„Beobachtungskonzept" wird die Oberfläche genauso wie bei AngularJS bei Datenänderungen automatisch aktualisiert. Darüber hinaus besitzt Knockout allerdings keine weiteren Funktionen. Somit müssen zwangsweise weitere Bibliotheken wie etwa jQuery eingesetzt werden. Knockout eignet sich daher sehr gut für kleinere Anwendungsfälle. AngularJS hingegen kann tendenziell immer verwendet werden, kann viele Vorteile aber erst bei größeren Anwendungen ausspielen.

Marc André Zhou ist als freiberuflicher .NET-Technologieberater tätig. Seine Schwerpunkte liegen im Bereich .NET-Entwicklung, SharePoint, Softwarearchitekturen und Frameworks. Er ist als MCPD, MCITP und MCTS im Bereich SharePoint zertifiziert.

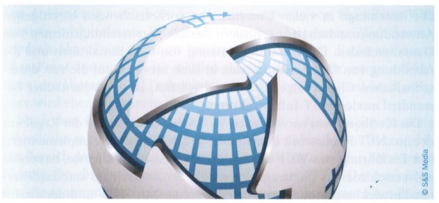
zhou@dev-sky.net

Links & Literatur

[1] AngularJS-Konzepte: https://docs.angularjs.org/guide/concepts

[2] Download AngularJS: https://angularjs.org/

[3] AngularJS Promise/deferred: https://docs.angularjs.org/api/ng/service/$q



© S&S Media

WS Designer für SAP-gestützte SharePoint Web Services

Web Services im Handumdrehen

Jürgen Bäurle

Seit geraumer Zeit bietet Microsoft Entwicklern von SharePoint-Lösungen die Möglichkeit, neben den im Standard mitgelieferten Web Services auch selbst entwickelte Dienste für SharePoint bereitzustellen. Einen auf dem SOAP- oder ODATA-Protokoll basierenden Web Service zu entwickeln, setzt solide Programmierkenntnisse voraus. Dieser Artikel zeigt, wie sich Web Services mithilfe eines Workflow-Designers visuell gestalten und Geschäftsdaten aus dem SAP-System in SharePoint integrieren lassen.

Die heutzutage in vielen Unternehmen vorherrschenden heterogenen Anwendungslandschaften erfordern flexible Bereitstellungsformen von Datenbeständen. Die Zusammenfassung von Funktionalitäten und die Abbildung von Geschäftsprozessen in Web Services, auf die von unterschiedlichen Clients zugegriffen werden kann, ist ein wesentlicher Bestandteil moderner IT-Infrastrukturen.

Die für SharePoint entwickelten Web Services werden in der Regel mit der im .NET Framework enthaltenen WCF-Technologie implementiert. Mit Einführung von WCF können unterschiedlichste Dienste, basierend auf verschiedenen Übertragungsprotokollen, mit einem standardisierten Entwicklungsmodell entworfen werden. Mit WCF und der ergänzenden .NET-Komponente WCF Data Services lassen sich klassische SOAP- sowie OData-Protokoll-basierte Web Services entwickeln. Trotz der Vereinheitlichung erfordert die Entwicklung solcher WCF-Dienste sowie deren Zusammenspiel mit SharePoint ein umfangreiches technisches Entwicklerwissen. Hinzu kommt, dass sich durch den wachsenden Bedarf, Geschäftsdaten eines SAP-Systems in SharePoint integrieren zu können, der Komplexitätsgrad der Entwicklung deutlich erhöht hat. Wünschenswert wäre, wenn sich die Entwicklungsarbeit auf die ohnehin komplexen Geschäftsprozesse eines Unternehmens konzentrieren könnte, ohne sich mit den technischen Implementierungsdetails von Web Services in SharePoint abgeben zu müssen.

In diesem Artikel wird der Web Service Designer (WS Designer) der Firma Theobald Software zur visuellen Gestaltung und Verteilung von SAP-gestützten SOAP- und OData-Diensten für SharePoint vorgestellt. Der Designer ist Teil der Produkt-Suite ERPConnect Services (ECS) [1].

ERPConnect Services und Web Service Designer

Für den Einsatz des WS Designers muss zunächst das Hauptprodukt ERPConnect Services von Theobald Software auf einer bestehenden SharePoint-Farm installiert werden. Der Hersteller bietet auf seiner Homepage eine Testversion zum Download an. Die Produkt-Suite ECS besteht aus mehreren Komponenten, die sich jeweils auf unterschiedlichste Art in das SharePoint-System einbinden und SAP-Anbindungsmöglichkeiten bereitstellen. Der WS Designer ist eine dieser Produktkomponenten und steht nach der Installation von ECS als Windows-Anwendung zur Verfügung.

Die Anwendung ermöglicht das visuelle Gestalten von SharePoint Web Services mithilfe eines Workflow-Designers. Der fertige Web Service wird als SharePoint-Solution bereitgestellt und kann direkt auf einem SharePoint-System deployt werden. Der Dienst basiert intern auf der oben erwähnten WCF-Technologie. Ein optionaler Visual-Studio-Export wird ebenfalls angeboten.

Nach dem Öffnen des Designers präsentiert sich eine überschaubare Oberfläche mit einer Ribbon-Leiste als Menü (**Abb. 1**). Im linken oberen Bereich des Hauptfensters werden Grunddaten zum Web Service wie Namen, Beschreibung oder Versionsnummer abgefragt. Darunter werden die so genannten Custom Data Types gelistet, Strukturen mit Eigenschaften von skalaren Datentypen. Custom-Datentypen werden vom WS Designer in .NET-Klassen übersetzt und als Parameter- und Variablendatentypen für die Operationen verwendet. Die definierten Operationen listet die Anwendung im rechten Bereich. Ein Doppelklick auf einen der Einträge öffnet den Workflow-Designer. **Abbildung 2** zeigt einen komplexen Beispielworkflow.

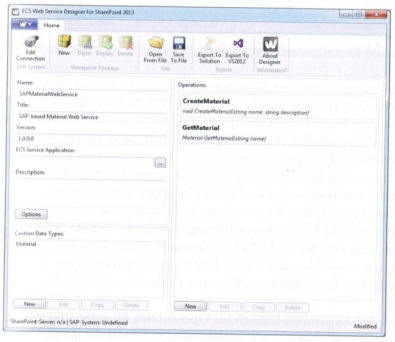


Abb. 1: Oberfläche des WS Designers mit Ribbon

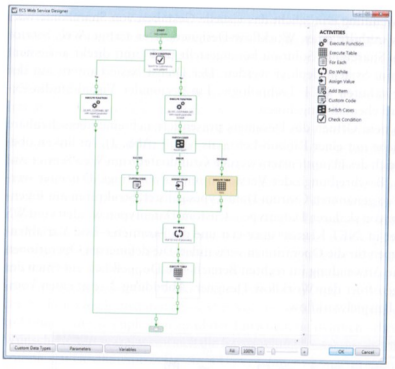


Abb. 2: Beispiel für einen komplexen Workflow

Die Anwendung verwaltet Web Services projektbasierend. Jede Projektdatei definiert genau einen SharePoint Web Service, der aus einer unbegrenzten Anzahl von Operationen (Methoden) besteht. Für jede Operation wird ein beliebig komplexer Workflow definiert, der sich aus Workflowaktivitäten zusammensetzt. Die Aktivitäten unterteilen sich in basis- und SAP-relevante Ablaufelemente. Die basisrelevanten Aktivitäten umfassen u. a. Elemente für bedingte Anweisungen (Check Condition Activity) und Verzweigungen (Switch Cases Activity), aber auch Elemente zur Schleifenverarbeitung (For Each oder Do While Activity) werden angeboten. Die SAP-relevanten Aktivitäten bestehen im Wesentlichen aus Elementen zum Lesen von Geschäftsdaten aus unterschiedlichen SAP-Objekten (Execute Table oder Execute Function Activity).

Neben den Aktivitäten können für Operationen Eingabe- und Ausgabeparameter sowie Variablen festgelegt werden. Die Variablen und Parameter können primitive oder Custom-Datentypen sowie deren jeweiligen

Listentyp annehmen. Custom-Datentypen werden auf Projektebene für alle Operationen übergreifend definiert und setzen sich aus den Eigenschaften primitiver Datentypen zusammen. Primitive Datentypen sind *Boolean*, *DateTime*, *Decimal*, *Double*, *Int32*, *Int64*, *Single* und *String*. Parameter lassen sich als *Input-*, *Output-*, *Return-* und *InputOutput-*Parameter verwenden. Komplexe, ineinander verschachtelte Custom-Datentypen lassen sich nicht definieren.

Der Datenaustausch und die Datenverarbeitung zwischen Aktivitäten erfolgt durch ein Mapping und ausschließlich über Variablen, Parameter und Festwerte, die je nach Leistungsumfang der einzelnen Workflowaktivitäten entsprechend zugeordnet werden. Neben skalaren Werten werden Listen durch die Zuordnung der Elementstruktur gemappt, d. h. für jedes Listenelement wird die gleiche Zuweisung durchgeführt. Festwerte können als skalare Werte oder in Form von Tabellen angegeben werden. Die Zeilenstruktur tabellenartiger Werte wird durch den Zielwert bestimmt. Der Wert der Spalte „Mapping Expression" in Mapping-Dialogen (**Abb. 3**) ist für die finale Zuordnung entscheidend und stellt einen C#-Ausdruck inklusive Platzhalter dar, der einen Wert vom Datentyp des zuzuordnenden Felds oder der entsprechenden Eigenschaft zurückgeben muss. Das ermöglicht die Formatierung und Umwandlung einzelner Werte oder die Festlegung dynamischer Datumswerte. Für einfache Mappings stellen die Dialoge Auswahlboxen mit 1-1-Feldzuordnung bereit, die nach Selektion den entsprechenden Platzhalter als Mapping Expression übernehmen.

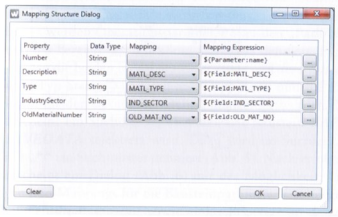


Abb. 3: Mapping-Dialog

SAP-Objekte

Bevor ein Beispiel den Einsatz des WS Designers zeigt, soll in diesem Abschnitt ein kleiner Exkurs eine kurze Einführung und Definition der wichtigsten Objekte in SAP bringen. Derzeit werden für die Datenextraktion nur SAP-Tabellen und Funktionsbausteine unterstützt.

SAP-Tabellen unterscheiden sich im Wesentlichen nicht von klassischen relationalen Datenbanktabellen, wie man sie vom SQL Server kennt. Funktionsbausteine sind Programmroutinen, die systemweit in einer SAP-Umgebung bekannt sind und von SAP-Programmen (ABAP) eingesetzt werden. **Abbildung 4** zeigt einen Funktionsbaustein in der SAP-Entwicklungsumgebung ABAP Workbench. Alle Funktionsbausteine, die betriebswirtschaftliche Standardvorgänge abbilden, werden in so genannten Business-APIs (BAPI) zusammengefasst und über das SAP Business Object Repository (BOR) verwaltet.

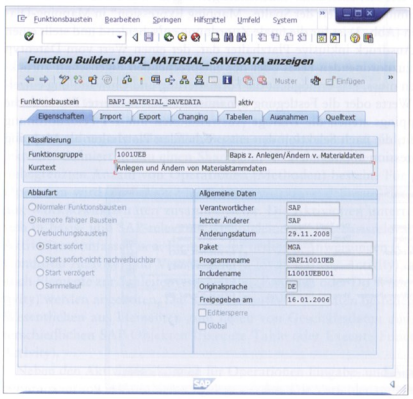


Abb. 4: Funktionsbaustein in ABAP Workbench

Beispiel: SAP-Material-Web-Service

Nun ist es an der Zeit, mit einem praktischen Beispiel die Möglichkeiten des WS Designers darzustellen. Der Web Service besteht aus zwei Operationen, einer Operation mit dem Namen *CreateMaterial* zur Neuanlage eines SAP-Materials sowie *GetMaterial* zum Lesen der Materialstammdaten.

Das Beispiel zeigt die Mindestanforderungen zum Erstellen und Lesen von Materialstammdaten in SAP. Ein Material besteht aus verschiedenen Ebenen, die in SAP Sichten genannt werden. Jede Sicht kann dabei mehrfach existieren, beispielsweise eine Sicht für Werk A und eine weitere für Werk B. Um das Beispiel einfach und überschaubar zu halten, werden nur die grundlegenden Daten für den Web Service verwendet. Die Grunddaten bestehen aus der Materialnummer, der so genannten alten Materialnummer, die aus einem ERP-Vorgängersystem stammen kann, der Sparte, dem Materialtyp und der Basismengeneinheit.

Für die Materialneuanlage wird in *CreateMaterial* der BAPI-Funktionsbaustein *BAPI_MATERIAL_SAVEDATA* verwendet. *BAPI_MATERIAL_GET_DETAIL* hingegen wird als Baustein zum Lesen verschiedener Materialeigenschaften in der Operation *GetMaterial* eingesetzt.

Im ersten Schritt zur Erstellung der Operationen müssen die SAP-Verbindungsdaten angegeben werden. Dies geschieht im *SAP-Connection*-Dialog, erreichbar über „Edit Connection" in der Ribbon-Leiste. Für eine genaue Beschreibung der notwendigen Daten muss auf die Dokumentation des Herstellers verwiesen werden. Die Neuanlage einer Operation erfordert zunächst die Eingabe von *CreateMaterial* als Operationsnamen. Daraufhin wird der Workflow-Designer angezeigt. Als Erstes werden Operationsparameter über den „Parameters"-Button im unteren Bereich des Workflow-Designers definiert. Für *CreateMaterial* werden die beiden Inputparameter *name* und *description* benötigt, beide vom Typ *String*. Mit Drag and Drop kann nun die *ExecuteFunction*-Aktivität der Designoberfläche hinzugefügt werden. Nach dem Drop erscheint ein Suchdialog, über den der Baustein *BAPI_MATERIAL_SAVEDATA* selektiert wird. Dazu wird im Suchfeld der Name oder mit „*" ein Suchmuster definiert (**Abb. 5**). Nach erfolgreicher Selektion erscheint ein Dialog (**Abb. 6**) mit der Möglichkeit, Labelbezeichnungen und Mappings für die Bausteinparameter festzulegen. Nachfolgend müssen die Parameter sowie einige Fixwerte an den Baustein mittels Mappings übergeben werden.

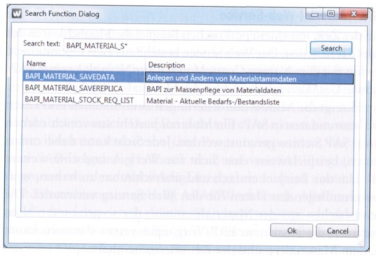


Abb. 5: Selektion des Bausteins „BAPI_MATERIAL_SAVEDATA"

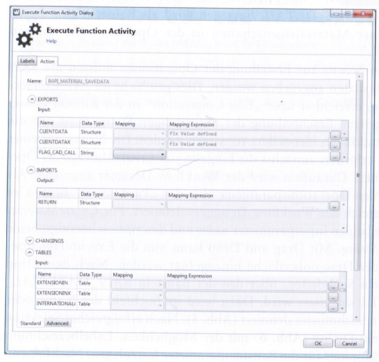


Abb. 6: Dialog zur Festlegung von Labelbezeichnungen und Mappings für die Bausteinparameter

Für *BAPI_MATERIAL_SAVEDATA* müssen im Exportparameter *HEADDATA* die Felder *MATERIAL*, *IND_SECTOR*, *MATL_TYPE* und *BASIC_VIEW*, in den Exportparametern *CLIENTDATA* und *CLIENTDATAX* jeweils die Felder *OLD_MAT_NO* und *BASE_UOM* gesetzt werden. Für das Mapping der Materialtexte wird der Tabelle *MATERIALDESCRIPTION* pro Sprache ein neuer Datensatz mit den Feldern *LANGU* und *MATL_DESC* angelegt. **Abbildung 7** zeigt das Mapping des Bausteinparameters *HEADDATA*. Dem Feld MATERIAL wird der Operationsparameter *name* durch den Platzhalter *${Parameter:name}* zugewiesen. Eine Liste der möglichen Platzhalter zeigt der Dialog an, der durch den Button rechts neben dem Textfeld geöffnet wird. Der Platzhalter wird durch einen Doppelklick in der jeweiligen Liste direkt ausgewählt (**Abb. 8**). Um ein besseres Verständnis für die anzugebenden Daten zu bekommen, wird der Artikel „Read/Write Material Master Data" [2] empfohlen.

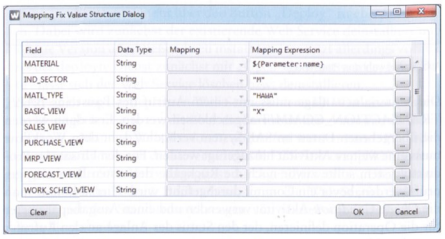


Abb. 7: Mapping des Bausteinparameters „HEADDATA"

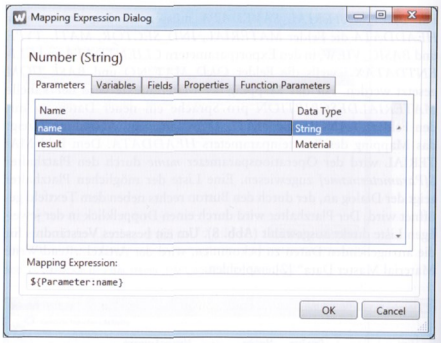


Abb. 8: Auswahl des Platzhalters

Die Materialneuanlage muss mit einem Aufruf des Bausteins *BAPI_TRANSACTION_COMMIT* abgeschlossen werden. Erst dann werden die übergebenen Daten im SAP-System verbucht, d. h. dem Workflow muss eine weitere Aktivität hinzugefügt werden. Für den Einsatz im Produktivsystem sollte zuvor noch die Rückgabe des ersten Bausteins geprüft werden, bevor ein Commit durchgeführt wird. Hierzu könnte man die *Check-Condition*-Aktivität verwenden und einen Ausgabeparameter für die Operation definieren, der den Status der Anlage an den Aufrufer zurückmeldet.

Als weitere Operation wird *GetMaterial* hinzugefügt. Diese soll einen Custom Data Type namens *MaterialInfo* mit Grunddaten an den Web-Service-Aufrufer zurückgeben. Dazu muss zunächst der Custom-Datentyp namens *Material* mit den fünf String-Eigenschaften *Number*, *Description*, *Type*, *IndustrySector* und *OldMaterialNumber* angelegt werden. Als Parameter für *GetMaterial* wird die Materialnummer vom Datentyp String benötigt. Anschließend wird die Aktivität zur Ausführung des Bausteins *BAPI_MATERIAL_GET_DETAIL* dem Workflow

hinzugefügt und der Exportparameter *MATERIAL* mit dem Eingabeparameter gemappt. Fertig ist der Web Service.

Deployment und Verwaltung eines Web Service

Bevor der Materialservice einsatzbereit ist, muss eine so genannte ECS-Serviceapplikation angelegt werden. Die Serviceapplikation führt den eigentlichen SAP-Datenzugriff in SharePoint durch und wird im Bereich „Service Applications" in der Zentraladministration erstellt. Die Eigenschaften legen die SAP-Verbindungsdaten und die Authentifizierungsmethode fest.

Das Deployment des Beispielservice kann direkt im Designer durchgeführt werden, wenn die Anwendung auf einem der SharePoint-Server ausgeführt wird. Der Button „Deploy" der Ribbon-Liste führt zunächst eine logisch-formale Prüfung durch, generiert automatisch den Quellcode des Web Service, kompiliert diesen, verpackt das Gesamtpaket in einer SharePoint-Solution und deployt diese direkt auf dem SharePoint-Server. Ein Re-Deployment über den Button „Deploy" ist ebenfalls möglich. Dabei wird zunächst der existierende Web Service deinstalliert und die neue Version anschließend neu installiert. Das setzt allerdings voraus, dass der deployte Dienst zunächst mit „Open" geöffnet wurde. Mit „Delete" lässt sich der Web Service löschen bzw. deinstallieren.

Zur Verwaltung der deployten Web Services bietet ERPConnect Services eine Managementseite an, die über die Startseite der Zentraladministration (über „Manage Web Services") abgerufen werden kann. Für jeden Dienst kann dynamisch die zu verwendende ECS-Serviceapplikation festgelegt oder ein SAP-Verbindungstest durchgeführt werden. Unabhängig vom hier beschriebenen Deployment, kann die Web-Service-Solution jederzeit mit den Standardwerkzeugen von SharePoint installiert und deinstalliert werden.

Verwendung eines Web Service

Da sich ein mithilfe des Designers erstellter und deployter Web Service programmtechnisch nicht von anderen Custom-WCF-Diensten für SharePoint unterscheidet, entspricht die Vorgehensweise zur Einbindung in eigene Programme dem Standard. Der Dienst wird über den Menüpunkt „Service Referenz hinzufügen" in Visual Studio und der Angabe dessen URL (mit MEX-Angabe) eingebunden. Das erzeugt den notwendigen Proxycode. Listing 1 zeigt, wie ein neues Material in SAP über den

veröffentlichen Materialservice in einem Konsolenprogramm angelegt und anschließend ausgelesen wird.

Listing 1

```
using(var client = new SAPMaterialWebServiceClient())
{
  client.CreateMaterial("SAMPLEJB1", "Jürgen's Material");

  var material = client.GetMaterial("SAMPLEJB1");

  Console.WriteLine("Number: {0}", material.Number);
  Console.WriteLine("Description: {0}", material.Description);
  Console.WriteLine("Type: {0}", material.Type);
  Console.WriteLine("Industry Sector: {0}", material.IndustrySector);
  Console.WriteLine("Old Material Number: {0}", material.OldMaterialNumber);
}
```

OData-Unterstützung

Neben der Gestaltung SOAP-basierter Dienste, wie im Beispiel gesehen, unterstützt der WS-Designer die Erstellung und Verteilung von Web Services basierend auf dem Open-Data-Protokoll, kurz OData. OData ermöglicht die Erstellung eines auf REST gestützten, ressourcenorientierten Datendiensts. Die einzelnen Ressourcen werden mithilfe von Uniform Resource Identifiers (URIs) identifiziert. Mit dem WS Designer konzipierte Dienste verwenden als Unterbau häufig ein ressourcenähnliches SAP-Datenmodell. Die im Beispielservice verwendete Materialstruktur eignet sich ebenso als Ressource für einen OData-konformen Web Service. Im Vergleich zum SOAP-Protokoll sind für OData allerdings einige Einschränkungen festgelegt, u. a. hinsichtlich der einsetzbaren Parameter einer Serviceoperation. SOAP-Dienste lassen sich im Gegensatz dazu sehr flexibel entwerfen. Zur Generierung eines OData-Diensts verwendet der WS Designer intern WCF Data Services. Eine umfangreiche Einführung in das OData-Protokoll würde den Rahmen dieses Artikels sprengen. Dokumentation und weitergehende Informationen stellt die OData-Website [3] zur Verfügung.

Bei der Neuanlage eines Web Servics im WS Designer muss zunächst im *Web-ServiceType Selection*-Dialog die Option zur Erstellung eines

OData-Datendiensts ausgewählt werden. Im oberen rechten Bereich der Anwendung werden daraufhin zusätzlich alle für einen OData-Service erstellten Ressourcen angezeigt. Ressourcen werden im Kontext des Designers als Entities bezeichnet. Ausgangspunkt zur Definition einer Ressource ist der bereits beschriebene Custom Data Type (CDT). Im Dialog „New Entity" zur Ressourcenerstellung muss ein zuvor definierter CDT, eine Eigenschaft des CDT als Key und der Listenname der Entität angegeben werden. Zu jeder Entität existieren in ihrer Signatur vordefinierte Methoden, die zum einen alle Entitäten lesen (*GetAll*) und zum anderen einzelne Entitäten erstellen (*Create*), ändern (*Update*) oder löschen (*Delete*) können. Die *GetAll*-Operation muss für jede Entität festgelegt werden. Die Definition der *Create*-, *Update*- und *Delete*-Methoden ist optional. Sämtliche Methoden werden über den Workflow-Designer gestaltet. Neben den Entitätsmethoden können für den OData-Dienst weitere Web-Service-übergreifende Operationen definiert werden. Die Vorgehensweise für das Deployment und die Verwaltung eines OData-Web-Service entspricht der oben beschriebenen.

Fazit

Mit dem Web Service Designer stellt Theobald Software ein sinnvolles Werkzeug dafür bereit, SAP-relevante Web Services für SharePoint zu designen, ohne dabei Programmcode erstellen zu müssen. Das erleichtert technisch versierten Anwendern oder wenig mit SharePoint erfahrenen Entwicklern die schnelle Einbindung eigener SAP-Dienste in das SharePoint-System. SAP-unerfahrene Entwickler wiederum können auf die im WS Designer bereitgestellte SAP-Anbindung zurückgreifen. Der WS Designer ist klar strukturiert und einfach in der Handhabung. Die Verteilung der automatisch generierten SharePoint-Solution ist für den Anwender bequem. Durch den Export eines im WS Designer erstellten Web Service nach Visual Studio können .NET-Entwickler das Visual-Studio-Projekt als Ausgangspunkt für weitere individuelle Anpassungen nutzen.

An dieser Stelle darf man aber nicht verschweigen, dass für den effektiven Einsatz einer Anwendung wie dem WS Designer sicherlich noch immer gewisses technisches wie betriebswirtschaftliches Hintergrundwissen zu Themen wie Web Services, SAP und SharePoint notwendig ist. Insgesamt betrachtet stellen das Produkt und die damit verbundene Vorgehensweise zur Entwicklung von Web Services einen für viele Anwen-

dungsfälle interessanten und sinnvollen Einsatzbereich dar, insbesondere dann, wenn in Unternehmen Aufgaben im Bereich der SAP/SharePoint-Integration zu lösen sind.

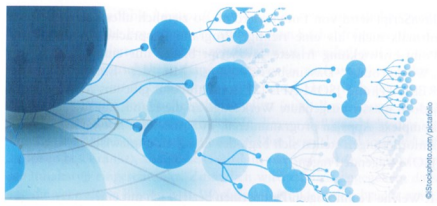
Jürgen Bäurle ist seit vielen Jahren als Softwarearchitekt und Entwickler selbstständig tätig. Der Schwerpunkt seiner Arbeit liegt in der .NET-Entwicklung, speziell im Bereich der SharePoint/SAP-Interoperabilität.
jbaurle@parago.de www.parago.de

Links & Literatur

[1] ERPConnect Services: www.theobald-software.com

[2] Read/Write Material Master Data: https://my.theobald-software.com/index.php?/Knowledgebase/Article/View/124/0/readwrite-material-master-data

[3] OData: http://www.odata.org



©iStockphoto.com/pictafolio

Wie man mit JavaScript und den neuen APIs am besten umgeht

JSOM und REST im Vergleich

Sebastian Schütze und Chris Spettmann

Microsoft hat mit der Einführung des App-Modells in SharePoint 2013 und dem Onlinebruder aus dem Office-365-Paket den Versuch gestartet, die Programmierung sicherer zu machen und hin zu mehr Standardisierung zu führen. Für die Programmierung von Apps, besonders von SharePoint-hosted Apps, kommt JavaScript mit den CSOM- und JSOM-APIs (Client-Side Object Model bzw. JavaScript Object Model) zum Tragen. Welche Vor- und Nachteile diese APIs haben und in welchen Situationen sie zum Einsatz kommen, soll der folgende Artikel exemplarisch an einer Search-driven App erläutern.