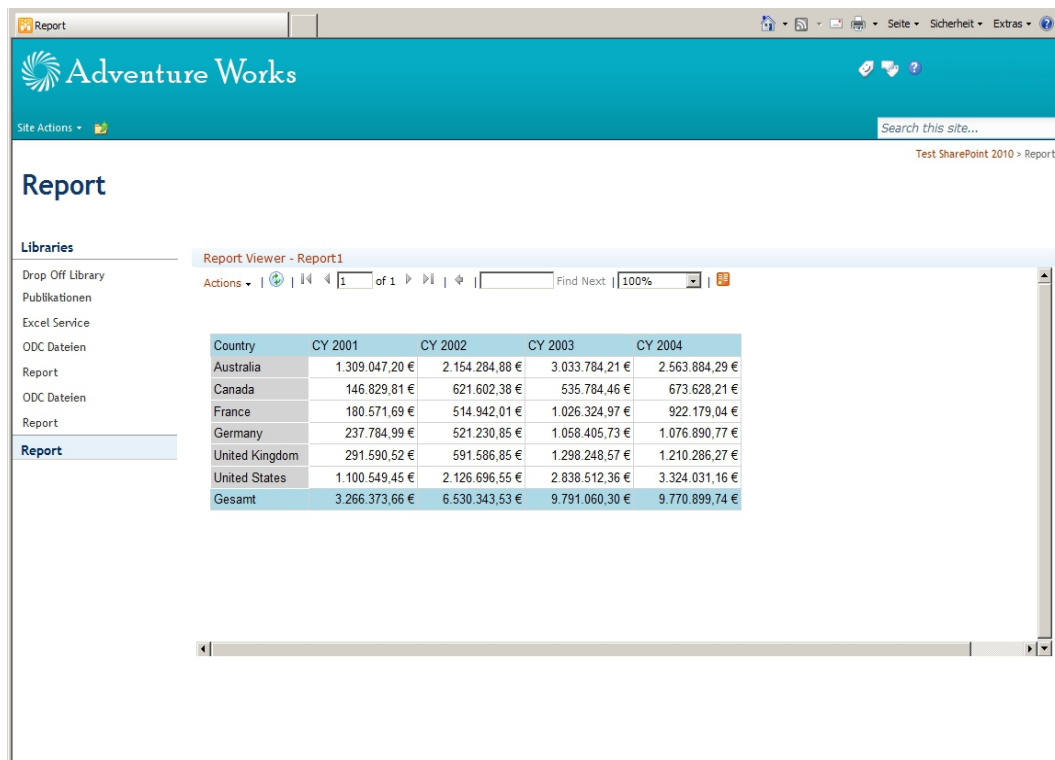


Abbildung 3.21: Reporting Services Bericht in SharePoint (eigene Darstellung)

**Empfehlungen** Die Excel Services bieten den Benutzern eine einfache Möglichkeit, mit den ihnen vertrauten Werkzeugen Berechnungsfunktionen, Auswertungen und interaktive Berichte ohne Hilfe eines Entwicklers bereitzustellen. Sie können umfassende Dashboards erstellen und ihren Bedürfnissen anpassen. Die Verantwortung, welche Daten angezeigt werden, liegt beim Benutzer. Tabelle 12 enthält allgemeine Handlungsempfehlungen bei der Verwendung der Excel Services und dient als Entscheidungsgrundlage für zukünftige Projekte.

Anforderung	Empfehlung	Alternative
Inhalt einer Arbeitsmappe darstellen	Excel Web Access Web-Part nutzen	Daten per Excel REST API anzeigen
Arbeitsmappen erstellen/ändern	nicht möglich	Office Web Applications verwenden
Anzeige externer Daten	Als externe Datenquelle einbinden, automatische Aktualisierung in Datenverbindung definieren	<ul style="list-style-type: none"> <li>• Daten per BCS anzeigen</li> <li>• Report per Report Server Add-in anzeigen</li> </ul>

Anzeige sensibler Daten	nur bedingt möglich, Titel des Excel Web Access WebParts ausblenden	Datenquelle muss Sicherheit selbst gewährleisten
-------------------------	---	--

Tabelle 12: Handlungsempfehlungen bei der Verwendung der Excel Services (eigene Darstellung)

## 4 Bewertung und Ausblick

Die Bedeutung von Collaborationsplattformen für Unternehmen hat mit der umfassenden Verbreitung der Internettechnologie und der zunehmenden Kooperation mit Lieferanten, Partnern und sonstigen Institutionen stark zugenommen. Sie kann durchaus als Wettbewerbsfaktor angesehen werden. Microsoft hat sich hier mit seiner SharePoint Produktfamilie positioniert und bietet eine ausgereifte Collaborationsplattform an. SharePoint bietet jedoch weit mehr als eine einfache Portallösung oder ein Dokumentenmanagementsystem.

### 4.1 Bewertung der Ergebnisse

Ziel dieser Arbeit war es, die Möglichkeiten von SharePoint als Applikations- und Integrationsplattform aufzuzeigen. Dazu wurde zunächst erarbeitet, welche Möglichkeiten der Lösungsbereitstellung existieren. Anschließend wurden verschiedene Technologien vorgestellt und Anwendungsszenarien skizziert. Zudem wurden Handlungsempfehlungen zu jeder Technologie erarbeitet. Dabei hat sich gezeigt, dass die Business Connectivity Services der zu bevorzugende Weg sind, Inhalte aus Datenbanken in SharePoint darzustellen und anderen Technologien wie WebParts oder Workflows zur Verfügung zu stellen. Obwohl mit ihnen auch Daten zurückgeschrieben werden können, ist hiervon abzuraten.

WebParts bilden einen wesentlichen Baustein einer SharePoint Applikation. Die Entwicklungsmöglichkeiten sind im hohen Maße davon abhängig, ob sie auf einer gehosteten SharePoint Umgebung ausgeführt werden oder nicht. Aufgrund der umfassenden Restriktionen von SharePoint Online Umgebungen ist es oftmals sinnvoller, die gewünschten Funktionen mit Silverlight zu realisieren.

Die Möglichkeiten von Workflows hängen sehr stark von der eingesetzten Umgebung ab. In einer SharePoint Online Umgebung können nur mit dem SharePoint Designer erstellte, sequenzielle Workflows eingesetzt werden. In der SharePoint Foundation 2010 können nur ASPX Seiten als Workflowformulare genutzt werden. In jedem Falle sollten eigene Workflowfunktionen als Custom Activity erstellt werden.

Mit der Windows Communication Foundation können mit geringem Aufwand Dienste erstellt werden, welche von SharePoint gehostet werden können. Es hat sich gezeigt, dass Dienste nicht in SharePoint Online Umgebungen bereitgestellt werden können. Die Verwendung aus SharePoint heraus gestaltet sich aufgrund der Definition der Endpunkte schwierig. Sie sollten entweder in einer eigenen Konfigurationsdatei oder per Programmcode erfolgen.

Mit den Access Services lassen sich Access Datenbanken als Website veröffentlichen und online nutzen. Es hat sich gezeigt, dass sie nur sehr begrenzt eingesetzt werden können. Die Migration bestehender Datenbanken ist sehr aufwändig.

Mit den Excel Services können Benutzer umfangreiche und interaktive Dashboards auf Basis von Excel Arbeitsmappen erstellen. Dies lässt sich allerdings auch mit dem Report Server Ad-in realisieren, welches kostenlos bei Microsoft erhältlich ist.

Es hat sich gezeigt, dass bereits mit der SharePoint Foundation 2010 viele Anforderungen bedient werden können. Die Unterschiede der Versionen betreffen zwar auch die Entwicklungsmöglichkeiten, jedoch muss gründlich geprüft werden, ob eine umfassendere Version wirklich erforderlich ist. Oftmals bieten sich Alternativen, die zwar einen gewissen Entwicklungsaufwand mitbringen, ein Upgrade auf eine umfassendere Version und die damit verbundenen Kosten aber überflüssig machen können. Die Entscheidung für eine bestimmte SharePoint Version sollte also nicht alleine Aufgrund der Entwicklungsmöglichkeiten getroffen werden.

Die volle Leistungsfähigkeit von SharePoint als Applikationsplattform ergibt sich erst mit der Kombination der Entwicklungsmöglichkeiten. Bei Entwicklungsprojekten muss folglich nicht eine bestimmte Technologie, sondern die sinnvollsten Kombination ermittelt werden.

## **4.2 Ausblick**

Im Rahmen dieser Arbeit wurden verschiedene Entwicklungsmöglichkeiten aufgezeigt. Diese sind jedoch nicht allumfassend. So wurden die Möglichkeiten, die von Feature- und EventReceivern, Timer Jobs und administrativen Erweiterungen ausgehen, nur am Rande behandelt. Auch wurden im Rahmen dieser Arbeit keine Vorlagen zur Bedienung von Anforderungen, wie etwa eine Vorlage für die Nutzung von ASPX Workflow Task Formularen erstellt. Es wurde auf Schwierigkeiten bei der Definition von Endpunkten hingewiesen. Hier könnte eine Applikation entwickelt werden, mit der aus WebParts und anderen SharePoint Anwendungen heraus, Endpunkte für vorher definierte Dienste erstellt werden.

Da SharePoint gegenwärtig auf dem .NET Framework 3.5 basiert, wurde nicht betrachtet, welche Unterschiede bei der Entwicklung zu erwarten sind, sobald auch das .NET Framework 4.0 genutzt werden kann. Im Rahmen dieser Arbeit wurde nur betrachtet, wie SharePoint als Applikationsplattform genutzt werden kann. Eine vergleichbare Auswertung von anderen Collaborationsplattformen kann die Auswahl einer Plattform im Falle einer Neueinführung erheblich erleichtern.

Collaborationslösungen werden oftmals mit dem Ziel eingeführt, den Strom an Informationen, der auf einen Benutzer einfließt, zu koordinieren und den Benutzer vor einer Überkommunikation zu bewahren. Oftmals werden aber nur weitere Informationskanäle geöffnet, welche vom Benutzer beachtet werden müssen. Diese Herausforderung muss angegangen werden, damit die Mitarbeiter des Unternehmens nicht im Informationschaos untergehen.

## Literaturverzeichnis

- [Andrew 2009] ANDREW, Paul: *Collaborative Workflows - Improvements in SharePoint 2010*. MSDN Magazin, November 2009. – <http://msdn.microsoft.com/de-de/magazine/ee335710.aspx>
- [Angermeier 2010] ANGERMEIER, Dr. Georg: *Collaboration*. 2010. – <http://www.projektmagazin.de/glossar/gl-0820.html>
- [Baginski 2010] BAGINSKI, Todd: *Business Connectivity Services Part 1*. SharePoint Pro, November 2010. – <http://www.sharepointpromag.com/article/sharepoint/Business-Connectivity-Services-Part-I.aspx>
- [Bauer u. Rief 2010] BAUER, Wilhelm ; RIEF, Stefan: *Green Office, Ökonomische und ökologische Potenziale nachhaltiger Arbeits- und Bürogestaltung*. 2010
- [Bube 2010] BUBE, Lars: *Collaboration als Schlüsseltechnologie*. 2010. – <http://www.crn.de/datacenter/artikel-83800.html>
- [Comundus ] COMUNDUS: *Mitarbeiterportal und Collaboration*. – <http://www.comundus.com/leistungen/Mitarbeiterportal/>
- [Dreyßig 2010] DREYSSIG, Alexander: *Collaboration in deutschen Unternehmen*. 2010. – <http://www.computerwoche.de/software/office-collaboration/1938086/>
- [Driscill u. a. 2010] DRISGILL, Randy ; ROSS, John ; SANFORD, Jacob J. ; STUBB, Paul: *The Client Object Model and jQuery*. MSDN, 2010. – <http://msdn.microsoft.com/en-us/library/gg701783.aspx>
- [Galuschka 2009] GALUSCHKA, Christian: *Collaboration schafft Nähe in der Distanz*. 2009. – <http://www.computerwoche.de/software/crm/1912672/>
- [Heck 2010] HECK, Mike: *Review: Microsoft SharePoint Server 2010*. Network World, Mai 2010. – <http://www.networkworld.com/reviews/2010/051710-microsoft-sharepoint-server-2010-test.html>
- [Hey 2010] HEY, Martin: *Mehrsprachige Websites in SharePoint 2010*. Comundaro Techblog, Juni 2010. – <http://www.comunardo.de/home/techblog/2010/06/17/mehrsprachige-websites-in-sharepoint-2010/>
- [Joining Dots 2006] JOINING DOTS: *SharePoint History*. 2006. – <http://www.joiningdots.net/blog/2006/08/sharepoint-history.html>
- [Jung 2010] JUNG, Jakob: *Vier Alternativen: Es muss nicht immer SharePoint sein*. ZD NET, April 2010. – [http://www.zdnet.de/it\\_business\\_technik\\_vier\\_alternativen\\_es\\_muss\\_nicht\\_immer\\_sharepoint\\_sein\\_story-11000009-41530521-1.htm](http://www.zdnet.de/it_business_technik_vier_alternativen_es_muss_nicht_immer_sharepoint_sein_story-11000009-41530521-1.htm)
- [Kotz u. Hölzl 2009] KOTZ, Jürgen ; HÖLZL, Stephanie: *WCF Windows Communication Foundation - Verteilte Anwendungsentwicklung mit der Microsoft Kommunikationsplattform*. Addison Wesley, 2009. – 13–41 S.



- [Krause 2010] KRAUSE, Jörg: *SharePoint für Entwickler*. dot NET Magazin, August 2010. – <http://it-republik.de/dotnet/artikel/SharePoint-fuer-Entwickler-3272.html>
- [Krause u. a. 2010a] KRAUSE, Jörg ; LANGHIRT, Christian ; STERFF, Alexander ; PEHLKE, Bern ; DÖRING, Martin: *SharePoint 2010 as a Development Platform*. APress, 2010. – 90–112 S.
- [Krause u. a. 2010b] KRAUSE, Jörg ; LANGHIRT, Christian ; STERFF, Alexander ; PEHLKE, Bern ; DÖRING, Martin: *SharePoint 2010 as a Development Platform*. APress, 2010. – 483 S.
- [Krause u. a. 2010c] KRAUSE, Jörg ; LANGHIRT, Christian ; STERFF, Alexander ; PEHLKE, Bern ; DÖRING, Martin: *SharePoint 2010 as a Development Platform*. APress, 2010. – 270–279 S.
- [Krause u. a. 2010d] KRAUSE, Jörg ; LANGHIRT, Christian ; STERFF, Alexander ; PEHLKE, Bern ; DÖRING, Martin: *SharePoint 2010 as a Development Platform*. APress, 2010. – 295–296 S.
- [Krause u. a. 2010e] KRAUSE, Jörg ; LANGHIRT, Christian ; STERFF, Alexander ; PEHLKE, Bern ; DÖRING, Martin: *SharePoint 2010 as a Development Platform*. APress, 2010. – 948–955 S.
- [Krause u. a. 2010f] KRAUSE, Jörg ; LANGHIRT, Christian ; STERFF, Alexander ; PEHLKE, Bern ; DÖRING, Martin: *SharePoint 2010 as a Development Platform*. APress, 2010. – 964–973 S.
- [Krause u. a. 2010g] KRAUSE, Jörg ; LANGHIRT, Christian ; STERFF, Alexander ; PEHLKE, Bern ; DÖRING, Martin: *SharePoint 2010 as a Development Platform*. APress, 2010. – 1022–1032 S.
- [Krist 2010] KRIST, Joel: *Creating Web Databases with Access 2010 and Access Services*. MSDN, Februar 2010. – <http://msdn.microsoft.com/en-us/library/ff402351.aspx>
- [Lewis 2009] LEWIS, Ric: *Microsoft Access - Publish to SharePoint*. Microsoft Office Blog, November 2009. – <http://blogs.office.com/b/microsoft-access/archive/2009/11/23/publish-to-sharepoint-part-1.aspx>
- [Microsoft 2010a] MICROSOFT: *Improving the reach and manageability of Access 2010 database applications with Access Services*. White Paper, Januar 2010. – <http://technet.microsoft.com/en-us/library/ff397963.aspx>
- [Microsoft 2010b] MICROSOFT: *SharePoint 2010 Editionen im Vergleich*. 2010. – <http://sharepoint.microsoft.com/de-de/buy/Seiten/Editions-Comparison.aspx>
- [Microsoft 2010c] MICROSOFT: *SharePoint 2010 Germany Homepage*. 2010. – <http://sharepoint.microsoft.com/de-de/Seiten/default.aspx>
- [Microsoft 2010d] MICROSOFT: *SharePoint 2010 Leistungsvermögen*. 2010. – <http://sharepoint.microsoft.com/de-de/product/capabilities/Seiten/default.aspx>

- [Microsoft 2011a] MICROSOFT: *Build an Access database to share on the Web*. Office Access Support, 2011. – <http://office.microsoft.com/en-us/access-help/build-an-access-database-to-share-on-the-web-HA010356866.aspx>
- [Microsoft 2011b] MICROSOFT: *Microsoft Access Services on SharePoint 2010 does not support Custom Workflows*. Microsoft Knowledge Base, März 2011. – <http://support.microsoft.com/kb/2528459>
- [Microsoft SharePoint Team Blog 2009] MICROSOFT SHAREPOINT TEAM BLOG: *Walkthrough of creating a SharePoint 2010 external list using Visual Studio 2010 Beta*. Microsoft, Februar 2009. – <http://sharepoint.microsoft.com/blog/Pages/BlogPost.aspx?PageType=4&ListId={72C1C85B-1D2D-4A4A-90DE-CA74A7808184}&pID=444>
- [Moritz 2010] MORITZ, Fabian: *SharePoint ist viel mehr als ein Produkt*. dot NET Magazin, Mai 2010. – <http://it-republik.de/dotnet/news/SharePoint-ist-viel-mehr-als-ein-Produkt---SharePoint-MVP-Fabian-Moritz-im-Gespraech-055432.html>
- [MSDN 2010a] MSDN: *Bereitstellen einer Lösung*. Mai 2010. – <http://msdn.microsoft.com/de-de/library/aa544500.aspx>
- [MSDN 2010b] MSDN: *Elements Scope*. Mai 2010. – <http://msdn.microsoft.com/en-us/library/ms476615.aspx>
- [MSDN 2010c] MSDN: *Erstellen von .NET Verbindungsassemblies und Webdiensten*. Mai 2010. – <http://msdn.microsoft.com/de-de/library/ff464398.aspx>
- [MSDN 2010d] MSDN: *Excel Services Overview*. Mai 2010. – <http://msdn.microsoft.com/en-us/library/ms546696.aspx>
- [MSDN 2010e] MSDN: *Feature Events*. Mai 2010. – <http://msdn.microsoft.com/en-us/library/ms469501.aspx>
- [MSDN 2010f] MSDN: *How to: Install and Configure SharePoint Integration on Multiple Servers*. Mai 2010. – <http://msdn.microsoft.com/en-us/library/bb677365.aspx>
- [MSDN 2010g] MSDN: *SharePoint 2010 SDK*. 2010. – <http://msdn.microsoft.com/en-us/library/ee557253.aspx>
- [MSDN 2010h] MSDN: *SharePoint Foundation REST Schnittstelle*. Mai 2010. – <http://msdn.microsoft.com/de-de/library/ff521587.aspx>
- [MSDN 2010i] MSDN: *Unterstützung von Business Connectivity Services Tools*. Mai 2010. – <http://msdn.microsoft.com/de-de/library/ee556789.aspx>
- [MSDN 2010j] MSDN: *Walkthrough: Creating a Basic Web Part*. 2010. – <http://msdn.microsoft.com/en-us/library/ms415817.aspx>
- [MSDN 2010k] MSDN: *WCF Dienste in SharePoint Foundation 2010*. Mai 2010. – <http://msdn.microsoft.com/de-de/library/ff521586.aspx>

- [MSDN 2010l] MSDN: *Web Parts Overview*. 2010. – <http://msdn.microsoft.com/de-de/library/ms432401.aspx>
- [MSDN 2010m] MSDN: *Wo können externe Daten angezeigt werden?* Mai 2010. – <http://msdn.microsoft.com/de-de/library/ee558737.aspx>
- [MSDN 2010n] MSDN: *Workflow Object Model Overview in SharePoint Foundation*. Mai 2010. – <http://msdn.microsoft.com/en-us/library/ms463007.aspx>
- [MSDN 2010o] MSDN: *Workflow Types*. Mai 2010. – <http://msdn.microsoft.com/en-us/library/ms468447.aspx>
- [MSDN 2011a] MSDN: *ASP.NET WebParts Overview*. 2011. – <http://msdn.microsoft.com/en-us/library/hhy9ewf1.aspx>
- [MSDN 2011b] MSDN: *Elements by Scope*. 2011. – <http://msdn.microsoft.com/en-us/library/ms454835.aspx>
- [MSDN 2011c] MSDN: *Manually Creating a Solution*. Januar 2011. – <http://msdn.microsoft.com/en-us/library/aa543741.aspx>
- [MSDN 2011d] MSDN: *Restrictions in Sandboxed Solutions*. Februar 2011. – <http://msdn.microsoft.com/en-us/library/gg615454.aspx>
- [MSDN 2011e] MSDN: *Sandboxed Solutions Architecture*. Februar 2011. – <http://msdn.microsoft.com/en-us/library/ee539417.aspx>
- [Namesnik 2007] NAMESNIK, Joe: *Die Kluft zwischen Collaboration und Kultur*. 2007. – [http://www.iknowledge.biz/fileadmin/user\\_upload/18-19\\_07330\\_1\\_Joe\\_Namesnik\\_ECM\\_05.pdf](http://www.iknowledge.biz/fileadmin/user_upload/18-19_07330_1_Joe_Namesnik_ECM_05.pdf)
- [Niemeier 2010] NIEMEIER, Joachim: *Neue Studie zu Enterprise 2.0 belegt vielfältige Potenziale für die Unternehmen*. 2010. – <http://intranetberater.de/index.php/fachartikel/35-organisation/490-neue-studie-zu-enterprise-20-belegt-vielfaeltige-potenziale-fuer-die-unternehmen>
- [O'Reilly 2005] O'REILLY, Tim: *What Is Web 2.0*. 2005. – <http://oreilly.com/pub/a/web2/archive/what-is-web-20.html>
- [Outcalt 2010] OUTCALT, Adam: *Authenticating to Your External System*. Microsoft Business Connectivity Services Team Blog, März 2010. – <http://blogs.msdn.com/b/bcs/archive/2010/03/12/authenticating-to-your-external-system.aspx>
- [Ovcak 2010] OVCAK, Boris: *Arbeiten mit SharePoint 2010*. 2010. – <http://www.computerwoche.de/software/bi-ecm/1928622/index.html>
- [Paulke u. a. 2008] PAULKE, Sebastian ; SIMONS, Stefan ; STEIMEL, Bernhard: *„Kollaborieren oder Kollabieren?“ Team Collaboration in der Enterprise 2.0*. 2008
- [Peiris u. Mudler 2007] PEIRIS, Chris ; MUDLER, Dennis: *Hosting and Consuming WCF Services*. MSDN, März 2007. – <http://msdn.microsoft.com/en-us/library/bb332338.aspx>

- [Posey 2010] POSEY, Brien: *SharePoint 2010: Gestalten von Abläufen - Arbeiten mit SharePoint Workflows*. TechNet Magazin, Juli 2010. – <http://technet.microsoft.com/de-de/magazine/ff848709.aspx>
- [Rohles 2008] ROHLES, Björn: *Social Collaboration: Eine Chance für Unternehmen?* 2008. – <http://www.netzpiloten.de/2008/05/13/social-collaboration-eine-chance-fur-unternehmen/>
- [Rüsche u. Quix 2008] RÜSCHE, Sebastian ; QUIX, Christoph: *Eigene WebParts für Windows SharePoint Services 3 (WSS3) und Microsoft Office SharePoint Server 2007 (MOSS) erstellen*. 2008
- [SaaS-Forum 2010] SAAS-FORUM: *Collaboration-Studie: Virtuelles Arbeiten boomt in deutschen Unternehmen*. 2010. – <http://www.saas-forum.net/blog/collaboration-studie-virtuelles-arbeiten-boomt-in-deutschen-unternehmen/21102010>
- [Steinke 2010] STEINKE, C.: *SharePoint als Lösungsplattform*. TechNet Blogs, Januar 2010. – <http://blogs.technet.com/b/tspiwger/archive/2010/01/14/sharepoint-als-l-sungsplattform.aspx>
- [Stevenson 2009] STEVENSON, Brad: *Overview of Business Connectivity Services*. Microsoft Business Connectivity Services Team Blog, Oktober 2009. – <http://blogs.msdn.com/b/bcs/archive/2009/10/19/overview-of-business-connectivity-services.aspx>
- [TechNet 2010a] TECHNET: *Bereitstellen von Websiteelementen mithilfe von Features (SharePoint Foundation 2010)*. Mai 2010. – <http://technet.microsoft.com/de-de/library/ff607883.aspx>
- [TechNet 2010b] TECHNET: *Configure and deploy WebParts (SharePoint Foundation 2010)*. Mai 2010. – <http://technet.microsoft.com/en-us/library/cc288040.aspx>
- [TechNet 2010c] TECHNET: *Excel Services (Übersicht)(SharePoint Server 2010)*. Mai 2010. – <http://technet.microsoft.com/de-de/library/ee424405.aspx>
- [TechNet 2010d] TECHNET: *Planen und Erstellen von Workflows (SharePoint Server 2010)*. Microsoft, Mai 2010. – <http://technet.microsoft.com/de-de/library/cc263308>.
- [TechNet 2010e] TECHNET: *Workflows (Übersicht)(SharePoint 2010)*. Mai 2010. – <http://technet.microsoft.com/de-de/library/cc263148.aspx>
- [Teper 2009] TEPER, Jeff: *SharePoint History*. 2009. – <http://blogs.msdn.com/b/sharepoint/archive/2009/10/05/sharepoint-history.aspx>
- [Thiessenhusen 2008] THIESSENHUSEN, Mario: *Microsoft SharePoint Technologien und die Einführung in Unternehmen*. 2008. – [http://www.iwinet.net/download/vortrag\\_20081115\\_sharepoint\\_folien.pdf](http://www.iwinet.net/download/vortrag_20081115_sharepoint_folien.pdf)
- [Tylla 2010] TYLLA, Christoph: *Microsoft SharePoint, wie sieht der Markt wirklich aus*. 2010. – <http://intranetberater.de/index.php/fachartikel/35-organisation/519-neue-studie-von-pentadoc-radar-microsoft-sharepoint-wie-sieht-der-markt-wirklich-aus>

[Uytgeerts 2009] UYTGEERTS, Danny: *The Martyrdom of a Solution Architect*. 2009. – [http://www.microsoft.com/belux/architect/issue\\_2/the\\_martyrdom.aspx](http://www.microsoft.com/belux/architect/issue_2/the_martyrdom.aspx)

[Wheeler 2010] WHEELER, Ryan: *SharePoint Feature Receivers - the hidden details*. Pentalogic sharePoint Blog, Juni 2010. – <http://blog.pentalogic.net/2010/06/sharepoint-feature-receivers-events-details/>

## Quellcode Verzeichnis

L.1	Usercontrol mit Zugriffsmöglichkeit auf die WebPart Eigenschaften	37
L.2	Visual WebPart mit notwendigen Anpassungen . . . . .	38
L.3	ItemMetadata.xml . . . . .	54
L.4	Excel Diagramm per REST API abrufen . . . . .	81
L.1	Beispiel eines Custom Timer Jobs . . . . .	100
L.2	Beispiel eines FeatureReceivers . . . . .	104
L.3	Eigenschaften eines WebParts . . . . .	110
L.4	Interface der WebParts . . . . .	111
L.5	Provider WebPart . . . . .	111
L.6	Consumer WebPart . . . . .	112
L.7	Programmcode der Kartenanwendung . . . . .	113
L.8	Definitionsdatei der Kartenanwendung . . . . .	119
L.9	Beispiel eines FeatureReceivers . . . . .	119
L.10	Definition des ASPX Formulars . . . . .	124
L.11	Quellcode des ASPX Formulars . . . . .	125
L.12	Elements.xml . . . . .	127
L.13	Definition der Felder . . . . .	128
L.14	Definition des Content Types . . . . .	129
L.15	Definition des ASPX Task Formulars . . . . .	130
L.16	Quellcodes des ASPX Task Formulars . . . . .	131
L.17	Erstellen eines Tasks mit Content Type . . . . .	135
L.18	Schnittstelle einer Custom Workflow Activity . . . . .	137
L.19	Quellcode einer Custom Workflow Activity . . . . .	138
L.20	Programmcode des Usercontrol . . . . .	141
L.21	Definition des Usercontrol . . . . .	143
L.22	Schnittstelle des Dienstes . . . . .	145
L.23	Programmcode des Dienstes . . . . .	146
L.24	Berechtigungsobjekt des Dienstes . . . . .	152
L.25	Rückgabeobjekt des Dienstes . . . . .	152
L.26	Definition des Service Hosts . . . . .	153
L.27	Schnittstelle des Dienstes . . . . .	153
L.28	Programmcode des Dienstes . . . . .	154
L.29	Schnittstelle des Dienstes . . . . .	156
L.30	Programmcode des Dienstes . . . . .	157
L.31	Berechtigungsobjekt des Dienstes . . . . .	159
L.32	web.config des Dienstes . . . . .	159

# A Anhang

## A.1 SharePoint 2010 Development Platform Stack

### SharePoint 2010 Development Platform Stack

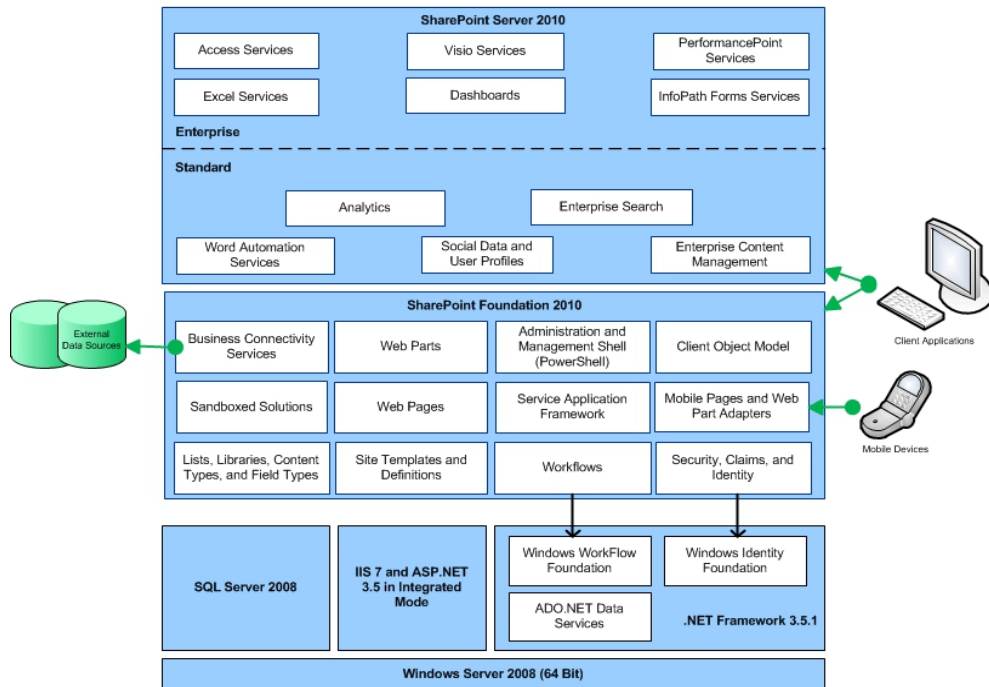


Abbildung A.1: SharePoint 2010 Development Platform Stack (MSDN [2010g])

## A.2 Historie von SharePoint

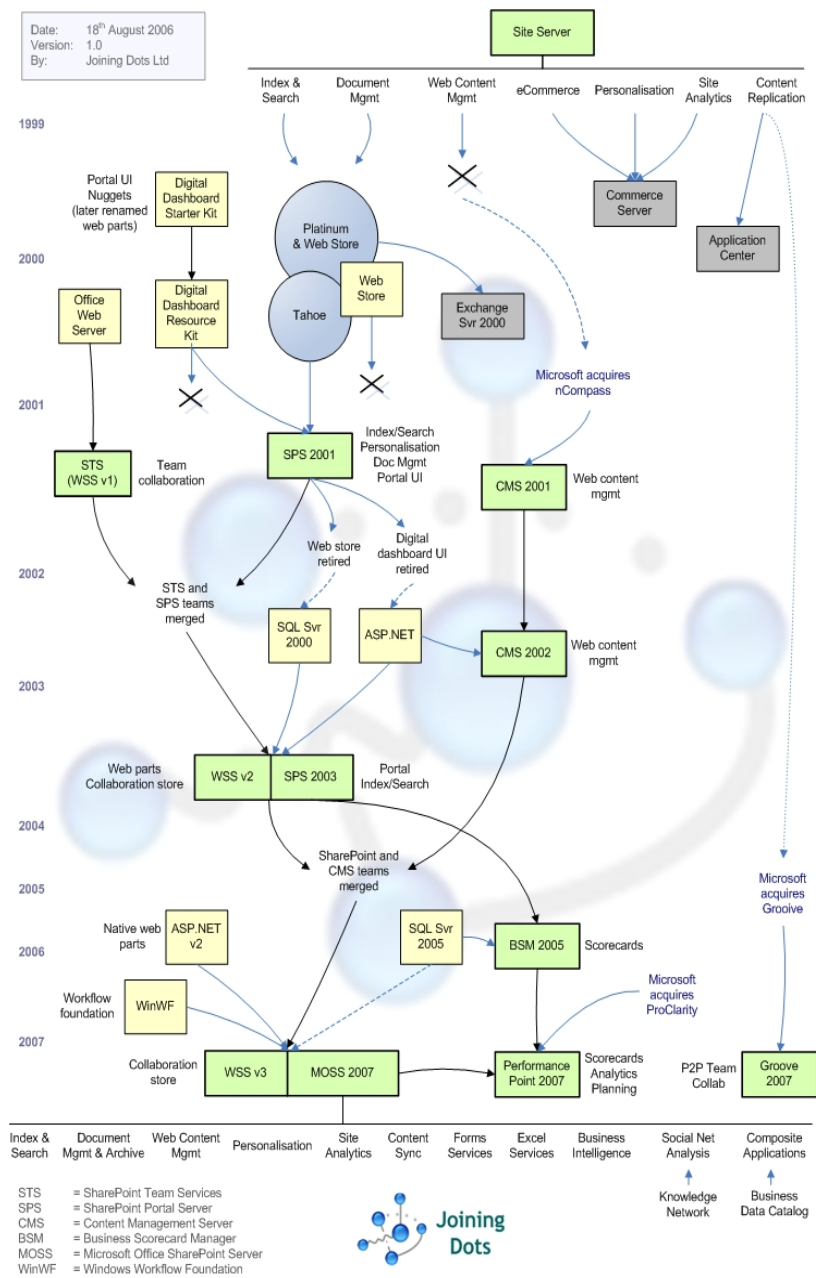


Abbildung A.2: Die Geschichte von SharePoint (Joining Dots [2006])



## A.3 External Content Type im SharePoint Designer

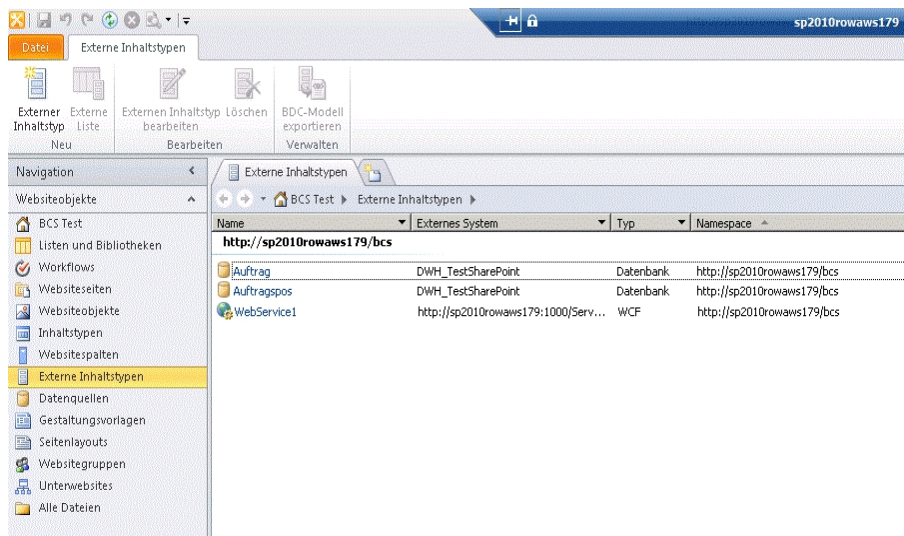


Abbildung A.3: Mit dem SharePoint Designer können External Content Types erstellt werden (eigene Darstellung)

## A.4 Temporäre BCS Verbindungen

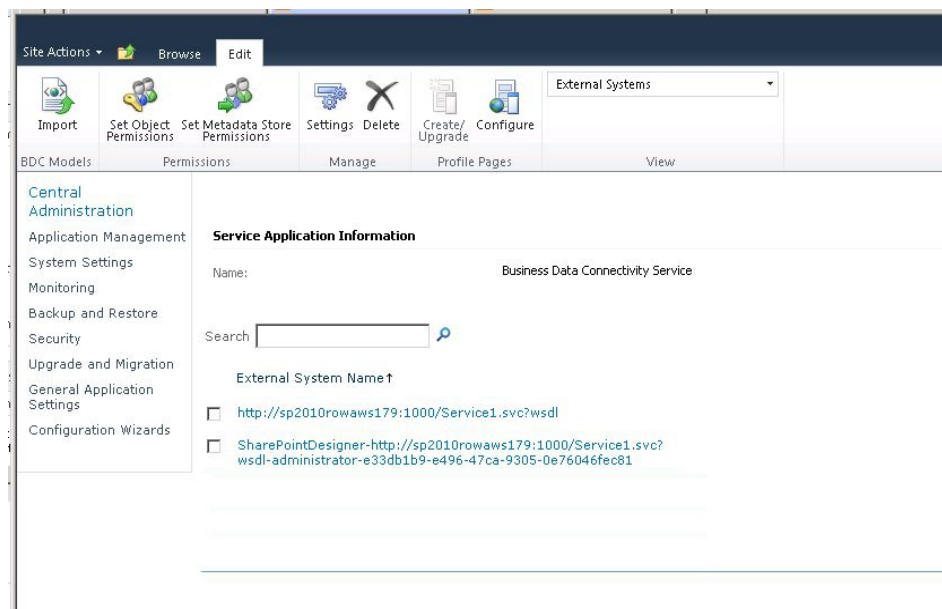
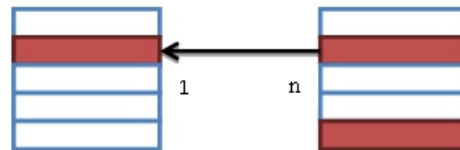


Abbildung A.4: Temporäre Verbindungen zum Drittsystem bleiben nach Beendigung der Entwicklung zurück (eigene Darstellung)



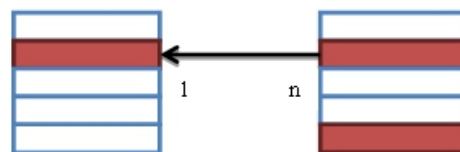
## A.6 Beziehungen zwischen External Content Types

### Fall 1: Fremdschlüsselbeziehung in Datenbank



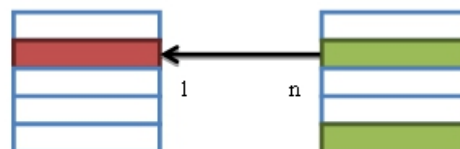
Beide ECT bieten eine Löschoperation an, ein Datensatz aus dem primären ECT (linke Tabelle) wird gelöscht. Es gelten die Einschränkungen der Fremdschlüsselbeziehungen, die in der Datenbank definiert sind. Entweder wird Löschoperation verhindert, oder die abhängigen Datensätze gelöscht/aktualisiert.

### Fall 2: Fremdschlüsselbeziehung in Datenbank



Nur der primäre ECT (linke Tabelle) bietet eine Löschoperation an. Der abhängige ECT bietet nur Leseoperationen. Es wird ein Datensatz aus dem primären ECT (linke Tabelle) gelöscht. Es gelten die Einschränkungen der Fremdschlüsselbeziehungen, die in der Datenbank definiert sind. Entweder wird die Löschoperation verhindert, oder die abhängigen Datensätze gelöscht/aktualisiert.

### Fall 3: Beziehung nur im BDC Modell definiert



Es werden zwei Entitäten im BDC Modell verknüpft, für die keine Beziehung in der Datenbank definiert ist oder die aus verschiedenen Systemen stammen. Beide ECT bieten eine Löschoperation an. Es wird ein Datensatz aus dem primären ECT (linke Tabelle) gelöscht. Abhängige Datensätze werden nicht gelöscht.

Abbildung A.6: Auswirkung von Löschoperationen in den BCS (eigene Darstellung)

## A.7 Definition eines Custom Timer Jobs

```
1 namespace Rowa.SharePoint2010.Administration.Jobs
2 {
3     using System;
4     using System.Collections.Generic;
5     using System.Data;
6     using System.Data.SqlClient;
7     using System.Linq;
8     using System.Text;
9     using Microsoft.SharePoint;
10    using Microsoft.SharePoint.Administration;
11
12    /// <summary>
13    /// Klasse des Timer Job
14    /// </summary>
15    public class SynchPersonal2SPList : SPJobDefinition
16    {
17        /// <summary>
18        /// Name der Site Collection
19        /// </summary>
20        private const string SITE = "/";
21
22        /// <summary>
23        /// Name der WebSite
24        /// </summary>
25        private const string WEB = "MyWeb";
26
27        /// <summary>
28        /// Konstruktor des Jobs. Es wird automatisch
29        /// der Konstruktor der Basisklasse aufgerufen
30        /// </summary>
31        public SynchPersonal2SPList() : base()
32        {
33        }
34
35        /// <summary>
36        /// Konstruktor des Jobs. Es wird automatisch
37        /// der Konstruktor der Basisklasse aufgerufen
38        /// </summary>
39        /// <param name="jobName">Name des Jobs</param>
40        /// <param name="service">Service Instanz</param>
41        /// <param name="server">Server Instanz</param>
```

```

42     /// <param name="targetType">Typ des Jobs</param>
43     public SynchPersonal2SPList(string jobName, SPService
        service, SPServer server, SPJobLockType targetType)
        : base(jobName, service, server, targetType)
44     {
45     }
46
47     /// <summary>
48     /// Konstruktor des Jobs. Es wird automatisch
49     /// der Konstruktor der Basisklasse aufgerufen
50     /// </summary>
51     /// <param name="jobName">Name des Jobs</param>
52     /// <param name="webApplication">Name der Web
        Application </param>
53     public SynchPersonal2SPList(string jobName,
        SPWebApplication webApplication)
54         : base(jobName, webApplication, null,
        SPJobLockType.ContentDatabase)
55     {
56     }
57
58     /// <summary>
59     /// Methode wird durch den Timer Dienst
60     /// aufgerufen. Hier werden die
61     /// eigentlichen Jobanweisungen implementiert
62     /// </summary>
63     /// <param name="targetInstanceId">ID der Content
        Datenbank</param>
64     public override void Execute(Guid targetInstanceId)
65     {
66         SPWebApplication webApp =
            (SPWebApplication) this.Parent;
67         SPContentDatabase contentDB =
            webApp.ContentDatabases[targetInstanceId];
68
69         DataTable employeeTable = this.GetEmployee();
70         SPList employeeList =
            contentDB.Sites["/"].AllWebs["/"].Lists["Mitarbeiter"];
71         SPListItemCollection employeeCollection =
            employeeList.Items;
72
73         foreach (DataRow row in employeeTable.Rows)
74         {

```

```
75 // Prüfen ob der Mitarbeiter schon angelegt ist
76 bool newEmployee = true;
77 string defaultValue = string.Empty;
78
79 foreach (SPListItem item in employeeCollection)
80 {
81     if ((item["Personalnummer"] ??
82         defaultValue).ToString() ==
83         (row["Personalnummer"] ??
84         defaultValue).ToString())
85     {
86         // Bereits in Liste, prüfen ob ein
87         // Update erforderlich ist
88         newEmployee = false;
89         bool needUpdate = false;
90         foreach (DataColumn column in
91             employeeTable.Columns)
92         {
93             if ((item[column.ColumnName] ??
94                 defaultValue).ToString() !=
95                 (row[column] ??
96                 defaultValue).ToString())
97             {
98                 item[column.ColumnName] =
99                     row[column];
100                 needUpdate = true;
101             }
102         }
103
104         if (needUpdate == true)
105         {
106             item.Update();
107         }
108
109         break;
110     }
111 }
112
113 if (newEmployee == true)
114 {
115     SPListItem employee =
116         employeeList.Items.Add();
```

```

107         foreach (DataColumn column in
            employeeTable.Columns)
108         {
109             employee[column.ColumnName] =
                row[column.ColumnName];
110         }
111
112         employee.Update();
113     }
114 }
115 }
116
117 /// <summary>
118 /// Auslesen der Personalstammtabelle aus der Datenbank
119 /// </summary>
120 /// <returns>Tabelle mit Personaldaten</returns>
121 public DataTable GetEmployee()
122 {
123     SqlConnection connection = new SqlConnection("Data
        Source=MyDBServer;Initial Catalog=MyDB; User
        ID=MyUser; Password=MyPW; Integrated
        Security=false");
124     DataTable returnTable = new DataTable();
125
126     SqlDataAdapter dataAdapter = new
        SqlDataAdapter("Select * from Mitarbeiter",
            connection);
127     dataAdapter.Fill(returnTable);
128
129     return returnTable;
130 }
131 }
132 }

```

Listing L.1: Beispiel eines Custom Timer Jobs

## A.8 Definition eines FeatureReceivers zum Erstellen Custom Timer Jobs

```

1 namespace
    Rowa.SharePoint2010.Administration.Jobs.Features.Feature1
2 {
3     using System;
4     using System.Data;
5     using System.Data.SqlClient;
6     using System.Runtime.InteropServices;
7     using System.Security.Permissions;
8     using Microsoft.SharePoint;
9     using Microsoft.SharePoint.Administration;
10    using Microsoft.SharePoint.Security;
11
12    /// <summary>
13    /// Event Receiver. Wird ausgeführt, wenn das Feature
14    /// aktiviert oder deaktiviert wird.
15    /// </summary>
16    [Guid("04c3ca95-e857-4e93-aa6a-b2e40d9020ec")]
17    public class SynchPersonal2SPListEventReceiver :
18        SPFeatureReceiver
19    {
20        /// <summary>
21        /// Name des Timer Job
22        /// </summary>
23        private const string JobName = "Mitarbeiterliste
24            synchronisieren";
25
26        /// <summary>
27        /// Name der WebSite, auf der die Liste angelegt werden
28        /// soll
29        /// </summary>
30        private const string Web = "MyWeb";
31
32        /// <summary>
33        /// Name der Liste, in die synchronisiert werden soll
34        /// </summary>
35        private const string ListName = "Mitarbeiter";
36
37        /// <summary>
38        /// Beschreibung der Liste, in die synchronisiert
39        /// werden soll

```



```
35     /// </summary>
36     private const string ListDesc = "Mitarbeiterliste";
37
38     /// <summary>
39     /// Wird ausgeführt, wenn das Feature aktiviert wird.
40     /// Hier wird der
41     /// Timer Job und die Zielliste erstellt.
42     /// </summary>
43     /// <param name="properties">Eigenschaften des
44     /// ausgelösten Events</param>
45     public override void
46         FeatureActivated(SPFeatureReceiverProperties
47         properties)
48     {
49         Guid listGuid = new Guid();
50         try
51         {
52             using (SPSite site =
53                 (SPSite)properties.Feature.Parent)
54             {
55                 // Anlegen der Liste
56                 listGuid = this.CreateSPList(site);
57
58                 // evtl. vorhandener Job löschen
59                 foreach (SPJobDefinition job in
60                     site.WebApplication.JobDefinitions)
61                 {
62                     if (job.Name == JobName + "_" +
63                         site.ToString())
64                     {
65                         job.Delete();
66                         break;
67                     }
68                 }
69
70                 // Definition des neuen Jobs
71                 SynchPersonal2SPList jobdesc = new
72                     SynchPersonal2SPList(JobName + "_" +
73                     site.ToString(), site.WebApplication);
74                 jobdesc.Title = JobName + "_" +
75                     site.ToString();
76                 SPMinuteSchedule schedule = new
77                     SPMinuteSchedule();
78             }
79         }
80     }
```

```
67         schedule.BeginSecond = 0;
68         schedule.EndSecond = 59;
69         schedule.Interval = 5;
70         jobdesc.Schedule = schedule;
71         jobdesc.Update();
72     }
73 }
74 catch (Exception ex)
75 {
76     try
77     {
78         this.DeleteList((SPSite)properties.Feature.Parent,
79             listGuid);
80     }
81     catch
82     {
83     }
84     throw ex;
85 }
86 }
87
88 /// <summary>
89 /// Methode wird ausgeführt, wenn das Feature
90 /// deaktiviert wird.
91 /// Die Methode löscht die Jobdefinition und die Liste
92 /// </summary>
93 /// <param name="properties">Eigenschaften des
94 /// ausgelösten Events</param>
95 public override void
96     FeatureDeactivating(SPFeatureReceiverProperties
97         properties)
98 {
99     try
100     {
101         SPSite site = (SPSite)properties.Feature.Parent;
102         foreach (SPJobDefinition job in
103             site.WebApplication.JobDefinitions)
104         {
105             if (job.Name == JobName + "_" +
106                 site.ToString())
107             {
108                 job.Delete();
109             }
110         }
111     }
112     catch { }
113 }
```

```

103             break;
104         }
105     }
106
107     // Liste löschen
108     this.DeleteList(site,
109                     site.AllWebs[Web].Lists[ListName].ID);
110 }
111 catch
112 {
113 }
114
115 /// <summary>
116 /// Methode zum Erstellen der Zielliste in der Root
117   WebSite der Site Collection
118 /// </summary>
119 /// <param name="site">Site Collection, in der die
120   Liste angelegt werden soll.</param>
121 /// <returns>eindeutiger Bezeichner der neuen
122   Liste</returns>
123 private Guid CreateSPList(SPSite site)
124 {
125     SqlConnectionStringBuilder connectionString = new
126         SqlConnectionStringBuilder();
127     SqlConnection connection = new SqlConnection();
128     SqlCommand command = new SqlCommand();
129     DataTable tempTable = new DataTable();
130
131     connectionString.DataSource = "MyDBServer";
132     connectionString.InitialCatalog = "MyDB";
133     connectionString.IntegratedSecurity = true;
134     connection.ConnectionString =
135         connectionString.ConnectionString;
136
137     command.CommandText = "Select Top 1 * from
138         Mitarbeiter";
139     SqlDataAdapter dataAdapter = new
140         SqlDataAdapter(command.CommandText, connection);
141     dataAdapter.Fill(tempTable);
142
143     Guid newListGuid =
144         site.AllWebs[Web].Lists.Add(ListName, ListDesc,

```

```

        SPListTemplateType.GenericList);
137     SPList newList =
        site.AllWebs[Web].Lists[newListGuid];

138
139     foreach (DataColumn column in tempTable.Columns)
140     {
141         newList.Fields.Add(column.ColumnName,
            this.GetFieldType(column.DataType), false);
142     }
143
144     newList.Update();
145
146     return newListGuid;
147 }
148
149 /// <summary>
150 /// Methode löscht die angegebene Liste aus der Root
    WebSite der
151 /// Site Collection
152 /// </summary>
153 /// <param name="site">Site Collection der Liste </param>
154 /// <param name="listGuid">eindeutiger Bezeichner der
    Liste </param>
155 private void DeleteList(SPSite site, Guid listGuid)
156 {
157     try
158     {
159         site.AllWebs[Web].Lists.Delete(listGuid);
160     }
161     catch
162     {
163     }
164 }
165
166 /// <summary>
167 /// Methode übersetzt den .NET Typ in einen
168 /// SharePoint Datentyp
169 /// </summary>
170 /// <param name="type">zu übersetzender Typ</param>
171 /// <returns>SharePoint Typ</returns>
172 private SPFieldType GetFieldType(System.Type type)
173 {
174     if (type.GetType() == typeof(string))

```

```
175         {
176             return SPFieldType.Text;
177         }
178
179         if (type.GetType() == typeof(int) || type.GetType()
180             == typeof(double) || type.GetType() ==
181             typeof(float))
182         {
183             return SPFieldType.Number;
184         }
185
186         if (type.GetType() == typeof(DateTime))
187         {
188             return SPFieldType.DateTime;
189         }
190
191         return SPFieldType.Text;
192     }
```

Listing L.2: Beispiel eines FeatureReceivers

## A.9 WebPart Eigenschaften in der Tool Pane

```

1 namespace WebParts.WebPart1
2 {
3     [ToolboxItemAttribute(false)]
4     public class WebPart1 : WebPart
5     {
6         protected string eMail;
7
8         [Personalizable(PersonalizationScope.User),
9         WebBrowsable(true),
10        WebDisplayName("E-Mail"),
11        WebDescription("E-Mail Adresse, an die eine Nachricht
12        versendet werden soll")]
13        public string EMail
14        {
15            get{ return eMail; }
16            set{ eMail = value; }
17        }
18        ...
19    }
20 }
21 }

```

Listing L.3: Eigenschaften eines WebParts

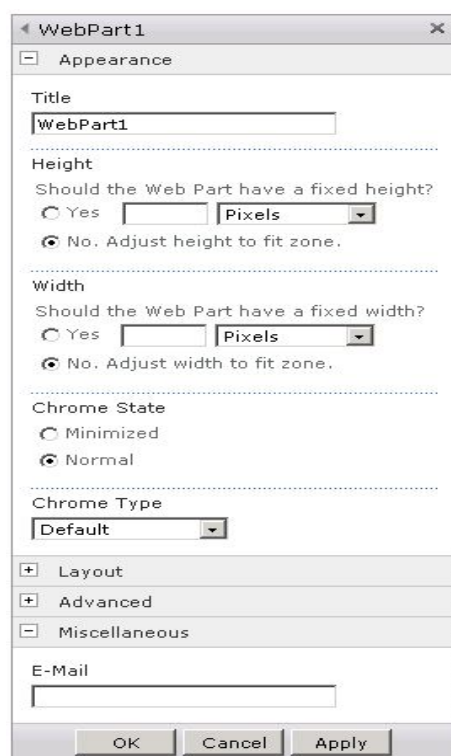


Abbildung A.7: WebPart Eigenschaften in der Tool Pane (eigene Darstellung)

## A.10 Verbundene WebParts

```
1 namespace Connectable
2 {
3     public interface IMitarbeiter
4     {
5         string Mitarbeitername { get; set; }
6     }
7 }
```

Listing L.4: Interface der WebParts

```
1 namespace Connectable
2 {
3     public class Provider : WebPart, IMitarbeiter
4     {
5         private string mitarbeitername;
6
7         [Personalizable()]
8         public string Mitarbeitername
9         {
10             get{ return mitarbeitername; }
11             set{ mitarbeitername = value; }
12         }
13
14         [ConnectionProvider("Mitarbeitername")]
15         public IMitarbeiter Mitarbeiter()
16         {
17             return this;
18         }
19     }
20 }
```

Listing L.5: Provider WebPart

```

1 namespace Connectable
2 {
3     [ToolboxItemAttribute(false)]
4     public class Consumer : WebPart
5     {
6         private IMitarbeiter mitarbeiterProvider;
7
8         [ConnectionConsumer("Mitarbeitername")]
9         public void MitarbeiternameConsumer(IMitarbeiter
10             provider)
11         {
12             mitarbeiterProvider = provider;
13         }
14 }

```

Listing L.6: Consumer WebPart



Abbildung A.8: Verbindung zwischen zwei WebParts (eigene Darstellung)

## A.11 Grafischer Silverlight Designer in Visual Studio

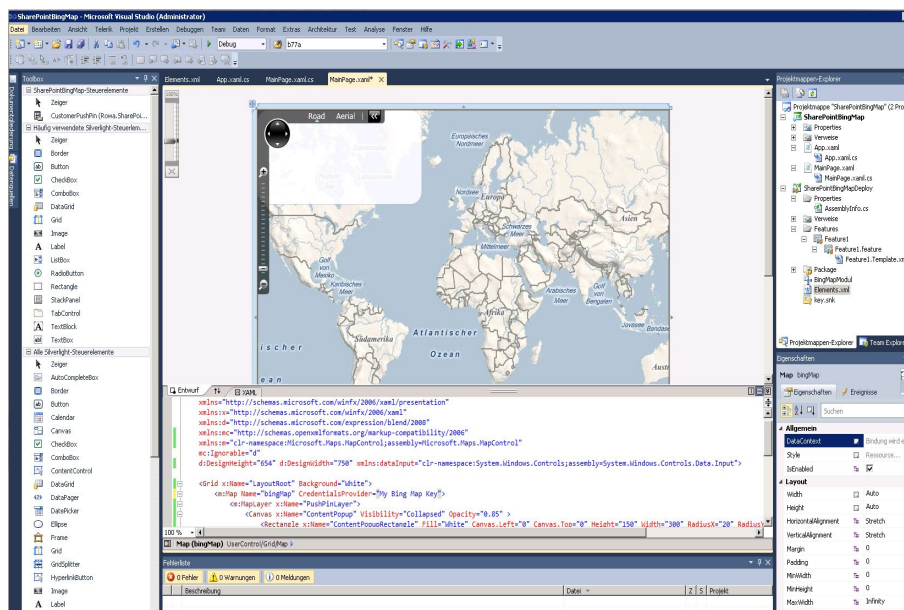


Abbildung A.9: Grafischer Silverlight Designer (eigene Darstellung)



## A.12 SharePoint Silverlight Anwendung

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Net;
5 using System.Windows;
6 using System.Windows.Controls;
7 using System.Windows.Documents;
8 using System.Windows.Input;
9 using System.Windows.Media;
10 using System.Windows.Media.Animation;
11 using System.Windows.Shapes;
12 using Microsoft.Maps;
13 using Microsoft.Maps.MapControl;
14 using Microsoft.SharePoint.Client;
15
16 namespace Rowa.SharePoint2010.Silverlight.Kundenservice
17 {
18     public partial class MainPage : UserControl
19     {
20         /// <summary>
21         /// Liste mit allen Serviceeinsätzen
22         /// </summary>
23         private List<ListItem> kunden;
24
25         /// <summary>
26         /// Feld zur Aufnahme des aktuellen Benutzers
27         /// </summary>
28         private User currentUser;
29
30         /// <summary>
31         /// Konstruktor der Applikation.
32         /// </summary>
33         public MainPage()
34         {
35             InitializeComponent();
36             GetUser();
37         }
38
39         /// <summary>
40         /// Methode liest den aktuell an SharePoint angemeldeten
41         /// Benutzer aus und überträgt den Wert an das
```

```
42     /// globale Feld.
43     /// </summary>
44     private void GetUser()
45     {
46         /// Benutzerkontext der SharePoint Seite erstellen
47         ClientContext context = new
48             ClientContext(ApplicationContext.Current.Url);
49
50         /// es soll der aktuelle Benutzer geladen werden
51         currentUser = context.Web.CurrentUser;
52         context.Load(currentUser);
53
54         /// Aufruf des Webservice, der den Benutzer ausliest
55         context.ExecuteQueryAsync(OnUserSucceeded, null);
56     }
57
58     /// <summary>
59     /// Methode wird aufgerufen, wenn Auslesen des
60     /// Benutzers erfolgreich war.
61     /// </summary>
62     private void OnUserSucceeded(object sender,
63         ClientRequestSucceededEventArgs e)
64     {
65         LoadData(currentUser.LoginName);
66     }
67
68     /// <summary>
69     /// Methode liest die Daten aus der SharePoint
70     /// Liste mit den Serviceeinsätzen
71     /// </summary>
72     /// <param name="user">Benutzer der Einsätze</param>
73     private void LoadData(string user)
74     {
75         ClientContext context = new
76             ClientContext(ApplicationContext.Current.Url);
77         context.Load(context.Web);
78
79         /// es soll die Liste mit Serviceeinsätzen gelesen
80         /// werden
81         List services =
82             context.Web.Lists.GetByTitle("Serviceeinsätze");
83         context.Load(services);
```

```

80         // CAML Query liest nur die offenen Aufträge des
           aktuellen
81         // Benutzers ein
82         string camlExpr =
           string.Format("<View><Query><Where><And><Eq><FieldRef
           Name='Erledigt' /><Value
           Type='Boolean'>False </Value></Eq><Eq><FieldRef
           Name='Techniker' /><Value
           Type='Text'>{0} </Value>
           </Eq></And></Where></Query></View>", user );

83
84         kunden = services.GetItems(new CamlQuery { ViewXml
           = camlExpr });
85         context.Load(kunden);
86
87         // Laden der Daten aus der SharePoint Liste
88         // Dies muss asynchron geschehen, um das UI nicht
89         // zu blockieren. Methode OnRequestSucceeded wird
90         // aufgerufen, wenn Query erfolgreich gewesen ist.
91         context.ExecuteQueryAsync((OnRequestSucceeded),
           null);
92     }
93
94     /// <summary>
95     /// Methode wird aufgerufen, wenn die Serviceeinsätze
96     /// eingelesen sind.
97     /// </summary>
98     private void OnRequestSucceeded(Object sender,
           ClientRequestSucceededEventArgs e)
99     {
100         // Nur der Hauptthread kann auf die Objekte des UI
           zugreifen.
101         // Daher muss die Methode, welche die UI Objekte
           ändern soll
102         // an den Dispatcher des Hauptthread delegiert
           werden.
103         Dispatcher.BeginInvoke(AddPushPins);
104     }
105
106     /// <summary>
107     /// Methode fügt für jeden Kunden eine
108     /// Markierung in die Karte ein.
109     /// </summary>

```

```
110     private void AddPushPins()
111     {
112         // Liste zum speichern der Geo Informationen
113         // des Kunden
114         List<Location> locations = new List<Location>();
115
116         // Rechteck basierend auf Geo Daten
117         LocationRect locationRect;
118
119         // Für jeden Serviceeinsatz eine Markierung
120         // auf der Karte erstellen
121         foreach (ListItem kunde in kunden)
122         {
123             // neue Kundenmarkierung erstellen
124             CustomerPushPin pin = new CustomerPushPin();
125
126             // Metadaten des Kunden setzen
127             pin.KundenName = kunde["Title"].ToString();
128             pin.Strasse = kunde["Stra_x00df_e"].ToString();
129             ;
130             pin.Ort = kunde["Plz"].ToString() + " " +
131                 kunde["Ort"].ToString();
132             pin.Beschreibung =
133                 kunde["Bezeichnung"].ToString();
134
135             // Geo Daten des Kunden erstellen
136             Location gps = new
137                 Location(Convert.ToDouble(kunde["Longitude"]),
138                     Convert.ToDouble(kunde["Latitude"]));
139             pin.Location = gps;
140
141             // Geo Information der Liste hinzufügen
142             locations.Add(gps);
143
144             // Event Handler der Mausbewegungen definieren
145             pin.MouseEnter += new
146                 MouseEventHandler(pin_MouseEnter);
147             pin.MouseLeave += new
148                 MouseEventHandler(pin_MouseLeave);
149
150             // Markierung dem Kartenlayer hinzufügen
151             PushPinLayer.AddChild(pin, gps);
152         }
153     }
```

```

146         }
147
148         // Rechteck der Geo Koordinaten erstellen
149         locationRect = new LocationRect(locations);
150
151         // Zoom und Fokus entsprechend der Koordinaten
152         // setzen
153         bingMap.SetView(locationRect);
154     }
155
156     /// <summary>
157     /// Event Handler, wenn Maus auf eine Markierung
158     /// positioniert wird
159     /// </summary>
160     void pin_MouseEnter(object sender, MouseEventArgs e)
161     {
162         CustomerPushPin pin = sender as CustomerPushPin;
163         if (pin != null)
164         {
165             // Kartenlayer mit Kundeninformationen (PopUp)
166             // an Markierung positionieren
167             MapLayer.SetPosition(ContentPopup,
168                                 pin.Location);
169
170             // Kundeninformationen zusammensetzen und
171             // ausgeben
172             string contentstring =
173                 pin.KundenName.ToString() + "\n " +
174                 pin.Strasse.ToString() + "\n " +
175                 pin.Ort.ToString() + "\n\n" +
176                 pin.Beschreibung.ToString();
177             ContentPopupText.Text = contentstring;
178
179             // Kartenlayer (PopUp) sichtbar machen
180             ContentPopup.Visibility =
181                 System.Windows.Visibility.Visible;
182         }
183     }
184
185     /// <summary>
186     /// Event Handler, wenn Maus von einer
187     /// Markierung entfernt wird
188     /// </summary>

```

```
181     void pin_MouseLeave(object sender, MouseEventArgs e)
182     {
183         CustomerPushPin pin = sender as CustomerPushPin;
184         ContentPopup.Visibility =
185             System.Windows.Visibility.Collapsed;
186     }
187 }
188
189 /// <summary>
190 /// Klasse definiert die Markierung eines Kunden
191 /// </summary>
192 public class CustomerPushPin : Pushpin
193 {
194     public string KundenName { get; set; }
195     public string Strasse { get; set; }
196     public string Ort { get; set; }
197     public string Beschreibung { get; set; }
198
199     public CustomerPushPin()
200     {
201         this.Name = Guid.NewGuid().ToString();
202         SolidColorBrush scb = new
203             SolidColorBrush(Colors.Blue);
204         this.Background = scb;
205     }
206 }
207
208 /// <summary>
209 /// Klasse mit den Kundendaten
210 /// Dient zum Auslesen der Liste
211 /// </summary>
212 public class Kundendaten
213 {
214     public string Kunde { get; set; }
215     public string Straße { get; set; }
216     public string Plz { get; set; }
217     public string Ort { get; set; }
218     public string Bezeichnung { get; set; }
219     public double Latitude { get; set; }
220     public double Longitude { get; set; }
```

221 }

## Listing L.7: Programmcode der Kartenanwendung

```

1 <UserControl
  x:Class="Rowa.SharePoint2010.Silverlight.Kundenservice.MainPage"
2   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
5   xmlns:mc="http://schemas.openxmlformats.org/
6     markup-compatibility/2006"
7   xmlns:m="clr-namespace:Microsoft.Maps.MapControl;
8     assembly=Microsoft.Maps.MapControl"
9   mc:Ignorable="d"
10  d:DesignHeight="654" d:DesignWidth="750"
11  xmlns:dataInput="clr-namespace:System.Windows.Controls;
12    assembly=System.Windows.Controls.Data.Input">
13
14  <Grid x:Name="LayoutRoot" Background="White">
15    <m:Map Name="bingMap" CredentialsProvider="My Bing Map
16      Key">
17      <m:MapLayer x:Name="PushPinLayer">
18        <Canvas x:Name="ContentPopup"
19          Visibility="Collapsed" Opacity="0.85" >
20          <Rectangle x:Name="ContentPopupRectangle"
21            Fill="White" Canvas.Left="0"
22            Canvas.Top="0" Height="150" Width="300"
23            RadiusX="20" RadiusY="20"/>
24          <StackPanel Canvas.Left="10"
25            Canvas.Top="10">
26            <TextBlock x:Name="ContentPopupText"
27              FontSize="12" FontWeight="Bold"
28              TextWrapping="Wrap" Width="250">
29
30          </TextBlock>
31        </StackPanel>
32      </Canvas>
33    </m:MapLayer>
34  </m:Map>
35  </Grid>
36 </UserControl>

```

## Listing L.8: Definitionsdatei der Kartenanwendung

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <Elements xmlns="http://schemas.microsoft.com/sharepoint/">

```

```

3 <Module Name="BingMapModul">
4 <File Path="BingMapModul\SharePointBingMap.xap"
    Url="SilverlightLibrary/SharePointBingMap.xap" />
5 </Module>
6 </Elements>

```

Listing L.9: Beispiel eines FeatureReceivers

## A.13 Site Workflows im Site Settings Menü

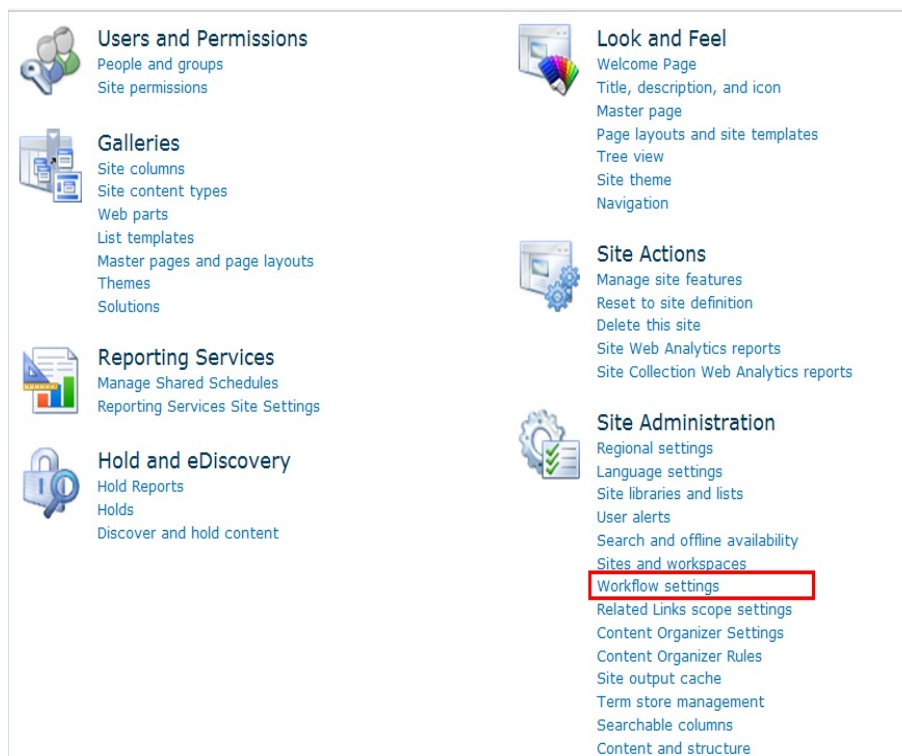


Abbildung A.10: Site Workflows im Site Settings Menü (eigene Darstellung)



## A.14 Task Wizard des SharePoint Designers

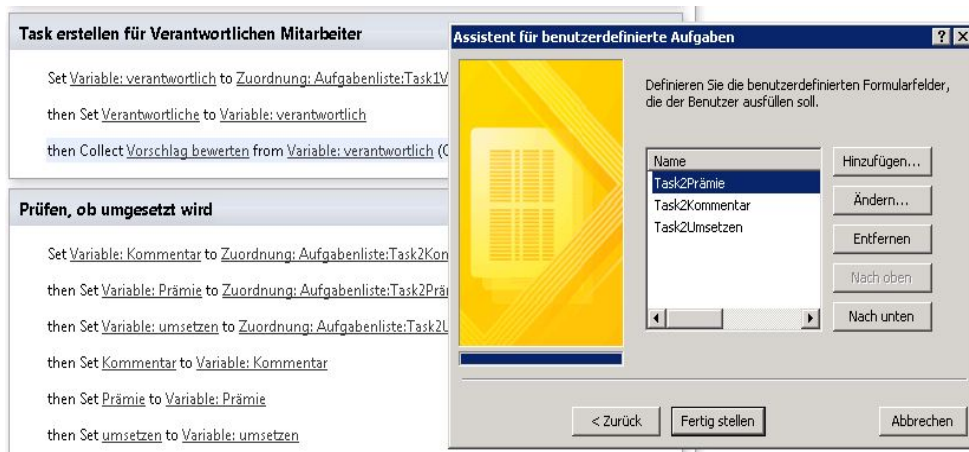


Abbildung A.11: Task Wizard des SharePoint Designers (eigene Darstellung)

## A.15 Personalanforderungsworkflow

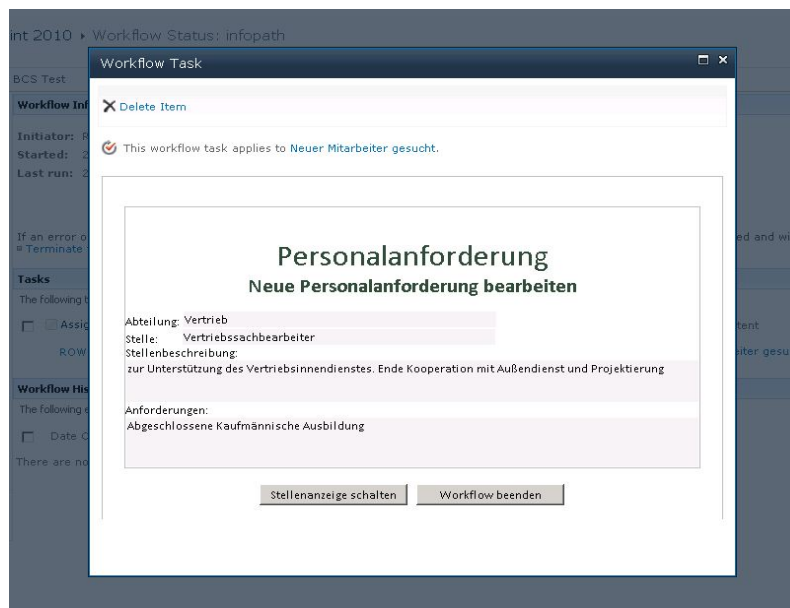


Abbildung A.12: erstes Task Formular (InfoPath) (eigene Darstellung)

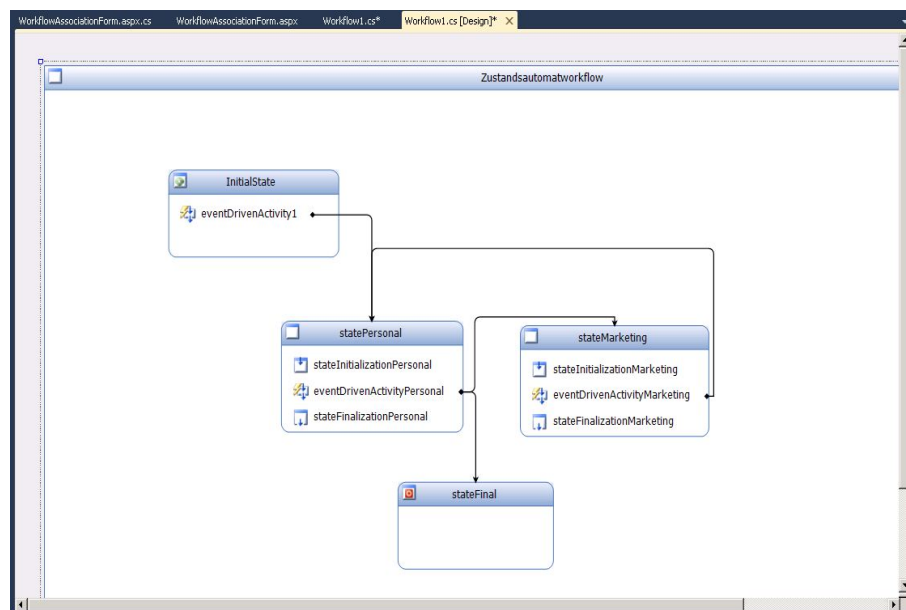


Abbildung A.13: State Machine der Personalanforderung (eigene Darstellung)

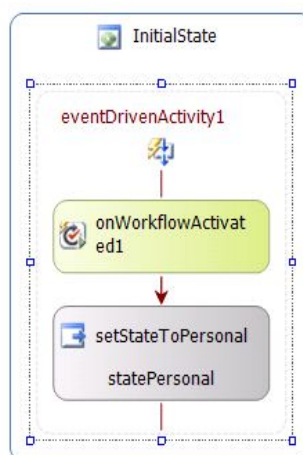


Abbildung A.14: Inhalt des Initialen States (eigene Darstellung)

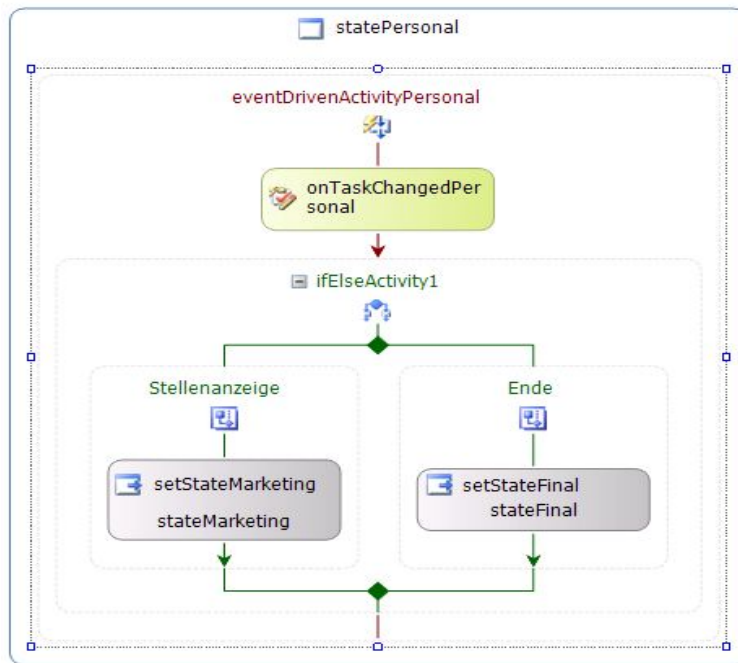


Abbildung A.15: Inhalt der EventDriven Aktivität beim Ändern des Tasks (eigene Darstellung)

## A.16 Definition eines AssociationForm

```

1 <%@ Assembly Name="$SharePoint.Project.AssemblyFullName$" %>
2 <%@ Assembly Name="Microsoft.Web.CommandUI, Version=14.0.0.0,
   Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>
3 <%@ Import Namespace="Microsoft.SharePoint" %>
4 <%@ Import Namespace="Microsoft.SharePoint.ApplicationPages" %>
5 <%@ Register Tagprefix="SharePoint"
   Namespace="Microsoft.SharePoint.WebControls"
   Assembly="Microsoft.SharePoint, Version=14.0.0.0,
   Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>
6 <%@ Register Tagprefix="Utilities"
   Namespace="Microsoft.SharePoint.Utilities"
   Assembly="Microsoft.SharePoint, Version=14.0.0.0,
   Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>
7 <%@ Register Tagprefix="asp" Namespace="System.Web.UI"
   Assembly="System.Web.Extensions, Version=3.5.0.0,
   Culture=neutral, PublicKeyToken=31bf3856ad364e35" %>
8
9 <%@ Page Language="C#"
10     DynamicMasterPageFile="~masterurl/default.master"
11     AutoEventWireup="true"
12     Inherits="Rowa.SharePoint2010.Workflow.Personal.WorkflowAssociationForm"
13     CodeBehind="WorkflowAssociationForm.aspx.cs" %>
14
15 <asp:Content ID="Main" ContentPlaceHolderID="PlaceHolderMain"
   runat="server">
16     <table>
17         <tr>
18             <td>Bearbeiter Personalabteilung </td>
19             <td><SharePoint:PeopleEditor ID="peoplePersonal"
   runat="server" AllowEmpty="false"
   MultiSelect="false" SelectionSet="User" /></td>
20         </tr>
21         <tr>
22             <td>Bearbeiter Marketingabteilung </td>
23             <td><SharePoint:PeopleEditor ID="peopleMarketing"
   runat="server" AllowEmpty="false"
   MultiSelect="false" SelectionSet="User" /></td>
24         </tr>
25     </table>
26     <asp:Button ID="AssociateWorkflow" runat="server"
   OnClick="AssociateWorkflow_Click" Text="Workflow

```

```

1 using System;
2 using System.Globalization;
3 using System.Web;
4 using System.Web.UI;
5 using Microsoft.SharePoint;
6 using Microsoft.SharePoint.Utilities;
7 using Microsoft.SharePoint.WebControls;
8 using Microsoft.SharePoint.Workflow;
9
10 namespace Rowa.SharePoint2010.Workflow.Personal
11 {
12     public partial class WorkflowAssociationForm :
13         LayoutsPageBase
14     {
15         private const int CreateListTryCount = 100;
16         private string historyListDescription =
17             "Benutzerdefinierte Verlaufsliste";
18         private string taskListDescription =
19             "Benutzerdefinierte Aufgabenliste";
20         private string listCreationFailed = "Fehler beim
21             Erstellen der Liste {0}, da bereits eine Liste mit
22             demselben Namen vorhanden ist.";
23         private string workflowAssociationFailed = "Fehler beim
24             Zuordnen der Workflowvorlage. {0}";
25
26         protected void Page_Load(object sender, EventArgs e)
27         {
28             // ...
29         }
30     }
31 }

```

```

1 using System;
2 using System.Globalization;
3 using System.Web;
4 using System.Web.UI;
5 using Microsoft.SharePoint;
6 using Microsoft.SharePoint.Utilities;
7 using Microsoft.SharePoint.WebControls;
8 using Microsoft.SharePoint.Workflow;
9
10 namespace Rowa.SharePoint2010.Workflow.Personal
11 {
12     public partial class WorkflowAssociationForm :
13         LayoutsPageBase
14     {
15         private const int CreateListTryCount = 100;
16         private string historyListDescription =
17             "Benutzerdefinierte Verlaufsliste";
18         private string taskListDescription =
19             "Benutzerdefinierte Aufgabenliste";
20         private string listCreationFailed = "Fehler beim
21             Erstellen der Liste {0}, da bereits eine Liste mit
22             demselben Namen vorhanden ist.";
23         private string workflowAssociationFailed = "Fehler beim
24             Zuordnen der Workflowvorlage. {0}";
25
26         protected void Page_Load(object sender, EventArgs e)
27         {
28             // ...
29         }
30     }
31 }

```

```
21     {
22         InitializeParams();
23     }
24
25     private void PopulateFormFields(SPWorkflowAssociation
26         existingAssociation)
27     {
28         // Optional hier Code einfügen, um die
29         // Formularfelder vorab zu füllen.
30     }
31
32     // Diese Methode wird aufgerufen, wenn der Benutzer auf
33     // die Schaltfläche zum Zuordnen des Workflows klickt.
34     private string GetAssociationData()
35     {
36         PickerEntity entityPersonal =
37             (PickerEntity)peoplePersonal.ResolvedEntities[0];
38         PickerEntity entityMarketing =
39             (PickerEntity)peopleMarketing.ResolvedEntities[0];
40         string returnValue =
41             string.Format("<associationData><personal>{0}
42                 </personal><marketing>{1}</marketing></associationData>",
43                 entityPersonal.EntityData["AccountName"].ToString(),
44                 entityMarketing.EntityData["AccountName"].ToString());
45         return returnValue;
46     }
47
48     protected void AssociateWorkflow_Click(object sender,
49         EventArgs e)
50     {
51         // Optional hier Code einfügen, um vor dem Zuordnen
52         // des Workflows zusätzliche Schritte auszuführen.
53         try
54         {
55             CreateTaskList();
56             CreateHistoryList();
57             HandleAssociateWorkflow();
58             SPUtility.Redirect("WrkSetng.aspx",
59                 SPRedirectFlags.RelativeToLayoutsPage,
60                 HttpContext.Current, Page.ClientQueryString);
61         }
62         catch (Exception ex)
63         {
64         }
```

```

51         SPUtility.TransferToErrorPage( String.Format(
           CultureInfo.CurrentCulture,
           workflowAssociationFailed, ex.Message));
52     }
53 }
54
55 protected void Cancel_Click(object sender, EventArgs e)
56 {
57     SPUtility.Redirect("WrkSetng.aspx",
           SPRedirectFlags.RelativeToLayoutsPage,
           HttpContext.Current, Page.ClientQueryString);
58 }
59
60 }
61 }

```

Listing L.11: Quellcode des ASPX Formulars

## A.17 Definition der InfoPath Task Formulare in der Workflowdefinition

```

1 <?xml version="1.0" encoding="utf-8" ?>
2
3 <!-- Text in eckigen Klammern anpassen.
4 Klammern beim Ausfüllen entfernen, z.B.
5 Name=" [NAME] " ==> Name="MyWorkflow" -->
6
7 <Elements xmlns="http://schemas.microsoft.com/sharepoint/">
8     <Workflow
9         Name="Personalanforderung"
10        Description="State Machine Workflow zur
           Personalanforderung"
11        Id="8c802d3c-aa40-4dfe-9e4a-0cba421150de"
12        CodeBesideClass="Rowa.SharePoint2010.Workflow.Personalanforderung"
13        CodeBesideAssembly="$assemblyname$"
           AssociationUrl="_layouts/Workflow/
           Personalanforderung/WorkflowAssociationForm.aspx"
14        TaskListContentTypeId="0x01080100C9C9515DE4E24001905074F980F93160">
15     <Categories/>
16     <MetaData>
17         <AssociationCategories>List </AssociationCategories>
18         <!-- Tags zur Angabe von InfoPath-Formularen für den
           Workflow; Tags für nicht vorhandene Formulare löschen

```

```

-->
19 <!--<Association_FormURN>[URN FOR ASSOCIATION
    FORM] </Association_FormURN>
20 <Instantiation_FormURN>[URN FOR INSTANTIATION
    FORM] </Instantiation_FormURN>
21 -->
22 <Task0_FormURN>urn:schemas-microsoft-com:office:infopath:
    PersonalTask1:-myXSD-2011-02-16T09-32-29</Task0_FormURN>
23 <Task1_FormURN>urn:schemas-microsoft-com:office:infopath:
    MarketingTask1:-myXSD-2011-02-16T09-32-29</Task1_FormURN>
24 <!-- Änderungsformulare: eine eindeutige GUID für jedes
    Änderungsformular erstellen -->
25 <!--<Modification_[UNIQUE GUID]_FormURN>[URN FOR
    MODIFICATION FORM] </Modification_[UNIQUE GUID]_FormURN>
26 <Modification_[UNIQUE GUID]_Name>[NAME OF MODIFICATION TO
    BE DISPLAYED AS A LINK ON WORKFLOW STATUS
    PAGE</Modification_[UNIQUE GUID]_Name>
27 -->
28 <StatusPageUrl>_layouts/WrkStat.aspx</StatusPageUrl>
29 </MetaData>
30 </Workflow>
31 </Elements>

```

Listing L.12: Elements.xml

## A.18 Definition eines Task Content Types

```

1 <?xml version="1.0" encoding="utf-8" ?>
2 <Elements xmlns="http://schemas.microsoft.com/sharepoint/">
3   <Field ID="ADA16AC2-63D2-4DD4-B30C-072FF5336170"
4     Name="_Abteilung"
5     DisplayName="Abteilung"
6     Description="Abteilung der Personalanforderung"
7     Group="Rowa"
8     Type="Text"
9     Required="false">
10 </Field>
11 <Field ID="863B18E9-E849-411B-9F60-63F8B52696B9"
12   Name="_Stelle"
13   DisplayName="Stelle"
14   Description="Stelle der Personalanforderung"
15   Group="Rowa"
16   Type="Text"

```



```

17         Required="false">
18     </Field>
19     <Field ID="FE9F82DD-032E-4418-B899-B9C5E481FA6F"
20         Name="_Stellenbeschreibung"
21         DisplayName="Stellenbeschreibung"
22         Description="Stellenbeschreibung der
                Personalanforderung"
23         Group="Rowa"
24         Type="Text"
25         Required="false">
26     </Field>
27     <Field ID="FA0E53C9-3FEF-4BF3-9E45-FE0B36DFE017"
28         Name="_Anforderungen"
29         DisplayName="Anforderungen"
30         Description="Anforderungen der Personalanforderung"
31         Group="Rowa"
32         Type="Text"
33         Required="false">
34     </Field>
35 </Elements>

```

Listing L.13: Definition der Felder

```

1 <?xml version="1.0" encoding="utf-8" ?>
2 <Elements xmlns="http://schemas.microsoft.com/sharepoint/">
3     <ContentType
4         ID="0x01080100C98D710D-5495-46D9-8B89-85B46D304C9A"
5         Name="PersonalTaskCT"
6         Group="Rowa"
7         Description="CT für State Machine WF mit ASPX
                Task Form"
8         Version="0"
9         Hidden="False">
10         <FieldRefs>
11             <FieldRef ID="ADA16AC2-63D2-4DD4-B30C-072FF5336170"
12                 Name="_Abteilung"/>
13             <FieldRef ID="863B18E9-E849-411B-9F60-63F8B52696B9"
14                 Name="_Stelle"/>
15             <FieldRef ID="FE9F82DD-032E-4418-B899-B9C5E481FA6F"
16                 Name="_Stellenbeschreibung"/>
17             <FieldRef ID="FA0E53C9-3FEF-4BF3-9E45-FE0B36DFE017"
18                 Name="_Anforderungen"/>
19         </FieldRefs>
20     </ContentType>
21 </Elements>
22 </XmlDocuments>

```

```

20      <XmlDocument NamespaceURI
          =" http://schemas.microsoft.com/sharepoint/v3/contenttype/
            forms/url">
21      <FormUrls
          xmlns=" http://schemas.microsoft.com/sharepoint/v3/contenttype/
            forms/url">
22          <New>_layouts/Personalanforderung/TaskForm.aspx</New>
23          <Display>_layouts/Personalanforderung/TaskForm.aspx</Display>
24          <Edit>_layouts/Personalanforderung/TaskForm.aspx</Edit>
25      </FormUrls>
26      </XmlDocument>
27  </XmlDocuments>
28  </ContentType>
29 </Elements>

```

Listing L.14: Definition des Content Types

## A.19 Definition eines ASPX Task Formulars

```

1 <%@ Assembly Name="$SharePoint.Project.AssemblyFullName$" %>
2 <%@ Import Namespace="Microsoft.SharePoint.ApplicationPages" %>
3 <%@ Register Tagprefix="SharePoint"
    Namespace="Microsoft.SharePoint.WebControls"
    Assembly="Microsoft.SharePoint, Version=14.0.0.0,
    Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>
4 <%@ Register Tagprefix="Utilities"
    Namespace="Microsoft.SharePoint.Utilities"
    Assembly="Microsoft.SharePoint, Version=14.0.0.0,
    Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>
5 <%@ Register Tagprefix="asp" Namespace="System.Web.UI"
    Assembly="System.Web.Extensions, Version=3.5.0.0,
    Culture=neutral, PublicKeyToken=31bf3856ad364e35" %>
6 <%@ Import Namespace="Microsoft.SharePoint" %>
7 <%@ Assembly Name="Microsoft.Web.CommandUI, Version=14.0.0.0,
    Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>
8 <%@ Page Language="C#" AutoEventWireup="true"
    CodeBehind="PersonalTaskForm.aspx.cs"
    Inherits="Rowa.SharePoint2010.Workflow.Personal.TaskForm"
    DynamicMasterPageFile="~masterurl/default.master" %>
9
10 <asp:Content ID="PageHead"
    ContentPlaceHolderID="PlaceHolderAdditionalPageHead"
    runat="server">

```

```
11
12 </asp:Content>
13
14 <asp:Content ID="Main" ContentPlaceHolderID="PlaceHolderMain"
    runat="server">
15 <table>
16     <tr>
17         <td>Abteilung </td>
18         <td><asp:Label ID="lblAbteilung" runat="server" /></td>
19     </tr>
20     <tr>
21         <td>Stelle </td>
22         <td><asp:Label ID="lblStelle" runat="server" /></td>
23     </tr>
24     <tr>
25         <td>Stellenbeschreibung </td>
26         <td><asp:Label ID="lblStellenbeschreibung"
            runat="server" /></td>
27     </tr>
28     <tr>
29         <td>Anforderungen </td>
30         <td><asp:Label ID="lblAnforderungen" runat="server"
            /></td>
31     </tr>
32 </table>
33 <asp:Button ID="btn_advert" runat="server"
    OnClick="btn_advert_Click" Text="Stellenanzeige anfordern" />
34 <asp:Button ID="btn_stop" runat="server"
    OnClick="btn_stop_Click" Text="Workflow beenden" />
35 </asp:Content>
36
37 <asp:Content ID="PageTitle"
    ContentPlaceHolderID="PlaceHolderPageTitle" runat="server">
38 Anwendungsseite
39 </asp:Content>
40
41 <asp:Content ID="PageTitleInTitleArea"
    ContentPlaceHolderID="PlaceHolderPageTitleInTitleArea"
    runat="server" >
42 Meine Anwendungsseite
43 </asp:Content>
```

Listing L.15: Definition des ASPX Task Formulars

```
1 using System;
2 using System.Collections;
3 using System.Web;
4 using Microsoft.SharePoint;
5 using Microsoft.SharePoint.Workflow;
6 using Microsoft.SharePoint.Utilities;
7 using Microsoft.SharePoint.WebControls;
8
9 namespace Rowa.SharePoint2010.Workflow.Personal.TaskForm
10 {
11     public partial class PersonalTaskForm : LayoutsPageBase
12     {
13         /// <summary>
14         /// Referenz auf die Webseite
15         /// </summary>
16         private SPWeb web;
17
18         /// <summary>
19         /// Referenz auf die Taskliste
20         /// </summary>
21         protected SPList taskList;
22
23         /// <summary>
24         /// Referenz auf das Element
25         /// der Taskliste
26         /// </summary>
27         protected SPListItem taskItem;
28
29         /// <summary>
30         /// URL Parameter mit der
31         /// ID des Elements der
32         /// Taskliste
33         /// </summary>
34         string paramTaskListItemID;
35
36         /// <summary>
37         /// URL Parameter mit der
38         /// ID der Taskliste
39         /// </summary>
40         string paramListGuid = string.Empty;
41
42         /// <summary>
43         /// Referenz auf die Instanz des
```

```
44     /// Workflows
45     /// </summary>
46     private SPWorkflow activeWF;
47
48     /// <summary>
49     /// ID der Workflow Instanz
50     /// </summary>
51     private Guid workflowInstanceGuid;
52
53     /// <summary>
54     /// URL Parameter auslesen
55     /// </summary>
56     private void GetWFParameter()
57     {
58         paramListGuid = Request.Params["List"];
59         paramTaskListItemID = Request.Params["ID"];
60     }
61
62     /// <summary>
63     /// Auslesen der Taskliste ,
64     /// des Elements der Taskliste
65     /// und der Workflow Instanz
66     /// </summary>
67     private void GetTaskListInfo()
68     {
69         taskList = web.Lists[new Guid(paramListGuid)];
70         taskItem =
71             taskList.GetItemById(System.Convert.ToInt32(
72                 paramTaskListItemID));
73         workflowInstanceGuid = new
74             Guid(Convert.ToString(taskItem["WorkflowInstanceID"]));
75         activeWF = new SPWorkflow(taskItem ,
76             workflowInstanceGuid);
77     }
78
79     /// <summary>
80     /// Übergabe der Inhalte des
81     /// Task Elements an die Steuerelemente
82     /// des Formulars
83     /// </summary>
84     private void GetTaskInfo()
85     {
86         if (taskItem["Abteilung1"] != null)
```

```
83         {
84             lblAbteilung.Text =
85                 taskItem["Abteilung1"].ToString();
86         }
87
88         if (taskItem["Stelle1"] != null)
89         {
90             lblStelle.Text = taskItem["Stelle1"].ToString();
91         }
92
93         if (taskItem["Stellenbeschreibung1"] != null)
94         {
95             lblStellenbeschreibung.Text =
96                 taskItem["Stellenbeschreibung1"].ToString();
97         }
98
99         if (taskItem["Anforderungen1"] != null)
100        {
101            lblAnforderungen.Text =
102                taskItem["Anforderungen1"].ToString();
103        }
104    }
105
106    /// <summary>
107    /// Aktueller SharePoint Context ermitteln
108    /// </summary>
109    protected override void OnPreInit(EventArgs e)
110    {
111        base.OnPreInit(e);
112        web = SPControl.GetContextWeb(Context);
113    }
114
115    /// <summary>
116    /// Aufruf der privaten Methoden
117    /// </summary>
118    protected void Page_Load(object sender, EventArgs e)
119    {
120        GetWFPParameter();
121        GetTaskListInfo();
122        GetTaskInfo();
123    }
124
125    /// <summary>
```

```

123     /// Event Handler des Buttons
124     /// Stellenanzeige schalten
125     /// </summary>
126     protected void btn_advert_Click(object sender,
        EventArgs e)
127     {
128         /// Hashtabelle zur Rückgabe definieren. Es
129         /// muss immer eine Hashtabelle an den Task
130         /// zurückgegeben werden. Diese ist in den
131         /// ExtendedProperties des Tasks gespeichert
132         Hashtable returnUrl = new Hashtable();
133         returnUrl["status"] = "advert";
134
135         /// Ändern des aktuellen Task Elements
136         SPWorkflowTask.AlterTask(taskItem, returnUrl,
            true);
137
138         /// Redirect auf die ursprüngliche Seite
139         SPUtility.Redirect(taskList.DefaultViewUrl,
            SPRedirectFlags.UseSource, HttpContext.Current);
140     }
141
142     protected void btn_stop_Click(object sender, EventArgs
        e)
143     {
144         /// Redirect auf die ursprüngliche Seite
145         SPUtility.Redirect(taskList.DefaultViewUrl,
            SPRedirectFlags.UseSource, HttpContext.Current);
146     }
147 }
148 }

```

Listing L.16: Quellcodes des ASPX Task Formulars

## A.20 Erstellen eines Tasks mit Content Type

```

1 private void createTask_Personal(object sender, EventArgs e)
2 {
3     /// ID des Task setzen
4     Personal_TaskId = Guid.NewGuid();
5
6     /// Prüfen, ob ContentTypes von der Task Liste
7     /// unterstützt werden

```

```
8      if (workflowProperties.TaskList.ContentTypesEnabled ==
          false)
9      {
10         workflowProperties.TaskList.ContentTypesEnabled = true;
11     }
12
13     // ContentType des Tasks ermitteln. Der Wert stammt aus der
        Aktivität
14     // createTaskWithContentType1
15     SPContentTypeId myCTID = new
        SPContentTypeId(createTaskWithContentType1.ContentTypeId);
16     SPContentType myCT =
        workflowProperties.Site.RootWeb.ContentTypes[myCTID];
17
18     bool ctExists = false;
19
20     // Prüfen, ob der ContentType bereits der Liste angefügt ist
21     foreach (SPContentType ct in
        workflowProperties.TaskList.ContentTypes)
22     {
23         if (ct.Name == myCT.Name)
24         {
25             ctExists = true;
26             break;
27         }
28     }
29
30     // Wenn ContentType noch nicht der Liste zugewiesen wurde,
31     // diesen zuweisen
32     if (ctExists == false)
33     {
34         workflowProperties.TaskList.ContentTypes.Add(myCT);
35         workflowProperties.TaskList.Update();
36     }
37
38     // Übergabe der Werte an den task
39     Personal_TaskProperties.Title = "Neue Personalanforderung";
40     Personal_TaskProperties.AssignedTo = personal;
41     Personal_TaskProperties.ExtendedProperties["Abteilung1"] =
        workflowProperties.Item["Abteilung"];
42     Personal_TaskProperties.ExtendedProperties["Stelle1"] =
        workflowProperties.Item["Stelle"];
```



```

43     Personal_TaskProperties.ExtendedProperties["Stellenbeschreibung1"]
        = workflowProperties.Item["Stellenbeschreibung"];
44     Personal_TaskProperties.ExtendedProperties["Anforderungen1"]
        = workflowProperties.Item["Anforderungen"];
45 }

```

Listing L.17: Erstellen eines Tasks mit Content Type

## A.21 Schnittstellendefinition einer Custom Workflow Activity

```

1 <Elements xmlns="http://schemas.microsoft.com/sharepoint/">
2   <WorkflowActions>
3     <Action
4       Name="Find ID By CAML Query"
5       Category="Query"
6       Assembly="$SharePoint.Project.AssemblyFullName$"
7       ClassName="Rowa.SharePoint2010.Workflow.CustomActivity.FindIdByCAML"
8       FunctionName="Find"
9       AppliesTo="all"
10      SandboxFunction="true">
11      <RuleDesigner
12        Sentence="Find first ID in %1 by query %2">
13          <FieldBind
14            Field="listguid"
15            Text="List"
16            Id="1"
17            DesignerType="ListNames" />
18          <FieldBind
19            Field="query"
20            Text="Query"
21            Id="2"
22            DesignerType="Text" />
23        </RuleDesigner>
24      <Parameters>
25        <Parameter
26          Name="__Context"
27          Type="Microsoft.SharePoint.WorkflowActions.WorkflowContext,
28            Microsoft.SharePoint.WorkflowActions"
29          Direction="In"
30          DesignerType="Hide" />
31        <Parameter
32          Name="listguid"
33          Type="System.Guid, mscorlib"

```

```

34         Direction="In "
35         DesignerType="ParameterNames"
36         Description="List name"
37     />
38     <Parameter
39         Name="query"
40         Type="System.String , mscorlib"
41         Direction="In "
42         DesignerType="ParameterNames"
43         Description="CAML Query"
44     />
45     <Parameter
46         Name = "ID"
47         Type="System.Int32 , mscorlib"
48         Direction="Out"
49         DesignerType="ParameterNames"
50         Description="Result of activity"
51     />
52 </Parameters>
53 </Action>
54 </WorkflowActions>
55 </Elements>

```

Listing L.18: Schnittstelle einer Custom Workflow Activity

## A.22 Quellcode einer Custom Workflow Activity

```

1 using System;
2 using System.Collections;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Text;
6 using Microsoft.SharePoint;
7 using Microsoft.SharePoint.UserCode;
8 using Microsoft.SharePoint.Workflow;
9
10 namespace Rowa.SharePoint2010.Workflow.CustomActivity
11 {
12     /// <summary>
13     /// Klassendefinition der CustomActivity
14     /// </summary>
15     public class FindIdByCAML
16     {

```

```
17      /// <summary>
18      /// Methode sucht in der angegebenen Liste mit dem
19      /// angegebenen Query und gibt das erste gefundene
20      /// Element zurück
21      /// </summary>
22      /// <param name="context">Kontext der SharePoint
        Sitzung </param>
23      /// <param name="listguid">ID der zu durchsuchenden
        Liste </param>
24      /// <param name="query">CAML Query</param>
25      /// <returns>Tabelle mit Ergebnis</returns>
26      public Hashtable Find(SPUserCodeWorkflowContext
        context, Guid listguid, string query)
27      {
28          // Definition der Hashtabelle, die zurückgegeben
        wird.
29          // Eine Custom Workflow Activity muss immer
30          // eine Hashtabelle zurück geben.
31          Hashtable returnTable = new Hashtable();
32          try
33          {
34              using (SPSite site = new
                SPSite(context.SiteUrl))
35              {
36                  using (SPWeb web =
                    site.OpenWeb(context.WebUrl))
37                  {
38                      // Auslesen der Liste, vorbereiten des
                        Query
39                      SPList list = web.Lists[listguid];
40                      SPQuery listQuery = new SPQuery();
41                      listQuery.Query = query;
42
43                      // Es soll nur das Feld ID gelesen
                        werden
44                      listQuery.ViewFields = "<FieldRef
                        Name='ID' />";
45
46                      // Suche nach Listenenelementen mit CAML
                        Query
47                      SPListItemCollection itemCollection =
                        list.GetItems(listQuery);
48
```

```
49         // Wurden Elemente gefunden, das erste
50         // zurück geben
51         if (itemCollection.Count > 0)
52         {
53             returnTable["ID"] =
54                 Convert.ToInt32(itemCollection[0]["ID"].
55                     ToString());
56         }
57         else
58         {
59             returnTable["ID"] = -1;
60         }
61     }
62     catch
63     {
64         returnTable["ID"] = -1;
65     }
66     return returnTable;
67 }
68 }
69 }
```

Listing L.19: Quellcode einer Custom Workflow Activity

## A.23 WebPart zum Anzeigen von Daten per REST Schnittstelle

```

1 namespace Rowa.SharePoint2010.WebParts.Termine
2 {
3     using System;
4     using System.Linq;
5     using System.Net;
6     using System.Web.UI;
7     using System.Web.UI.WebControls;
8     using Microsoft.SharePoint;
9     using Rowa.SharePoint2010.WebParts.RESTTermin;
10
11     /// <summary>
12     /// Code Behind des UserControl. Zeigt alle Messen an
13     /// </summary>
14     public partial class TermineUserControl : UserControl
15     {
16         /// <summary>
17         /// DataContext der REST Abfrage.
18         /// </summary>
19         private MesseplanungDataContext ctx;
20
21         /// <summary>
22         /// Klasse instantiiert die REST Schnittstelle
23         /// </summary>
24         /// <param name="e"></param>
25         protected override void OnInit(EventArgs e)
26         {
27             SPSecurity.RunWithElevatedPrivileges(delegate
28             {
29                 this.ctx = new MesseplanungDataContext(new
30                     Uri("http://sp2010rowaws179/exhibitions/_vti_bin/
31                     Listdata.svc"));
32                 this.ctx.Credentials =
33                     CredentialCache.DefaultCredentials;
34             });
35             base.OnInit(e);
36         }
37
38         /// <summary>
39         /// Methode liest den Inhalt der Messeliste aus
40         /// </summary>

```

```

38     /// <param name="e"></param>
39     protected override void OnLoad(EventArgs e)
40     {
41         if (!IsPostBack)
42         {
43             try
44             {
45                 SPSecurity.RunWithElevatedPrivileges(delegate
46                 {
47                     // Linq Ausdruck zum Auslesen der
48                     // Messetermine.
49                     // Es werden nur die Termine angezeigt,
50                     // die von
51                     // Partnern gesehen werden darf. Es
52                     // werden nur
53                     // bestimmte Felder der Liste gelesen
54                     var data = from t in this.ctx.Termine
55                                 where t.Partner == true
56                                 orderby t.Von
57                                 select new { t.Messe,
58                                             t.Land, t.Ort, t.Von,
59                                             t.Bis, t.Stand,
60                                             t.Messelink,
61                                             t.Beschreibung };
62
63                     // Grid mit Daten versorgen
64                     GridTermine.DataSource = data;
65                     GridTermine.DataBind();
66                 });
67             }
68             catch
69             {
70             }
71
72             base.OnLoad(e);
73         }
74     }
75
76     /// <summary>
77     /// Eventhandler des Buttons im Grid.
78     /// Methode erstellt Eintrag in der Terminanfrage Liste
79     /// </summary>
80     /// <param name="sender"></param>

```

```

74      /// <param name="e"></param>
75      protected void GridTerminе_RowCommand(object sender,
76          GridViewCommandEventArgs e)
77      {
78          try
79          {
80              // Index des Listenelementes und aktueller
81              // Benutzer auslesen
82              int index = Convert.ToInt32(e.CommandArgument);
83              int partnerid =
84                  SPContext.Current.Web.CurrentUser.ID;
85
86              // Neuer Listeneintrag erstellen
87              SPSecurity.RunWithElevatedPrivileges(delegate
88              {
89                  RESTTerminе.TerminanfragenItem item = new
90                      TerminanfragenItem();
91                  item.Messe =
92                      GridTerminе.DataKeys[index].Value.ToString();
93
94                  item.PartnerId = partnerid;
95                  this.ctx.AddToTerminanfragen(item);
96                  this.ctx.SaveChanges();
97              });
98          }
99      }

```

Listing L.20: Programmcode des UserControl

```

1 <?@ Assembly Name="$SharePoint.Project.AssemblyFullName$" %>
2 <?@ Assembly Name="Microsoft.Web.CommandUI, Version=14.0.0.0,
   Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>
3 <?@ Register Tagprefix="SharePoint"
   Namespace="Microsoft.SharePoint.WebControls"
   Assembly="Microsoft.SharePoint, Version=14.0.0.0,
   Culture=neutral, PublicKeyToken=71e9bce111e9429c" %>
4 <?@ Register Tagprefix="Utilities"
   Namespace="Microsoft.SharePoint.Utilities"
   Assembly="Microsoft.SharePoint, Version=14.0.0.0,

```

```

    Culture=neutral , PublicKeyToken=71e9bce111e9429c" %>
5 <%@ Register Tagprefix="asp" Namespace="System.Web.UI"
    Assembly="System.Web.Extensions , Version=3.5.0.0 ,
    Culture=neutral , PublicKeyToken=31bf3856ad364e35" %>
6 <%@ Import Namespace="Microsoft.SharePoint" %>
7 <%@ Register Tagprefix="WebPartPages"
    Namespace="Microsoft.SharePoint.WebPartPages"
    Assembly="Microsoft.SharePoint , Version=14.0.0.0 ,
    Culture=neutral , PublicKeyToken=71e9bce111e9429c" %>
8 <%@ Control Language="C#" AutoEventWireup="true"
    CodeBehind="TermineUserController.ascx.cs"
    Inherits="Rowa.SharePoint2010.WebParts.Termin.TerminUserController"
    %>
9 <asp:GridView ID="GridTermin" runat="server"
    AutoGenerateColumns="false" DataKeyNames="Messe"
    OnRowCommand="GridTermin_RowCommand">
10 <Columns>
11 <asp:BoundField DataField="Messe" HeaderText="Exhibition"
    />
12 <asp:BoundField DataField="Land" HeaderText="Country" />
13 <asp:BoundField DataField="Ort" HeaderText="City" />
14 <asp:BoundField DataField="Von" HeaderText="From"
    DataFormatString="{0:d}" />
15 <asp:BoundField DataField="Bis" HeaderText="To"
    DataFormatString="{0:d}" />
16 <asp:BoundField DataField="Stand" HeaderText="Location" />
17 <asp:HyperLinkField DataTextField="Messelink"
    DataNavigateUrlFields="Messelink" HeaderText="Link" />
18 <asp:BoundField DataField="Beschreibung"
    HeaderText="Description" />
19 <asp:TemplateField>
20 <ItemTemplate>
21 <asp:Button ID="more" runat="server"
    CommandName="more" CommandArgument="<%#
    ((GridViewRow) Container).RowIndex %>"
    Text="subscribe"
    OnClientClick="javascript:return confirm('Thank
    you for your interest. We will send you further
    information ');" />
22 </ItemTemplate>
23 </asp:TemplateField>
24
25 </Columns>

```



---

```
26 </asp:GridView>
```

### Listing L.21: Definition des Usercontrol

## A.24 Anlegen von Websites per WCF Dienst

```

1 namespace Rowa.SharePoint2010.WCF.SiteCreator
2 {
3     using System;
4     using System.Collections.Generic;
5     using System.Linq;
6     using System.ServiceModel;
7     using System.Text;
8
9     /// <summary>
10    /// Schnittstellendefinition des Dienstes
11    /// </summary>
12    [ServiceContract]
13    public interface ISiteCreator
14    {
15        /// <summary>
16        /// Methode erstellt neue Websites
17        /// </summary>
18        /// <param name="url">URL der neuen Seite </param>
19        /// <param name="title">Titel der neuen Seite </param>
20        /// <param name="description">Beschreibung der neuen
21        /// Site </param>
22        /// <param name="lcid">Sprachschlüssel der Seite </param>
23        /// <param name="template">zu verwendendes Site
24        /// Template </param>
25        /// <param name="pageLib">Name der Bibliothek mit
26        /// Seiten </param>
27        /// <param name="pageTitle">Titel der Startseite </param>
28        /// <param name="useUniquePermission">gibt an, ob Site
29        /// eigene Berechtigungen haben soll </param>
30        /// <param name="permissions">Liste der
31        /// Berechtigungen </param>
32        /// <returns>Rückgabeobjekt mit Status und
33        /// Nachricht </returns>
34        [OperationContract]
35        SiteCreatorResponse CreateNewWebSite(Uri url, string
36            title, string description, uint lcid, string
37            template, string pageLib, string pageTitle,

```

```

30         bool useUniquePermission ,
           List<SiteCreatorPermissions> permissions);
31     }
32 }

```

### Listing L.22: Schnittstelle des Dienstes

```

1 namespace Rowa.SharePoint2010.WCF.SiteCreator
2 {
3     using System;
4     using System.Collections.Generic;
5     using System.Linq;
6     using System.Runtime.Serialization;
7     using System.ServiceModel;
8     using System.ServiceModel.Activation;
9     using System.Text;
10    using Microsoft.SharePoint;
11    using Microsoft.SharePoint.Client.Services;
12
13    /// <summary>
14    /// Klasse implementiert die Funktion des
15    /// Dienstes
16    /// </summary>
17    [BasicHttpBindingServiceMetadataExchangeEndpoint]
18    [AspNetCompatibilityRequirements(RequirementsMode =
        AspNetCompatibilityRequirementsMode.Required)]
19    public class SiteCreator : ISiteCreator
20    {
21        /// <summary>
22        /// Methode erstellt neue Websites
23        /// </summary>
24        /// <param name="url">URL der neuen Seite </param>
25        /// <param name="title">Titel der neuen Seite </param>
26        /// <param name="description">Beschreibung der neuen
        /// Site </param>
27        /// <param name="lcid">Sprachschlüssel der Seite </param>
28        /// <param name="template">zu verwendendes Site
        /// Template </param>
29        /// <param name="pageLib">Name der Bibliothek mit
        /// Seiten </param>
30        /// <param name="pageTitle">Titel der Startseite </param>
31        /// <param name="useUniquePermission">gibt an, ob Site
        /// eigene Berechtigungen haben soll </param>

```

```
32      /// <param name="permissions">Liste der
      Berechtigungen </param>
33      /// <returns>Rückgabeobjekt mit Status und
      Nachricht </returns>
34      public SiteCreatorResponse CreateNewWebSite(Uri url ,
      string title , string description , uint lcid , string
      template , string pageLib , string pageTitle ,
35          bool useUniquePermission ,
      List<SiteCreatorPermissions> permissions)
36      {
37          /// Definition des Rückgabeobjektes
      SiteCreatorResponse response = new
38          SiteCreatorResponse() ;
39          try
40          {
41              using (SPSite site = new SPSite(url.Scheme +
      "://" + url.Host))
42              {
43                  /// Aufteilen der URL in einzelne Seiten
      string [] subWebUrl =
44                  url.AbsolutePath.Split ( '/' );
45                  string existingPath = string.Empty;
46
47                  foreach (string subweb in subWebUrl)
48                  {
49                      if (subweb == string.Empty)
50                      {
51                          continue ;
52                      }
53
54                      /// Seite existiert nicht , neu anlegen
      if (!this.WebExists(subweb, site))
55                      {
56                          using (SPWeb parentWeb =
      site.OpenWeb(existingPath))
57                          {
58                              parentWeb.AllowUnsafeUpdates =
      true ;
59
60                              /// Seite erstellen
      SPWeb newWeb =
61                              parentWeb.Webs.Add(subweb ,
      title , description , lcid ,
62
```

```
        template ,
        useUniquePermission , false );

63
64 // Berechtigungen vergeben
65 if (useUniquePermission == true)
66 {
67     bool permissionSuccess =
        this.SetPermissions(newWeb,
        permissions);
68     if (permissionSuccess ==
        false)
69     {
70         response.success = true;
71         response.warning = true;
72         response.message += "
            Can't create
            permissions for
            SPWeb";
73     }
74 }
75
76 // Titel der Startseite setzen
77 if
    (this.SetTitleDefaultPage(newWeb,
    pageLib, pageTitle) == false)
78 {
79     response.success = true;
80     response.warning = true;
81     response.message += "
        Unable to change page
        title";
82 }
83
84 parentWeb.Update();
85 parentWeb.AllowUnsafeUpdates =
    false;
86 existingPath = existingPath +
    "/" + subweb;
87 }
88 }
89 else
90 {
```

```

91             existingPath = existingPath + "/" +
92                 subweb;
93         }
94     }
95     response.success = true;
96 }
97 }
98 catch (Exception ex)
99 {
100     response.success = false;
101     response.warning = false;
102     response.message += " " + ex.Message;
103 }
104
105 return response;
106 }
107
108 /// <summary>
109 /// Methode prüft, ob übergebene Website existiert
110 /// </summary>
111 /// <param name="web">Name der Website</param>
112 /// <param name="site">übergeordnete Site
113     Collection</param>
114 /// <returns>gibt an, ob die Website existiert</returns>
115 private bool WebExists(string web, SPSPSite site)
116 {
117     bool exists = false;
118
119     SPSecurity.RunWithElevatedPrivileges(delegate
120     {
121         try
122         {
123             // Website öffnen und Existenz prüfen
124             using (SPWeb subweb = site.OpenWeb(web))
125             {
126                 exists = subweb.Exists;
127             }
128         }
129         catch
130         {
131         }
132     });

```

```
132
133         return exists;
134     }
135
136     /// <summary>
137     /// Methode setzt Berechtigungen auf die Website
138     /// </summary>
139     /// <param name="web">zu berechtigende Website</param>
140     /// <param name="permissions">Liste der
141         Berechtigungen</param>
142     /// <returns>gibt an, ob Berechtigungen gesetzt
143         wurden</returns>
144     private bool SetPermissions(SPWeb web,
145         List<SiteCreatorPermissions> permissions)
146     {
147         bool complete = true;
148
149         foreach (SiteCreatorPermissions permission in
150             permissions)
151         {
152             try
153             {
154                 // Benutzerobjekt erstellen
155                 SPUser user =
156                     web.EnsureUser(permission.user);
157                 web.AllowUnsafeUpdates = true;
158
159                 // Rollenzuordnung erstellen
160                 SPRoleAssignment roleAssign = new
161                     SPRoleAssignment((SPPrincipal)user);
162
163                 // Rolle Definieren
164                 SPRoleDefinition roleDef =
165                     web.RoleDefinitions[permission.roleName];
166                 roleAssign.RoleDefinitionBindings.Add(roleDef);
167
168                 // Rollenzuordnung der Website zuweisen
169                 web.RoleAssignments.Add(roleAssign);
170                 web.Update();
171                 web.AllowUnsafeUpdates = false;
172             }
173             catch
174             {

```

```
168         complete = false;
169     }
170 }
171
172     return complete;
173 }
174
175     /// <summary>
176     /// Setzen des Titels der Startseite
177     /// </summary>
178     /// <param name="web">übergeordnete Website</param>
179     /// <param name="pageLib">Bibliothek, welche die Seiten
180     /// enthält</param>
181     /// <param name="pageTitle">Titel der Startseite</param>
182     /// <returns>Gibt an, ob Titel gesetzt wurde</returns>
183     private bool SetTitleDefaultPage(SPWeb web, string
184     pageLib, string pageTitle)
185     {
186         bool complete = false;
187         try
188         {
189             // Startseite selektieren
190             SPListItem defaultpage =
191                 web.Lists[pageLib].Items[0];
192
193             // Auschecken der Seite
194             defaultpage.File.CheckOut();
195             defaultpage["Title"] = pageTitle;
196
197             // Einchecken und veröffentlichen
198             defaultpage.File.CheckIn(string.Empty);
199             defaultpage.File.Publish(string.Empty);
200
201             defaultpage.ParentList.Update();
202             web.Update();
203             complete = true;
204         }
205         catch
206         {
207         }
```

```
208 }
209 }
```

Listing L.23: Programmcode des Dienstes

```
1 namespace Rowa.SharePoint2010.WCF.SiteCreator
2 {
3     using System;
4     using System.Collections.Generic;
5     using System.Linq;
6     using System.Text;
7
8     /// <summary>
9     /// Klasse zur Repräsentation von Benutzerrechten
10    /// </summary>
11    public class SiteCreatorPermissions
12    {
13        /// <summary>
14        /// Name der Rolle
15        /// </summary>
16        public string roleName;
17
18        /// <summary>
19        /// Benutzer der Rolle
20        /// </summary>
21        public string user;
22    }
23 }
```

Listing L.24: Berechtigungsobjekt des Dienstes

```
1 namespace Rowa.SharePoint2010.WCF.SiteCreator
2 {
3     using System;
4     using System.Collections.Generic;
5     using System.Linq;
6     using System.Text;
7
8     /// <summary>
9     /// Rückgabeobjekt des Service
10    /// </summary>
11    public class SiteCreatorResponse
12    {
13        /// <summary>
14        /// gibt an, ob die Website erstellt wurde
```



```

15     /// </summary>
16     public bool success = false;
17
18     /// <summary>
19     /// gibt an, ob bei den weiteren Operationen
20     /// Fehler aufgetreten sind
21     /// </summary>
22     public bool warning = false;
23
24     /// <summary>
25     /// enthält Beschreibungen von aufgetretenen
26     /// Fehlern
27     /// </summary>
28     public string message = string.Empty;
29 }
30 }

```

Listing L.25: Rückgabeobjekt des Dienstes

```

1 <%@ServiceHost Language="C#" Debug="true "
2     Service="Rowa.SharePoint2010.WCF.SiteCreator.SiteCreator ,
    $SharePoint.Project.AssemblyFullName$"
3     Factory="Microsoft.SharePoint.Client.Services.
    MultipleBaseAddressBasicHttpBindingServiceHostFactory ,
    Microsoft.SharePoint.Client.ServerRuntime ,
    Version=14.0.0.0, Culture=neutral ,
    PublicKeyToken=71e9bce111e9429c" %>

```

Listing L.26: Definition des Service Hosts

## A.25 Jobüberwachung per WCF Dienst

```

1 namespace Rowa.SharePoint2010.WCF.JobCheck
2 {
3     using System;
4     using System.ServiceModel;
5
6     /// <summary>
7     /// Schnittstellendefinition des Dienstes
8     /// </summary>
9     [ServiceContract]
10    public interface IJobCheck
11    {
12        /// <summary>

```

```

13      /// Methode prüft, wie oft ein Job in einer Zeitspanne
        abgebrochen ist
14      /// </summary>
15      /// <param name="definition">Name des zu prüfenden
        Jobs</param>
16      /// <param name="start">Startzeit </param>
17      /// <param name="end">Endzeit </param>
18      /// <returns>Anzahl der Abbrüche oder -1 bei
        Fehler</returns>
19      [OperationContract]
20      int Check(string definition, DateTime start, DateTime
        end);
21  }
22 }

```

Listing L.27: Schnittstelle des Dienstes

```

1 namespace Rowa.SharePoint2010.WCF.JobCheck
2 {
3     using System;
4     using System.Linq;
5     using System.ServiceModel.Activation;
6     using Microsoft.SharePoint.Administration;
7     using Microsoft.SharePoint.Client.Services;
8
9     /// <summary>
10    /// Klasse implementiert die Funktion des Dienstes
11    /// </summary>
12    [BasicHttpBindingServiceMetadataExchangeEndpoint]
13    [AspNetCompatibilityRequirements(RequirementsMode =
        AspNetCompatibilityRequirementsMode.Required)]
14    public class JobCheck : IJobCheck
15    {
16        /// <summary>
17        /// Methode prüft, wie oft ein Job in einer Zeitspanne
            abgebrochen ist
18        /// </summary>
19        /// <param name="definition">Name des zu prüfenden
            Jobs</param>
20        /// <param name="start">Startzeit </param>
21        /// <param name="end">Endzeit </param>
22        /// <returns>Anzahl der Abbrüche oder -1 bei
            Fehler</returns>

```

```
23     public int Check(string definition, DateTime start,
24         DateTime end)
25     {
26         int count = 0;
27         bool jobfound = false;
28         try
29         {
30             foreach (SPService service in
31                 SPFarm.Local.Services)
32             {
33                 var jobs = service.JobDefinitions.Where(x
34                     => x.DisplayName == definition);
35                 if (jobs.Count() > 0)
36                 {
37                     jobfound = true;
38                 }
39
40                 foreach (SPJobDefinition job in jobs)
41                 {
42                     var history =
43                         job.HistoryEntries.Where(x =>
44                             x.StartTime > start && x.StartTime <
45                             end);
46                     foreach (SPJobHistory hist in history)
47                     {
48                         if (hist.Status ==
49                             SPRunningJobStatus.Aborted ||
50                             hist.Status ==
51                             SPRunningJobStatus.Failed)
52                         {
53                             count++;
54                         }
55                     }
56                 }
57             }
58
59             if (jobfound == false && count == 0)
60             {
61                 count = -1;
62             }
63         }
64         catch
65         {
66         }
```

```

57         count = -1;
58     }
59
60     return count;
61 }
62 }
63 }

```

Listing L.28: Programmcode des Dienstes

## A.26 WCF Dienst zum Hochladen und Berechtigen von Dokumenten per Client Objekt Modell

```

1 namespace Rowa.SharePoint2010.WCF.Client
2 {
3     using System;
4     using System.Collections.Generic;
5     using System.ServiceModel;
6
7     /// <summary>
8     /// Schnittstellenbeschreibung des Dienstes
9     /// </summary>
10    [ServiceContract]
11    public interface IUpload
12    {
13        /// <summary>
14        /// Methode zum hochladen von Dokumenten in eine
15        /// Bibliothek mit anschließender Berechtigungsvergabe
16        /// </summary>
17        /// <param name="targetURL">URL der Website</param>
18        /// <param name="targetLib">Name der Bibliothek</param>
19        /// <param name="fileName">Name der Datei</param>
20        /// <param name="bytes">Bytestream der Datei</param>
21        /// <param name="permissions">Liste mit
22        /// Berechtigungen</param>
23        /// <returns>True, wenn Dokument hochgeladen und
24        /// Berechtigungen gesetzt wurden</returns>
25        [OperationContract]
26        bool UploadDocument(Uri targetURL, string targetLib,
27            string fileName, byte[] bytes, List<Permissions>
28            permissions);
29    }
30 }

```

## Listing L.29: Schnittstelle des Dienstes

```
1 namespace Rowa.SharePoint2010.WCF.Client
2 {
3     using System;
4     using System.Collections.Generic;
5     using Microsoft.SharePoint.Client;
6
7     /// <summary>
8     /// Klasse implementiert die Funktion des Dienstes
9     /// </summary>
10    public class Upload : IUpload
11    {
12        /// <summary>
13        /// Methode zum hochladen von Dokumenten in eine
14        /// Bibliothek mit anschließender Berechtigungsvergabe
15        /// </summary>
16        /// <param name="targetURL">URL der Website</param>
17        /// <param name="targetLib">Name der Bibliothek</param>
18        /// <param name="fileName">Name der Datei</param>
19        /// <param name="bytes">Bytestream der Datei</param>
20        /// <param name="permissions">Liste mit
21        /// Berechtigungen</param>
22        /// <returns>True, wenn Dokument hochgeladen und
23        /// Berechtigungen gesetzt wurden</returns>
24        public bool UploadDocument(Uri targetURL, string
25            targetLib, string fileName, byte[] bytes,
26            List<Permissions> permissions)
27        {
28            bool returnvalue = true;
29            try
30            {
31                // Client Context Objekt erstellen
32                ClientContext spContext = new
33                    ClientContext(targetURL);
34                Web web = spContext.Web;
35
36                // Dateiinformation erstellen
37                FileCreationInformation newFile = new
38                    FileCreationInformation();
39                newFile.Content = bytes;
40                newFile.Url = fileName;
41                newFile.Overwrite = true;
```

```
35
36 // Zielbibliothek auswählen
37 List doclib = web.Lists.GetByTitle(targetLib);
38
39 // Dokument hochladen
40 File uploadFile =
    doclib.RootFolder.Files.Add(newFile);
41
42 // Durchführen der vorherigen Operationen
43 spContext.Load(uploadFile);
44 spContext.ExecuteQuery();
45
46 // erstelltes Listenelement lesen
47 ListItem item = uploadFile.ListItemAllFields;
48 item.BreakRoleInheritance(false, false);
49
50 // Berechtigungen setzen
51 foreach (Permissions perm in permissions)
52 {
53     User user =
54         spContext.Web.EnsureUser(perm.user);
55     RoleDefinitionBindingCollection
56         roleBindingColl = new
57             RoleDefinitionBindingCollection(spContext);
58     roleBindingColl.Add(spContext.Web.RoleDefinitions.
59         GetByName(perm.permission));
60
61     item.RoleAssignments.Add(user,
62         roleBindingColl);
63 }
64
65 // Durchführen der vorherigen Operationen
66 spContext.ExecuteQuery();
67
68 }
69 catch
70 {
71     returnvalue = false;
72 }
73
74 return returnvalue;
75 }
```

71 }

## Listing L.30: Programmcode des Dienstes

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Runtime.Serialization;
6
7 namespace Rowa.SharePoint2010.WCF.Client
8 {
9     [DataContract]
10    public class Permissions
11    {
12        [DataMember]
13        public string user;
14
15        [DataMember]
16        public string permission;
17    }
18 }
```

## Listing L.31: Berechtigungsobjekt des Dienstes

```
1 <?xml version="1.0"?>
2 <configuration>
3     <appSettings/>
4     <connectionStrings/>
5     <system.web>
6         <compilation debug="true" targetFramework="4.0">
7             </compilation>
8             <authentication mode="Windows"/>
9
10            <pages
11                controlRenderingCompatibilityVersion="3.5"
12                clientIDMode="AutoID"/> </system.web>
13
14            <system.serviceModel>
15                <serviceHostingEnvironment
16                    multipleSiteBindingsEnabled="true" />
17
18            <services>
19                <service
20                    name="Rowa.SharePoint2010.WCF.Client.WCFUpload2SharePointClient.Upload">
21                    <endpoint address=""
```

```
17         binding="basicHttpBinding"
18         contract="Rowa.SharePoint2010.WCF.Client.
           WCFUpload2SharePointClient.IUpload" />
19
20     <endpoint address="mex"
21         binding="mexHttpBinding"
22         contract="IMetadataExchange" />
23 </service>
24
25     </services>
26     <behaviors>
27 <serviceBehaviors>
28     <behavior name="">
29         <serviceMetadata httpGetEnabled="true" />
30         <serviceDebug includeExceptionDetailInFaults="false" />
31     </behavior>
32 </serviceBehaviors>
33 </behaviors>
34     </system.serviceModel>
35 </configuration>
```

Listing L.32: web.config des Dienstes



## A.27 Festlegen eines Navigationsformulars einer Web Datenbank

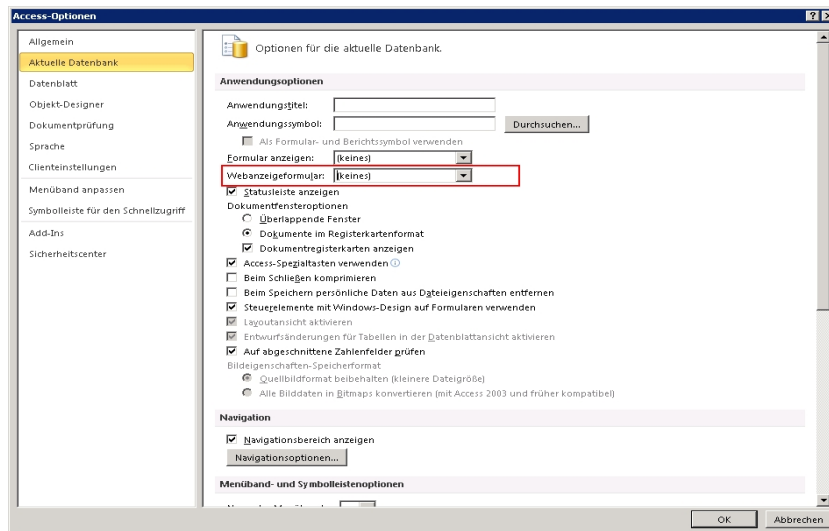


Abbildung A.16: Festlegen des Navigationsformulars (eigene Darstellung)

## A.28 Elemente einer Web Datenbank

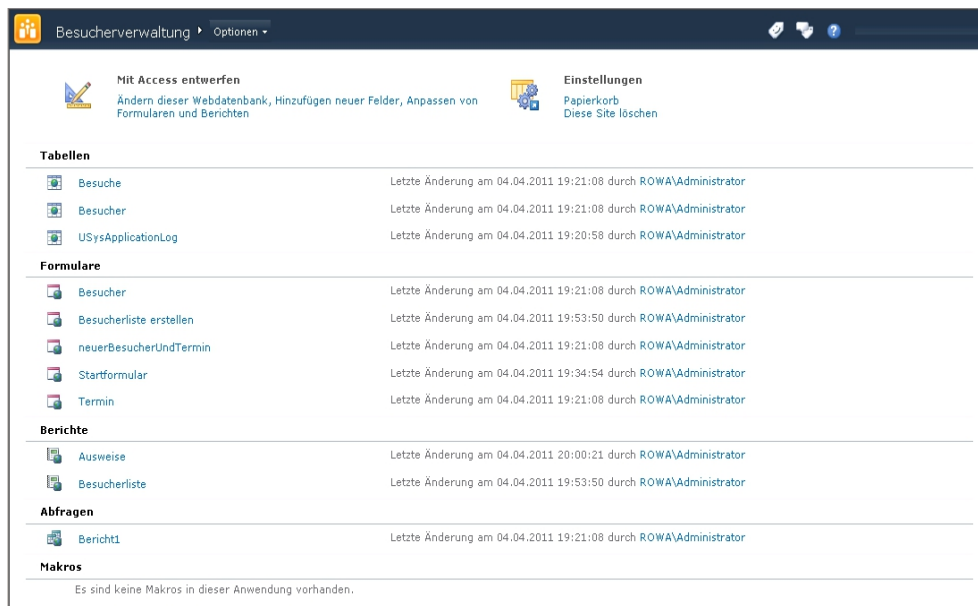


Abbildung A.17: Elemente einer Web Datenbank (eigene Darstellung)

## A.29 Website Inhalt einer Web Datenbank

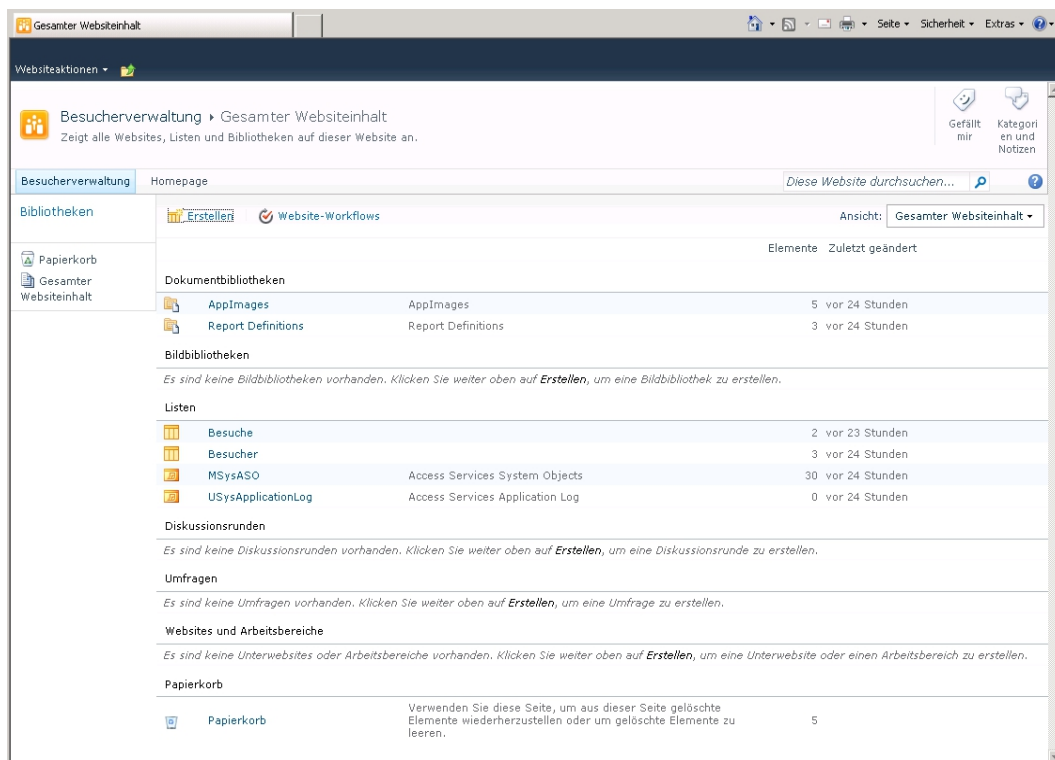


Abbildung A.18: Website Inhalt einer Web Datenbank (eigene Darstellung)

## A.30 Access Services Fehlermeldung bei unzureichenden Berechtigungen

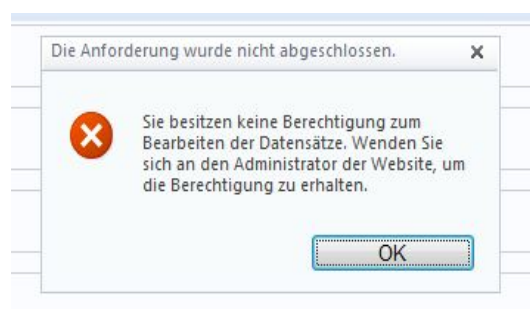


Abbildung A.19: Meldung bei unzureichenden Berechtigungen (eigene Darstellung)

## A.31 Startseite der Schulungsdatenbank

The screenshot shows a web browser window titled 'Homepage - Schulung'. The main content area is titled 'Navigationsformular' and contains a tabbed interface with tabs for 'Lehrer', 'Schulungen', 'Teilnehmer', 'Veranstaltungen', 'Veranstaltungsteilnehmer', and 'Teilnehmerübersicht'. The 'Lehrer' tab is active, displaying a form for entering teacher information. The form fields are as follows:

Vorname	Klaus
Nachname	Meier
Straße	Hauptstraße 18
PLZ	89897
Ort	München
Firma	Personalberatung Meier & Co.

At the bottom of the form, there is a status bar indicating 'Datensatz 1 von 3'.

Abbildung A.20: Navigationsformular der Schulungsdatenbank (eigene Darstellung)

## A.32 InfoPath Designer 2010 mit Listenformular

The screenshot shows the Microsoft InfoPath Designer 2010 interface. The main window displays a list form titled 'Kundenfeedback'. The form contains several sections with text prompts and input fields:

- Mitarbeiter:** A text field for the employee name.
- Datum des Kontaktes:** A date picker field.
- Grund des Kontaktes:** A dropdown menu.
- Freundlichkeit:** A rating scale from 1 to 6.
- Kompetenz:** A rating scale from 1 to 6.
- Beratung:** A rating scale from 1 to 6.
- Lösungsqualität:** A rating scale from 1 to 6.
- Kommentar:** A text area for additional comments.

On the right side, there is a 'Felder' (Fields) pane showing a list of fields available for the form, including 'ID', 'Title', 'Created By', 'Modified By', 'Created', 'Attachments', 'Kunde', 'Termin', 'Terminart', 'Freundlichkeit', 'Kompetenz', 'Beratung', 'Lösung', and 'Kommentar'. Below the fields list, there is an 'Aktionen' (Actions) pane with options like 'Feld hinzufügen' and 'Datenverbindungen verwalten...'.

Abbildung A.21: InfoPath Designer 2010 mit Listenformular (eigene Darstellung)

## A.33 Vertrauenswürdige Speicherorte der Excel Services

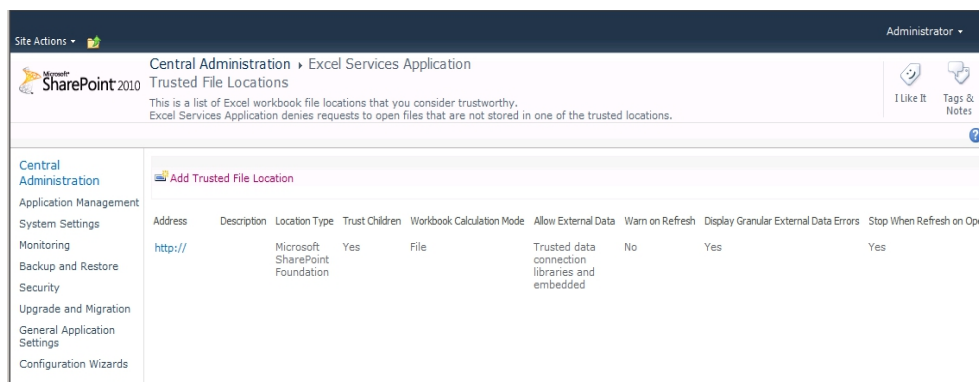


Abbildung A.22: vertrauenswürdige Speicherorte (eigene Darstellung)

## A.34 Unattended Service Account der Excel Services

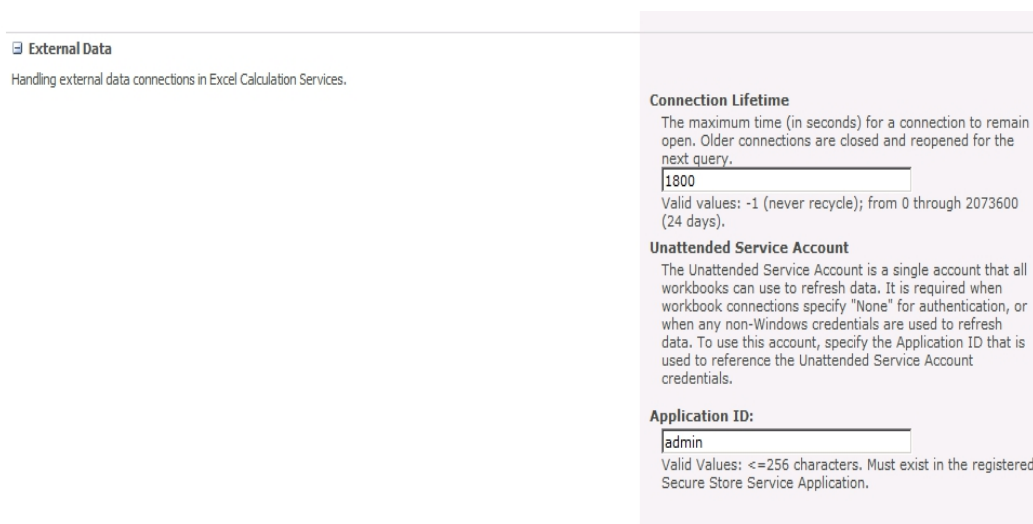


Abbildung A.23: Unattended Service Account (eigene Darstellung)

## A.35 Identität einer Web Application

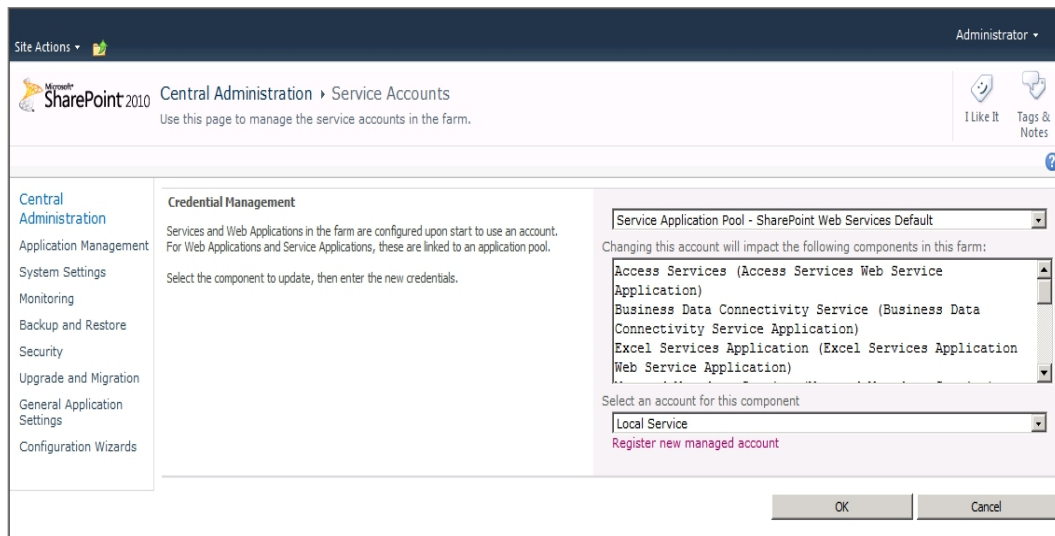


Abbildung A.24: Identität der Web Application (eigene Darstellung)

## A.36 Excel Services Parameter und benannte Bereiche

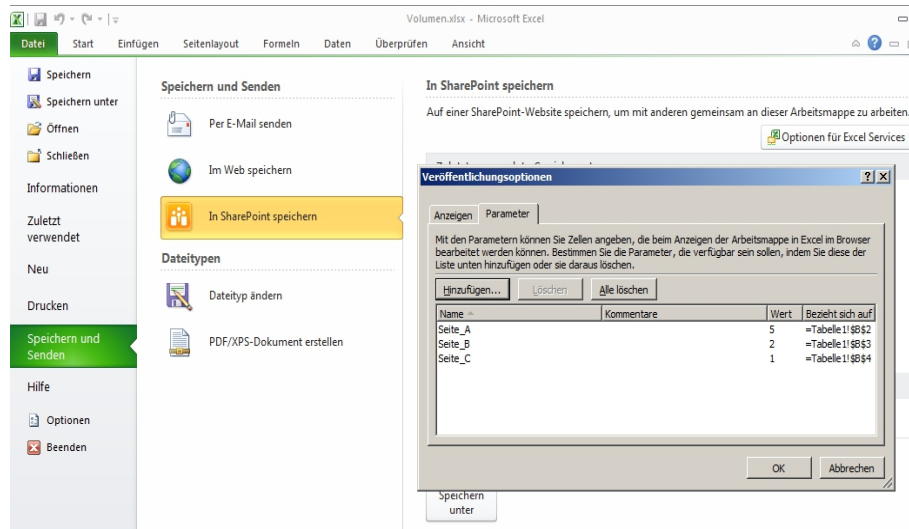


Abbildung A.25: Definition der verfügbaren Elemente einer Excel Arbeitsmappe (eigene Darstellung)

## A.37 Reporting Services Bericht

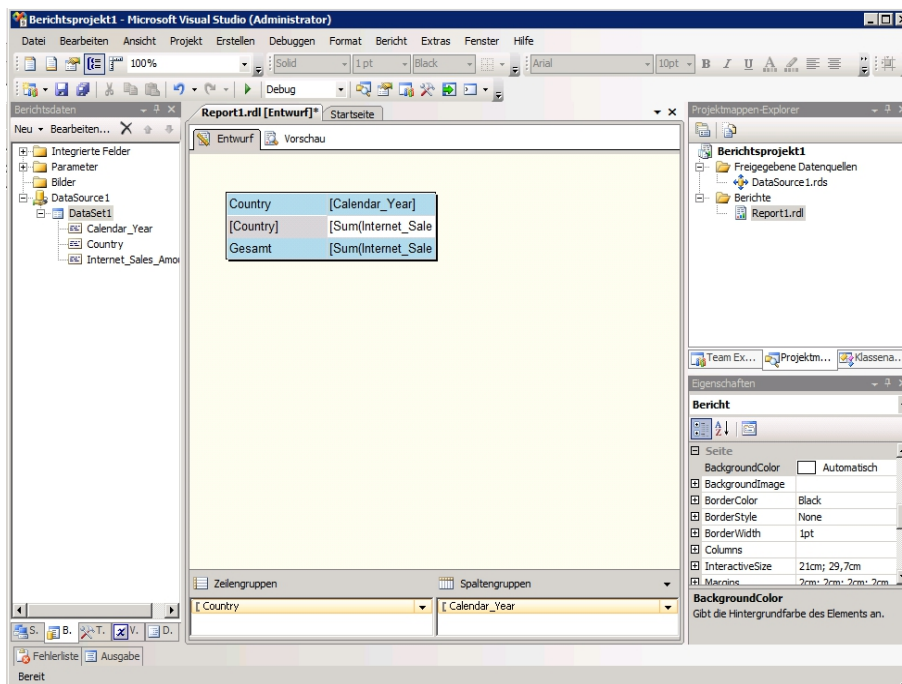


Abbildung A.26: Definition eines Reporting Services Berichts (eigene Darstellung)

## A.38 Bereitstellungsziel eines Reporting Services Berichts

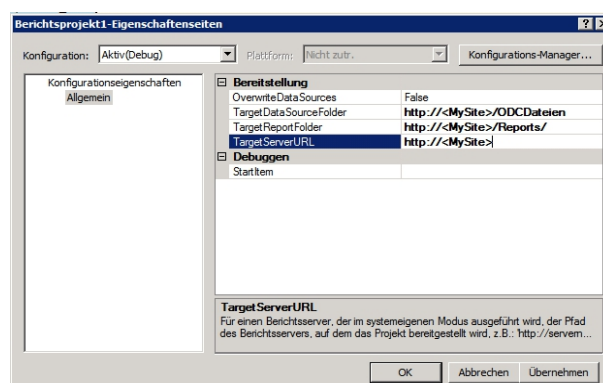


Abbildung A.27: Bereitstellungsziel eines Reporting Services Berichts (eigene Darstellung)

## **B Eidesstattliche Erklärung**

Patrick Weber

Matr. Nr. 877310

Hiermit erkläre ich, dass ich diese Arbeit selbstständig abgefasst und keine anderen als die angegebenen Hilfsmittel benutzt habe.

Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Gerolstein, 29.04.2011