Kapitel 1

# Ein erster Überblick über die VSTO

**In diesem Kapitel:**





Microsoft Office 2007-Programmierung – Das Entwicklerbuch, © 2008 Microsoft Press Deutschland, ISBN 978-3-86645-415-6

In diesem Kapitel geht es um einen ersten Überblick über die *Visual Studio Tools für Office* (VSTO). Dieses Kapitel ist für jene Leser gedacht, die noch keine Erfahrung im Umgang mit den VSTO gesammelt haben und/oder sich erst einmal einen Überblick über die zahlreichen neuen Begriffe verschaffen möchten. Dieses Kapitel ist daher auch entsprechend allgemein gehalten. Tiefer gehende Erläuterungen, Beispiele, noch mehr Beispiele und vor allem Code folgen in den nächsten Kapiteln, so viel sei bereits versprochen.

# Installation und Versionsvielfalt

Ganz ohne Formalitäten geht es leider nicht. Da die VSTO nicht an Office, sondern an Visual Studio »gekoppelt« sind, gibt es keine »VSTO 2007«, sondern eine Version 3.0 als Teil von Visual Studio 2008 Professional oder Visual Studio Team System 2008. In diesem Buch stehen die VSTO 3.0 im Mittelpunkt. Viele Beispiele in diesem Buch, wie zum Beispiel die Automatisierungsbeispiele oder die Dokumenterweiterungen, funktionieren auch mit den VSTO 2.0, die ein separates Microsoft-Produkt auf der Grundlage von Visual Studio 2005 darstellen. Ferner gibt es ein »Zwischenupdate« mit dem offiziellen Namen »Visual Studio 2005 Tools for the Office 2007 System Second Edition« (VSTO 2005 SE) als freien Download für Besitzer einer Visual Studio 2005-Lizenz. Damit lassen sich Add-Ins für zahlreiche Office 2003/2007-Anwendungen und Multifunktionsleistenerweiterungen auf der Basis einer XML-Vorlage (immerhin) entwickeln. Wer Visual Studio 2008 Professional nicht anschaffen kann oder möchte, kommt auch mit dem Gespann aus VS 2005 Professional und VSTO 2005 SE relativ weit. Dieses kleine »Versionsbiotop« wird in diesem Buch allerdings nicht gehegt und gepflegt. Alle Beispiele und Erläuterungen beziehen sich offiziell auf die Version 3.0.

## Die Installation der VSTO

Zur Installation der VSTO 3.0 müssen (zum Glück) keine Worte verloren werden, da diese automatisch geschieht. Es muss lediglich darauf geachtet werden, dass die entsprechende Option beim Setup von Visual Studio 2008 nicht abgewählt wird. Ansonsten stehen die VSTO 3.0 nach erfolgreicher Installation in Gestalt zusätzlicher Vorlagen bei C# und Visual Basic zur Verfügung. Spezielle Menübefehle oder andere Tools gibt es nicht. Weitere Infos finden sich unter *http://msdn2.microsoft.com/de-de/library/54ds2za4.aspx*.

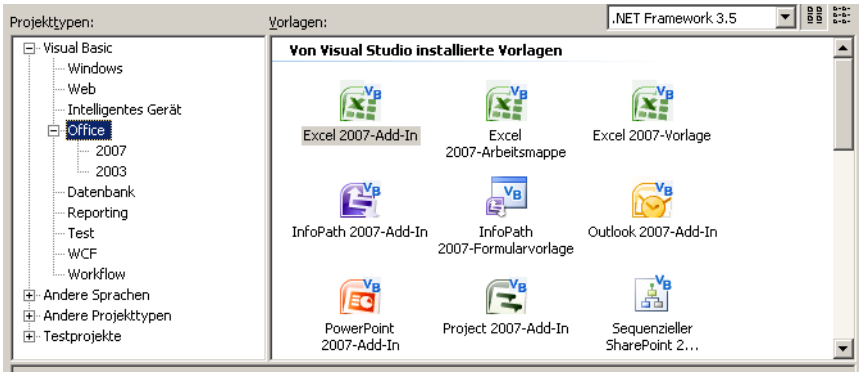


**Abbildung 1.1** Die VSTO treten bei Visual Studio 2008 durch einen Satz von Vorlagen in Erscheinung

**HINWEIS** Während die VSTO bei Office 2003 die Professional Version voraussetzen, genügt bei Office 2007 die Standard Edition.

### Welche Office-Version ist die richtige?

Natürlich immer die neueste, sagt zumindest Microsoft. Ganz so einfach dürften es sich die meisten Leser aber nicht machen, denn es gibt immer Gründe, nicht sofort die neuesten »Fashiontrends« aus Redmond mitzumachen. Für die VSTO ist bei Office 2003 die Professional-Variante Voraussetzung, darunter geht es leider nicht. Bei Office 2007 kommt dagegen jede Edition infrage[1]. Ob Office 2003 oder Office 2007, hat für den Einsatz der VSTO nur insofern eine Bedeutung, als dass es Merkmale wie Multifunktionsleisten, Word-Inhaltssteuerelemente und die Outlook-Formularbereiche natürlich nur in der aktuellen Office-Version gibt.

Mit den VSTO 3.0 lassen sich Dokument- sowie Anwendungserweiterungen sowohl für Office 2003 als auch für Office 2007 erstellen. Wird eine Office 2007-Vorlage gewählt, kann als Dokumentformat das alte binäre Format von Office 2003 und auch das neue, auf Open XML basierende Format gewählt werden. Präferenzen haben die VSTO in diesem Punkt nicht, da das Dateiformat für eine VSTO-Anwendung im Allgemeinen keine Rolle spielt.

**HINWEIS** Das ist ein wichtiger Hinweis. Es ist offiziell nicht möglich, mit den VSTO auf einer »Maschine« sowohl für Office 2003 als auch für Office 2007 zu entwickeln. Das Motto heißt »entweder – oder«. Wer bei seiner Entwicklung mit den VSTO 3.0 beide Office-Versionen einbeziehen möchte, muss zum Beispiel per Virtual PC 2007, Virtual Server 2005 R2 oder *VMWare Workstation* (alle drei gibt es kostenlos[2]) eine virtuelle Maschine einrichten (das setzt eine weitere Lizenz des verwendeten Betriebssystems voraus) und dort alles noch einmal installieren. Visual Studio 2008 bietet zwar immer alle Vorlagen für beide Office-Versionen an. Dass eine Vorlage zur Verfügung steht, bedeutet aber nicht, dass sich das Projekt auch anlegen lässt. Eine Office 2003-Vorlage kann nur dann genutzt werden, wenn die entsprechende Anwendung im Rahmen von Office 2003 Professional ab SP1 lokal installiert ist. Die Workflow-Vorlagen setzen zum Beispiel voraus, dass ein SharePoint-Web ansprechbar ist.

## Wie alles anfing

Wer sich seiner Vergangenheit nicht bewusst ist, ist dazu verdammt, sie zu wiederholen. So oder ähnlich mahnen Geschichtslehrer ihre Schüler manchmal, wenn sich diese scheinbar nicht für die Vergangenheit und die damit zusammenhängenden Details oder für Lehren, die man aus der Vergangenheit ziehen sollte, interessieren. Es fällt manchmal auch etwas schwer, Interesse für punische Kriege oder die Thronbesteigung von Karl dem Kahlen zu bekunden[3], denn die Gegenwart und vor allem die Zukunft ist im Allgemeinen sehr viel interessanter (und vor allem gegenwärtiger) als die Vergangenheit. Warum sollte man sich also mit Details der VSTO 1.0 oder 2.0 belasten, wenn doch die aktuelle Version 3.0 scheinbar alles besser kann? Nun, ganz so einfach gestrickt sind die Verhältnisse leider nicht. Zwar liegt es nahe, stets mit der aktuellsten Version zu arbeiten, doch bei den VSTO gibt es ein paar Gründe, warum dies nicht oder zumindest nicht sofort möglich ist:

- Es steht nur Visual Studio 2005 zur Verfügung, die aktuelle Version soll oder kann nicht angeschafft werden.
- Das .NET Framework 3.5 steht nicht zur Verfügung (es ist Voraussetzung für die Anpassungen für Office 2007).
- Es gibt unternehmensspezifische oder projektspezifische Gründe, warum das Projekt mit VSTO 2005 und .NET 2.0 fortgeführt werden muss.
- Die Anwendung muss auf Office 2003 basieren. Office 2007 liegt noch in weiter Ferne.

[1] Ob das auch für die ganz preiswerten Schülerversionen gilt, konnten die Autoren zwar nicht nachprüfen, sie gehen aber davon aus.

[2] Kaum zu glauben, aber wahr.

[3] Auch wenn das nicht direkt zum Thema gehört: Es gibt tatsächlich ein Buch, das sich mit der Frage beschäftigt, ob Karl der Kahle wirklich kahl war. Bei Amazon wird es ganz gut bewertet.

Microsoft Office 2007-Programmierung – Das Entwicklerbuch, © 2008 Microsoft Press Deutschland, ISBN 978-3-86645-415-6

Doch auch wenn keiner der Gründe zutreffen sollte und man sich nach dem Motto »Mit Volldampf voraus« auf die aktuelle Version stürzen kann, ein kurzer Blick zurück macht die Rolle der VSTO etwas deutlicher, die nicht erst gestern auf der Bildfläche erschienen, sondern bereits ein paar Jahre alt sind.

Alles begann, wie könnte es anders sein, mit der Version 1.0 der VSTO. Diese hatte funktional noch nicht viel zu bieten. Es gab keine eigene Laufzeit, sondern lediglich vier simple Vorlagen für Excel 2003 und Word 2003 sowie einen simplen »Loader«, der über zwei Dokumenteigenschaften dafür sorgte, dass mit dem Öffnen des Dokuments die über die zuständige Eigenschaft festgelegte Assembly geladen wurde. Eigentlich hätten die VSTO 1.0 ein kostenloses Add-On für Visual Studio .NET 2003 sein müssen, doch jemand aus dem Marketing war so clever, das Ganze mit Visual Studio .NET 2003, einer SQL Server 2005-Lizenz und der Access 2003 Runtime zu bündeln und für einige Hundert Euro zu verkaufen. Das ist zwar legitim, hatte aber zur Folge, dass die VSTO 1.0 lediglich eine relativ kleine Verbreitung erfuhren und viele an der Office-Entwicklung interessierte Entwickler sich ein wenig mehr darunter vorstellten, als tatsächlich im Paket enthalten war (nämlich nicht sehr viel). Aber die erste Brücke zwischen Office 2003 und .NET war gelegt.

Erst die VSTO 2.0 (auch VSTO 2005 genannt) war die erste richtige VSTO-Version. Es gab eine Laufzeit mit ein paar Objekten, die Host Controls, die Möglichkeit, Excel- und Word-Dokumente um ActionsPanes zu erweitern, und mit der Vorlage für Outlook-Add-Ins wurde ein komfortables Modell auf der Basis von Managed Code für die nach wie vor auf COM basierenden Office-Add-Ins zur Verfügung gestellt, das einige Schwächen des Shared Add-In-Modells ausbesserte. Auch die VSTO 2.0 waren noch ein separates Produkt und ebenfalls nicht gerade preiswert, da ein Visual Studio 2005 Professional (zwangsläufig) mit dabei war. Da kurz nach der Auslieferung der VSTO 2.0 Anfang 2006 bereits Office 2007 auf der Bildfläche erschien und klar war, dass es die nächste offizielle VSTO-Version erst mit Visual Studio 2008 geben würde, befand sich Microsoft schnell wieder unter Zugzwang, denn zwei Jahre konnte man die Entwickler nicht auf eine Version warten lassen, mit der sich zum Beispiel die Multifunktionsleisten von Office 2007 erweitern ließen. Daher erschien im Mai 2007 mit den *Visual Studio 2005 Tools for the 2007 Microsoft Office System Second Edition* (kurz VSTO 2005 SE) ein Produkt mit dem sicher genauso längsten wie verwirrendsten Namen in der Microsoft-Produktgeschichte, das als »Zungenverknoter« zahlreichen Küstenorten in Grönland zur Ehre gereicht hätte. Als kleine Entschädigung ist es ein kostenloser Download für Besitzer von Visual Studio 2005 Professional (die VSTO 2005 werden nicht vorausgesetzt) und bietet Add-In-Vorlagen für nahezu sämtliche Mitglieder der Office-Familie, sowohl für 2003 als auch für 2007. Trotz der aktuellen VSTO 3.0 stellt es immer noch eine attraktive Alternative für Entwickler dar, die in erster Linie an Erweiterungen (Add-Ins) auf Anwendungsebene interessiert sind (was anderes ist mit den VSTO 2005 SE auch nicht möglich, denn ihr Funktionsumfang ist nur eine Teilmenge des Funktionsumfangs der regulären VSTO 2005).

Mit der aktuellen Version 3.0 der VSTO haben die Office-Tools ein Stadium erreicht, das als ausgereift und funktional vollständig bezeichnet werden kann. Auch der Umstand, dass sie nicht mehr als separates Produkt gekauft werden müssen, sondern Teil von Visual Studio 2008 Professional sind, schafft ein wenig mehr Klarheit und »Planungssicherheit« und wird mit Sicherheit zur Verbreitung der VSTO-Funktionalität beitragen. Die Bezeichnung »VSTO« wird in Zukunft an Bedeutung verlieren, sodass mit der nächsten Version unter Umständen nur noch von »Office-Integration« die Rede sein wird[4]. Doch bis dahin werden noch ein paar Jahre vergehen. Mit der Version 2008 ist Visual Studio das Werkzeug für alle, die Erweiterungen für Office 2003/2007 mit Visual Basic oder C# entwickeln wollen. In diesem Buch stehen die VSTO daher weniger für eine bestimmte Version, sondern in erster Linie für diese Möglichkeiten.

[4] Allerdings noch nicht in diesem Buch, da wir zum einen nichts vorwegnehmen und zum anderen die sprachliche Vielfalt nicht unnötig steigern wollen.

## Was bieten die VSTO?

Dieser Abschnitt ist für Leser bestimmt, die zum ersten Mal mit den VSTO in Kontakt kommen. Es ist leider nicht ganz so einfach, einen vollständigen Überblick zu geben, da, was die Möglichkeiten der VSTO angeht, zwischen Office 2003 und 2007 (und streng genommen, das wurde aus dem letzten Abschnitt deutlich, auch zwischen den verschiedenen VSTO-Versionen) unterschieden werden muss. Tabelle 1.1 beschränkt sich der Übersichtlichkeit halber auf die wichtigsten Office-Versionen und die wichtigsten Verbesserungen bzw. allgemein Möglichkeiten, die die VSTO 2.0/3.0 für Entwickler mitbringen.

Die Aufgabe der VSTO ist schnell umschrieben. Diese erweitern Visual Studio durch Vorlagen um die Möglichkeiten, Erweiterungen (sie werden von Microsoft auch als »Customizations« bzw. »Anpassungen« bezeichnet[5]) in Visual Basic und C# auf der Grundlage des .NET Frameworks für Office-Anwendungen zu entwickeln. Die VSTO verbinden damit die Welt von Office mit der Welt des .NET Frameworks sowohl auf Anwendungsebene (durch Add-Ins) als auch auf Dokumentebene.

| Office-Version | Welche Möglichkeiten werden geboten? |
|---|---|
| Word 2003 | Anpassungen für Dokumente und Vorlagen sowie Erweiterungen von Dokumenten um ein ActionsPane und individuelle SmartTags. Das *Bookmark*-Objekt wird um Events und Datenbindung erweitert. |
| Word 2007 | Wie für Word 2003. Zusätzlich die Möglichkeit, die Multifunktionsleisten zu erweitern, Datenbindung bei Inhaltssteuerelementen und Anpassungen (Add-Ins) auf Anwendungsebene. |
| Excel 2003 | Anpassungen für Dokumente und Vorlagen sowie Erweiterungen von Dokumenten um ein ActionsPane und individuelle SmartTags. Das *Range*-Objekt wird in Gestalt des *NamedRange*-Objekts um Events und Datenbindung erweitert. In einem Excel 2007-Dokument, das im XML-Format gespeichert wird, lassen sich XML-Daten als »Custom XML Parts« unterbringen, die nicht angezeigt werden. |
| Excel 2007 | Wie für Excel 2003. Zusätzlich die Möglichkeit, die Multifunktionsleisten zu erweitern, und Anpassungen (Add-Ins) auf Anwendungsebene. |
| Outlook 2003 | Ein für Entwickler komfortableres Add-In-Modell, das mit den VSTO 2005 SE auch für die übrigen Office-Anwendungen zur Verfügung steht. |
| Outlook 2007 | Wie für Outlook 2003. Zusätzlich gibt es die Möglichkeit, die Multifunktionsleisten (für ein Inspektor-Fenster) zu erweitern, sowie einen »Designer« für Formularbereiche auf der Basis eines Benutzersteuerelements. |
| Project 2003 | Anpassung auf Anwendungsebene. |
| Project 2007 | Wie bei Project 2003, nur dass hier auch die Multifunktionsleisten erweitert werden können. |

**Tabelle 1.1** Diese Möglichkeiten stellen die VSTO für die wichtigsten Office-Anwendungen zur Verfügung

# Die Möglichkeiten der VTSO an Beispielen

Wer sich zum ersten Mal mit den VSTO beschäftigt (und damit mit den Möglichkeiten des .NET Frameworks 3.5, da beide untrennbar sind), wird unter Umständen bald vor lauter Wald die Bäume nicht mehr sehen. Da ist von so vielen neuen Begriffen die Rede, dass es schwer ist, den konkreten Nutzen für die Office-Entwicklung zu erkennen. Auch wenn dieser Nutzen im weiteren Verlauf des Buches sehr viel deutlicher werden wird, sollen in diesem Abschnitt bereits im Schnelldurchlauf die wichtigsten Einsatzgebiete der VSTO vorgestellt werden.

[5] Diesen Satz werden Sie in diesem Buch noch öfter lesen.

Microsoft Office 2007-Programmierung – Das Entwicklerbuch, © 2008 Microsoft Press Deutschland, ISBN 978-3-86645-415-6

## Dokumenterweiterungen

Eine Dokumenterweiterung bedeutet, dass mit dem Laden eines bestimmten Dokuments oder eines Dokuments, das auf einer bestimmten Vorlage basiert, ein .NET-Programm (Assembly genannt) gestartet wird. Das ist das typische Prinzip eines Add-Ins, nur dass sich dieses stets auf ein Dokument oder eine Vorlage bezieht. Microsoft nennt diese Erweiterungen auch *Anpassungen* (engl. »customizations«). Was die Erweiterung genau durchführt, spielt dabei keine Rolle. In der Regel wird sie eine neue Befehlsleiste oder eine oder mehrere neue Gruppen in der Multifunktionsleiste der Anwendung anlegen und dort Einträge hinzufügen, über die die Erweiterung gesteuert werden kann, oder sie wird ein *ActionsPane* anzeigen, das am rechten Rand des Dokumentfensters erscheint und Bedienelemente enthält und/oder Daten anzeigt, die zum Beispiel aus einer Datenbank oder einer LOB-Anwendung (*Line Of Business*) stammen.

Eine »Besonderheit« der Dokumenterweiterungsprojekte ist, dass das beim Anlegen des Projekts entweder neu angelegte oder ausgewählte Excel- oder Word-Dokument in Visual Studio erscheint und sich damit zum Beispiel benannte Bereiche, Textmarken, Inhaltssteuerelemente oder .NET-Steuerelemente direkt auf dem Dokument platzieren sowie XML-Verknüpfungen und Datenbindungen über Datenquellen, deren Inhalte direkt in dem Dokument angezeigt werden sollen, einrichten lassen. Das Dokument wird zum Teil der Anwendung.

Dokumenterweiterungen gibt es bei den VSTO nur für folgende Dokumenttypen: Word-Dokumente, Word-Vorlagen, Excel-Arbeitsmappen und Excel-Vorlagen. Jeweils für die Office-Versionen 2003 und 2007. Zusätzlich gibt es Dokumenterweiterungen auch für InfoPath 2007. Gegenüber den VSTO 2.0 bieten die VSTO 3.0 bei Dokumenterweiterungen keine Neuerungen. Mehr zu diesem Thema in Kapitel 9.

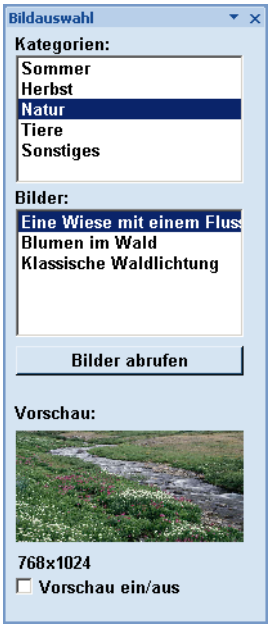


**Abbildung 1.2** Eine Dokumenterweiterung für Word, die ein ActionsPane anzeigt

## Anwendungserweiterungen

Dokumenterweiterungen sind, wie es der Name schon sagt, dokumentspezifisch. Das ist nicht immer gewünscht. Soll eine Erweiterung in der gesamten Anwendung zur Verfügung stehen, muss ein Add-In angelegt werden. Da der Begriff »Add-In« bei Office in der Vergangenheit beinahe inflationär verwendet wurde und man mit den VSTO nicht eine weitere Variante hinzufügen wollte, spricht Microsoft (was auch sehr sinnvoll ist) nicht von VSTO-Add-Ins, sondern von »Anpassungen auf Anwendungsebene«. Dahinter stecken die VSTO-Add-Ins, die wiederum »in Wirklichkeit« reguläre COM-Add-Ins sind. Sie werden mit dem Start der Anwendung geladen und fügen sich, genau wie die Dokumenterweiterungen, zum Beispiel mit Einträgen in eine vorhandene Befehlsleiste ein oder legen eine neue Gruppe in der Multifunktionsleiste der Anwendung an.

Anwendungserweiterungen gibt es bei den VSTO zu jeder »großen« Office-Anwendung, sowohl für 2003 als auch für 2007, mit Ausnahme von Publisher und FrontPage 2003[6]. Dies ist auch die wichtigste Neuerung der VSTO 3.0, da es bei den VSTO 2.0 nur für Outlook 2003 eine Add-In-Vorlage gab. Mehr zu diesem Thema in Kapitel 10.

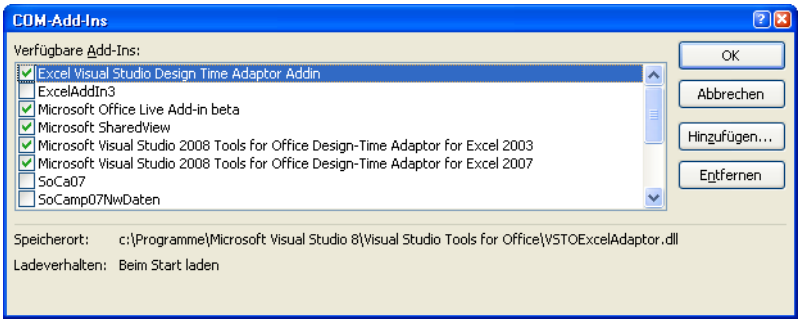


**Abbildung 1.3** VSTO-Anwendungserweiterungen werden von Office 2003/2007 wie reguläre Add-Ins behandelt

## SmartTags

Dieses Thema kommt in fast allen Betrachtungen zu den VSTO ein wenig zu kurz, dabei eröffnen auch SmartTags sehr interessante Möglichkeiten der Anpassung und sind zudem mit den VSTO noch sehr einfach zu erstellen. Ein SmartTag ist bei Office eine Liste mit Einträgen, die über die rechte Maustaste und das SmartTag-Symbol angezeigt werden, sobald die Anwendung ein zuvor vereinfachtes »Erkennungswort« im Dokumenttext identifiziert hat. Office selbst arbeitet intensiv mit SmartTags und es liegt nahe, diese Technik auch in einer Dokumenterweiterung zu nutzen.

Die VSTO bieten SmartTag-Erweiterungen für Excel- und Word-Dokumenterweiterungen (nicht für Add-Ins). Die Vorteile der VSTO liegen ganz einfach darin, dass das Erstellen von SmartTags ohne die VSTO (oder vergleichbare Hilfsmittel) eine recht knifflige Angelegenheit ist. Mit den VSTO besteht das Hinzufügen eines SmartTags zu einem Dokument aus wenigen Befehlszeilen.

[6] Das hat natürlich keine technischen Gründe, denn in einer frühen Vorabversion der VSTO 3.0 war auch eine Publisher-Vorlage dabei.

Die SmartTag-Unterstützung wurde mit den VSTO 2.0 eingeführt, bei den VSTO 3.0 hat es in diesem Punkt keine Änderungen gegeben. Mehr zu diesem Thema in Kapitel 9, in dem es allgemein um Dokumenterweiterungen geht.

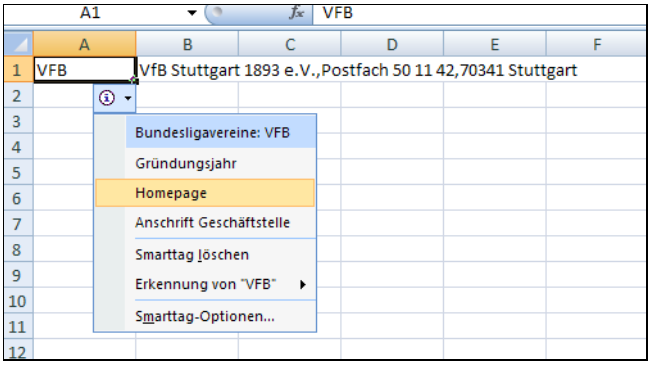


**Abbildung 1.4** Ein SmartTag besteht aus einer Auswahlliste, die eine Liste mit Einträgen anzeigt

# Die Rolle der Host Controls

Dieser Begriff ist genauso allgemein wie irreführend. Die Host Controls sind ein Satz von »Controls« (zu Deutsch »Steuerelemente«), die aus der VSTO-Laufzeit stammen, aber von der jeweiligen Office-Anwendung »gehostet« werden (daher der Name). Sie sind Teil der VSTO und nicht der Office-Anwendung. Es gibt insgesamt sieben Host Controls (Tabelle 1.2), wobei nicht alle gleich wichtig sind. Die interessanten Host Controls sind *NamedRange* für Excel und *Bookmark* für Word. Sie erweitern die Möglichkeiten des *Range*- und *Bookmark*-Objekts unter anderem um zusätzliche Ereignisse und die bei den VSTO allgegenwärtige Datenbindung. Die Host Controls wurden mit den VSTO 2.0 eingeführt, bei den VSTO 3.0 hat es in diesem Punkt keine Änderungen gegeben.

| Host Control | Bedeutung |
|---|---|
| *Bookmark* | Erweitert eine Textmarke (*Bookmark*-Objekt) bei Word 2003/2007. |
| *NamedRange* | Erweitert einen Zellbereich (*Range*-Objekt) bei Word 2003/2007. |
| *ListObject* | Erweitert eine Tabelle/Liste (*ListObject*-Objekt) bei Word 2003/2007. |
| *XmlMappedRange* | Steht für eine Verknüpfung eines Zellbereichs mit einem sich nicht wiederholenden Element einer XML-Datei bei Excel 2003/2007. |
| *Chart* | Steht für ein *Chart*-Objekt bei Excel 2003/2007. |
| *XmlNode3* | Steht für eine Verknüpfung zwischen einem Textelement und einem Element einer XML-Datei bei Word 2003/2007. |
| *XmlNode* | Steht für eine Verknüpfung zwischen einem Textelement und einem sich nicht wiederholenden Element einer XML-Datei bei Word 2003/2007. |

**Tabelle 1.2** Die Host Controls der VSTO

# Multifunktionsleisten erweitern

Multifunktionsleisten (im Original »Ribbons«) sind die auffälligste Neuerung bei Office 2007. Von vielen Anwendern (nicht Entwicklern) zunächst eher skeptisch aufgenommen, dürften sie sich inzwischen als »Fact of Life« etabliert haben. Viele Anwender, die am Anfang eher ablehnend waren, dürften inzwischen die Vorzüge des »Fluent UI« erkannt haben. Die Multifunktionsleisten sind keinesfalls nur eine optische Spielerei, sondern weisen mindestens zwei konkrete Vorteile auf. Sie zeigen immer nur jene Befehle an, die im aktuellen Kontext benötigt werden. Und sie bieten gegenüber den recht schlichten Befehlsleisten leistungsfähigere Bedienelemente, wie zum Beispiel Galerien, in denen jede zur Verfügung stehende Vorlage in Gestalt einer Vorschau angezeigt wird, und Auswahllisten, in denen zu jedem Eintrag auch eine Bitmap enthalten ist. Der Anwender profitiert von einer reichhaltigeren und trotzdem übersichtlicheren Funktionsleiste.

Die VSTO 3.0 stellen für das Erstellen eigener Multifunktionsleisten einen komfortablen Designer auf der Grundlage einer Vorlage bereit, der alle Möglichkeiten bietet, die für Multifunktionsleisten offiziell vorgesehen sind. Dies ist eine etwas angenehmere Alternative zur direkten Variante, bei der eine XML-Datei angelegt wird, durch die der Aufbau der Leiste vorgegeben wird. Mit anderen Worten, die VSTO 3.0 sind der Weg, um als Entwickler Multifunktionsleisten für Office 2007-Anwendungen anzulegen. Das Thema Multifunktionsleisten ist in Kapitel 17 an der Reihe.

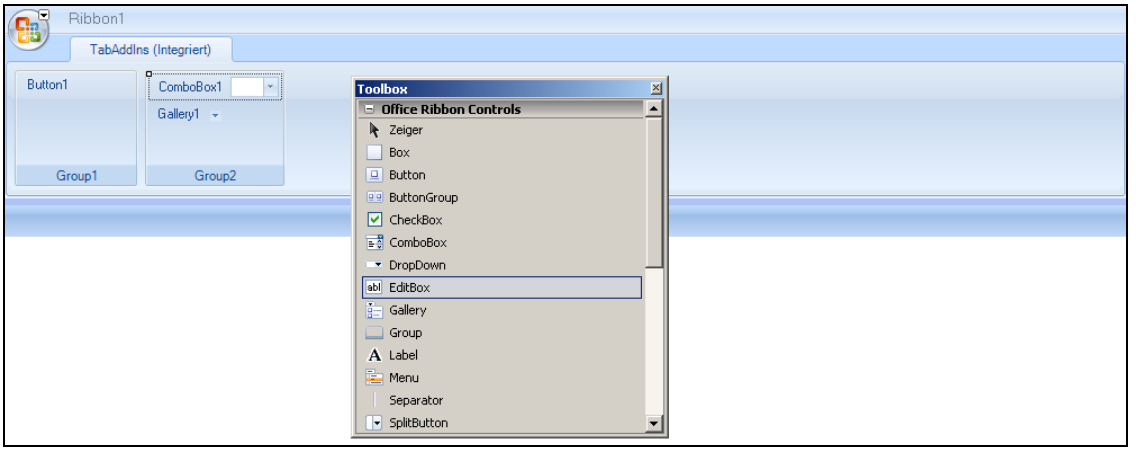


**Abbildung 1.5** Die VSTO 3.0 bieten für das Erweitern der Multifunktionsleisten einen komfortablen Designer

# Outlook-Formularbereiche

Mit Outlook 2007 wurden die Formularbereiche eingeführt, sie sind daher ein Merkmal von Outlook 2007 und nicht der VSTO. Ein Formularbereich erweitert oder ersetzt ein Outlook-Formular. Damit erhalten Anwender, in erster Linie aber Entwickler, die Möglichkeit, eigene Bereiche, in denen beliebige Daten angezeigt werden können, in einem Outlook-Formular unterzubringen. Outlook 2007 bietet von Haus aus einen eigenen Satz an (ActiveX-) Steuerelementen. Die VSTO 3.0 ermöglichen über eine Vorlage im Rahmen eines Outlook Add-Ins, dass anstelle der Outlook-Steuerelemente ein reguläres Benutzersteuerelement mit Win-Forms-Steuerelementen den Formularbereich bilden kann. Mithilfe eines Assistenten wird vorher festlegt, für welche Outlook-Formulare der Formularbereich auf welche Weise platziert werden soll. Alternativ kann ein Formularbereich auch auf einem bereits in Outlook 2007 erstellten Formularbereich basieren. Die VSTO

3.0 machen die Outlook-Formularbereiche damit so einfach erstellbar, erweiterbar und programmierbar wie zum Beispiel das ActionsPane bei Dokumenterweiterungen. Für viele Entwickler werden Outlook-Formulare damit zum ersten Mal auf dem Radarschirm erscheinen, was Outlook als »Informationsdrehscheibe« in Unternehmensanwendungen einen weiteren Schub geben dürfte. Mehr zu dem Thema in Kapitel 8.

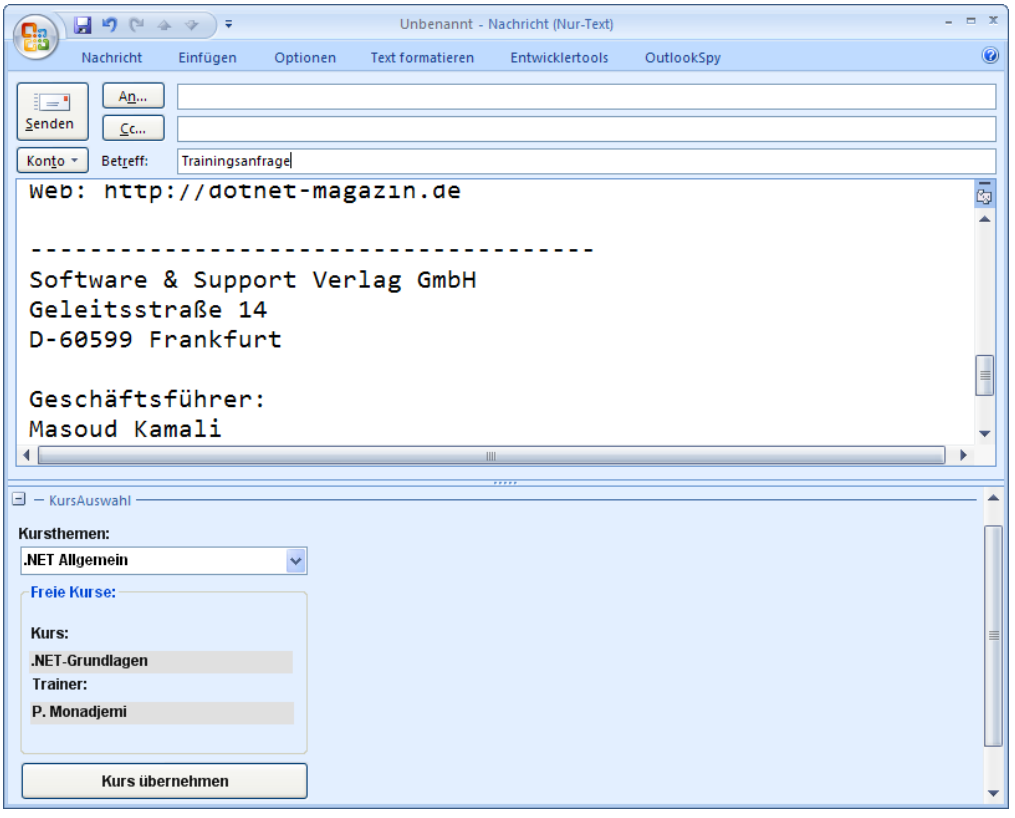


**Abbildung 1.6** Ein E-Mail-Formular wurde um einen Formularbereich erweitert

## Word 2007-Inhaltssteuerelemente

Inhaltssteuerelemente (engl. »content controls«) sind eine Neuerung, die mit Word 2007 eingeführt wurde, und die, wie die Outlook-Formularbereiche, zunächst vollkommen unabhängig von den VSTO oder einem Entwicklungsthema ist. Mit Inhaltssteuerelementen lassen sich Formularbereiche in einem Word-Dokument etwas komfortabler und wirkungsvoller anlegen, als das in der Vergangenheit mit den regulären Steuerelementen der Fall war. Der große Vorteil gegenüber den herkömmlichen Word-Steuerelementen ist, dass der Autor des Dokuments diese Elemente schützen (sperren) und der Anwender sie daher nicht verändern kann. Außerdem kann ein Inhaltssteuerelement Daten aus einer externen Datenquelle oder aus dem Dokument selbst anzeigen, die über eine XML-Schemaverknüpfung zugeordnet werden. Die VSTO 3.0 erweitern die Inhaltssteuerelemente, indem sie eine Datenbindung gegen einzelne Inhaltssteuerelemente bieten, sodass der Inhalt, den ein Inhaltssteuerelement anzeigt, direkt aus »unsichtbaren« Bereichen des Dokuments (den Custom XML Parts), einer externen XML-Quelle oder einer Datenbank stammen kann. Mehr zu den Inhaltssteuerelementen in Kapitel 7.

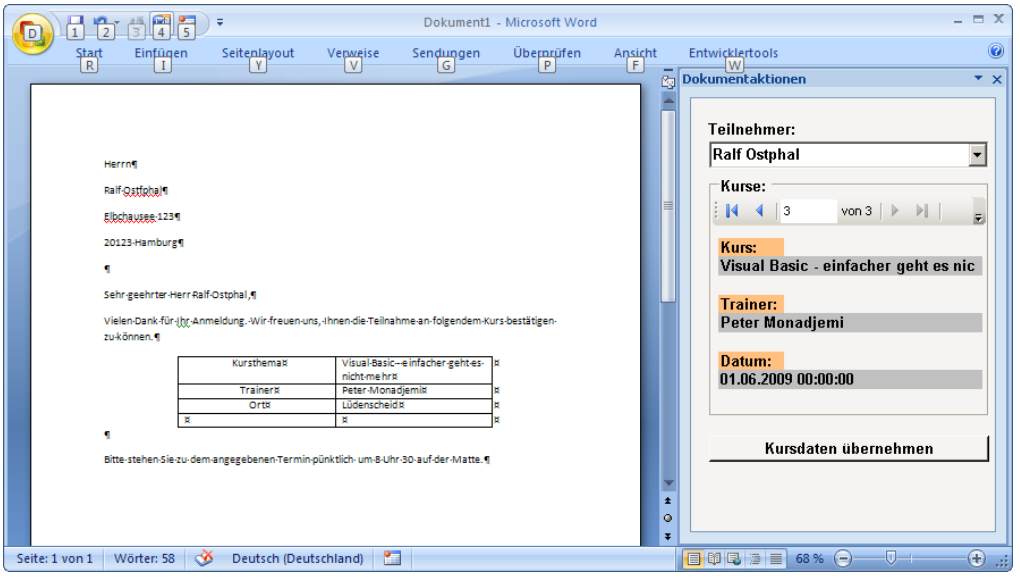


**Abbildung 1.7** Inhaltssteuerelemente zeigen in einem Word-Dokument Inhalte aus einer Datenbank an, die in einem ActionsPane ausgewählt werden

## Server-Dokumente und Dateninseln

Ein so allgemeiner Begriff wie »Server-Dokument« muss natürlich erst einmal spezifiziert werden, bevor deutlich wird, welche Möglichkeiten sich im Zusammenhang mit den VSTO ergeben. Ein Server-Dokument ist in diesem Zusammenhang ein Excel- oder Word-Dokument, das auf einem »Server« abgelegt wird. Ein Server ist wiederum ein regulärer Computer, auf dem Windows (im Allgemeinen aber Windows Server und nicht Windows XP oder Windows Vista) läuft und der von anderen Computern im Netzwerk in der Regel über den Browser angesprochen wird. Es geht also um ein typisches Intranet, zum Beispiel auf der Basis von SharePoint. Üblicherweise ruft ein Anwender eine (SharePoint-)Webseite auf, in der zum Beispiel eine Excel-Arbeitsmappe entweder direkt eingeblendet wird oder Daten aus der Mappe in der Webseite angezeigt werden. Und genau da beginnen die Probleme. Damit die Webanwendung (normalerweise wird sie auf ASP.NET basieren, wenngleich das keine Voraussetzung ist) auf die Excel-Arbeitsmappe zugreifen kann, muss sie dies bei Excel 2003 per Automatisierung tun. Dies bedeutet, dass Excel auf dem Server nicht nur installiert sein muss (was im Allgemeinen kein Thema ist), sondern auch als Anwendung gestartet werden muss. Und das nicht einmal, sondern bei jedem Abruf einer Webseite. Und da weder Excel noch Word für diese »Server-Einsätze« vorgesehen sind, kommt es in diesem Zusammenhang zu Problemen, die sich nicht programmiertechnisch lösen lassen.

Die VSTO bieten ab der Version 2.0 mit den »Dateninseln« eine Lösung, die aber sehr begrenzt ist. Eine Dateninsel ist ein Bereich innerhalb einer Excel-Arbeitsmappe oder eines Word-Dokuments, in dem theoretisch beliebige Daten abgelegt werden können. Für den Anwender sind diese Daten aber nicht sichtbar und er kann sie auch nicht direkt bearbeiten. Sie sind der unsichtbare Teil des Dokuments. Der Zugriff erfolgt bei Office 2003-Dokumenten, die noch im Binärformat vorliegen, über das *ServerDocument*-Objekt, das Teil der VSTO-Laufzeit ist. Die Idee ist, dass eine Arbeitsmappe, die auf dem Server liegt, Daten, die im Rahmen der Webanwendung angezeigt werden sollen, in den Dateninseln speichert, sodass diese per *ServerDocument*-Objekt einfach gelesen und geschrieben werden können, ohne dass es zu »Nebeneffekten« kommt. Denn Excel als Anwendung ist dabei nicht involviert (und muss auch nicht vorhanden sein), da die VSTO-Laufzeit

direkt auf die Excel-Arbeitsmappe zugreift. Ob dies wirklich etwas bringt, sei einmal dahingestellt. Die Dateninseln sind wohl in erster Linie der Versuch, eine Behelfslösung für ein Problem anzubieten, das eigentlich ganz anders gelöst werden müsste, indem zum Beispiel aus Excel eine echte Serveranwendung wird. Die neuen *Excel 2007 Services*, die ein »Excel auf dem Server« suggerieren könnten, sind übrigens keine Alternative, da sie nur ein Minimum an Excel-Funktionalität bieten (und Teil des nicht gerade preiswerten *Microsoft Office SharePoint Server 2007 Enterprise* sind).

Mit Office 2007 hat sich die »Sachlage« grundlegend gewandelt. Da Office 2007-Dokumente in einem XML-Format vorliegen, werden keine »VSTO-Spezialitäten« mehr benötigt, um auf die Dateninseln einer Excel-Arbeitsmappe zugreifen zu können. Das kann direkt geschehen über Bibliotheken, die entweder Teil der .NET-Klassenbibliothek sind, oder wie bei der Java-Bibliothek *OpenXML4J* nachträglich geladen werden müssen. Über sogenannte *Custom XML Parts* gibt es zudem die Möglichkeit, beliebige Daten in einem Office 2007-Dokument unterzubringen, die beim Laden des Dokuments nicht angezeigt werden.

## Dokumentzwischenspeicher (Data Cache)

Eng im Zusammenhang mit dem im letzten Abschnitt beschriebenen *ServerDocument*-Objekt steht das Prinzip der *Data Caches*, zu Deutsch »Zwischenspeicher für Daten«. Die Idee ist, dass sich beliebige Variablen durch Voranstellen des <*Cached*>-Attributs im Dokument speichern lassen, sodass sie mit dem nächsten Laden des Dokuments wieder zur Verfügung stehen (eine Alternative zur *Variables*-Eigenschaft in Word). Dies ist besonders für »Datenvariablen« wie ein DataSet attraktiv, da sich ein Dokument, zum Beispiel vor dem Abruf von einem Webserver, mit Daten »befüllen« lässt, die im Dokument gespeichert werden, aber für den Anwender nicht sichtbar sind, sodass das Dokument »auf Reisen gehen« kann und die Daten mit. Im Rahmen einer VSTO-Erweiterung können die Daten auch »offline« aktualisiert werden. Landet das Dokument später wieder auf dem Server, lassen sich die aktualisierten Daten mit den Daten auf dem Server abgleichen.

Die Dateninseln wurden mit den VSTO 2.0 eingeführt, mit den VSTO 3.0 hat es, wie beim *ServerDocument*-Objekt, keine Weiterentwicklung mehr gegeben. Auch das steht in dem Zusammenhang, dass ein Datenzwischenspeicher über *Custom XML Parts* in einem Office 2007-Dokument jederzeit direkt angelegt werden kann, sodass dazu kein spezielles VSTO-Hilfsmittel erforderlich ist.

## SharePoint-Anbindung durch Workflow-Vorlagen

Die VSTO 3.0 sind kein »Entwicklungswerkzeug« für SharePoint. Aus einem einfachen Grund, denn ein solches wird nicht unbedingt benötigt, da es dafür Visual Studio 2005/2008 und Vorlagen, zum Beispiel für Webparts, gibt. Die VSTO 3.0 bieten dennoch etwas für SharePoint-Entwickler, was es in dieser Form nur bei den VSTO gibt: zwei Projektvorlagen, mit denen sich auf der Grundlage der *Windows Workflow Foundation* Workflow-Komponenten erstellen lassen, die direkt in ein SharePoint-Web integriert werden. Das Thema SharePoint-Entwicklung ist in Kapitel 11 an der Reihe.

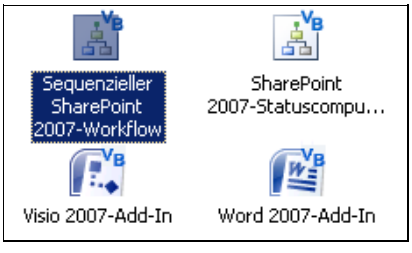


**Abbildung 1.8** Die VSTO 3.0 bieten zwei Vorlagen für das Entwickeln von Workflow-Komponenten für SharePoint

## War das alles?

Ja, mehr haben die VSTO in der Version 3.0 nicht zu bieten. Und das ist bereits eine ganze Menge. Die Zwischenüberschrift ist daher auch ein wenig ironisch gemeint, allerdings mit einem seriösen Hintergrund, denn die Office-Welt ist im Jahre 2008 mit Themen wie Office Communication Server, Office Live, Microsoft Office SharePoint Server usw. so vielfältig geworden, dass man der Meinung sein könnte, das alles müsste mit den VSTO auch irgendwie möglich sein. Nun, das ist es nicht, was aber nicht bedeutet, dass es nicht geht. Die VSTO sind für Erweiterungen auf Dokument- und Anwendungsebene zuständig, für die Brücke zwischen einer Office-Anwendung oder einem Office-Dokument mit einer .NET-Assembly. Soll eine VSTO-Anwendung vielleicht über einen Office Communication Server eine Instant Messaging-Sitzung starten oder auf die Inhalte einer Liste in einem SharePoint-Web zugreifen, ist dies natürlich problemlos möglich. Allerdings geschieht dies mit den Klassen der .NET-Klassenbibliothek oder einer speziell dafür zuständigen Assembly. Die VSTO sind in diesem Fall nur das Mittel zum Zweck. Unter diesem Blickwinkel betrachtet sind die Möglichkeiten der VSTO »unbegrenzt«, auch wenn die gewünschte Funktionalität nicht offiziell in der Liste der VSTO-Merkmale auftaucht.

# Die Neuerungen der VSTO 3.0

Die VSTO 3.0 bieten einige interessante Neuerungen, die in diesem Abschnitt stichwortartig aufgezählt und in den folgenden Kapiteln des Buches ausführlicher vorgestellt werden. Sofern es um einen Vergleich mit der letzten VSTO-Version geht, bezieht sich dieser auf die Version 2.0 und nicht auf die VSTO 2005 SE, da dies keine »echte« VSTO-Version war[7]. Tabelle 1.3 stellt die Neuerungen stichwortartig zusammen.

| Stichwort | Was steckt dahinter? |
|---|---|
| Add-In-Vorlagen | Das mit den VSTO 2.0 für Outlook 2003 eingeführte neue Add-In-Modell gibt es auch für alle übrigen Office 2003/2007-Anwendungen, wobei die Add-Ins bei den VSTO 3.0 per ClickOnce verteilt werden. |
| Ribbon-Vorlagen | Für die mit Office 2007 eingeführten Multifunktionsleisten gibt es einen komfortablen Designer. |
| Outlook-Formularbereiche | Für die mit Outlook 2007 eingeführten Formularbereiche steht für Outlook-Add-Ins eine Vorlage zur Verfügung. |
| Workflow-Vorlagen für SharePoint | Speziell im Zusammenspiel mit den SharePoint Services 3.0 und MOSS 2007 (und nur dort) gibt es zwei Vorlagen für das Erstellen von Workflow-Komponenten. |
| Word 2007 Content Controls | Dank dieser Controls wird das Ansprechen von Inhaltssteuerelementen in einem Word 2007-Dokument erleichtert. |
| ClickOnce-Auslieferung | Add-Ins und Dokumenterweiterungen werden über ClickOnce und einem auf Zertifikaten basierenden Sicherheitskonzept deutlich einfacher ausgeliefert als bei den VSTO 2.0. |
| VBA-Interoperabilität | Über eine Reihe neuer »COM-Eigenschaften« lassen sich die Komponenten eines VSTO-Projekts direkt in VBA-Projekten ansprechen und VBA-Code in einem VSTO-Projekt aufrufen. |

**Tabelle 1.3** Die wichtigsten Neuerungen der VSTO 3.0 auf einen Blick

[7] Eine unechte Version war es natürlich auch nicht, eher eine aus der »Not« geborene Zwischenversion.

## Add-Ins für alle Office-Anwendungen

Die VSTO 3.0 bieten scheinbar eine unüberschaubare Fülle von Vorlagen für Add-Ins, bei näherer Betrachtung stellt sich allerdings heraus, dass es doch nicht so viele sind. Für folgende Anwendungen sind Vorlagen für Erweiterungen auf Anwendungsebene verfügbar: Excel 2003 und 2007, Word 2003 und 2007, InfoPath 2007, Outlook 2003 und 2007, PowerPoint 2003 und 2007, Project 2003 und 2007 sowie Visio 2003 und 2007. Haben Sie mitgezählt? Es sind etwas mehr als ein Dutzend[8]. Nicht nur die Anzahl der Vorlagen wurde vergrößert, alle Add-Ins basieren auf einem einheitlichen Basismodell. Mehr zu diesem Thema in Kapitel 10.

## Dokumenterweiterungen

In diesem Bereich hat sich funktional nichts getan, außer dem Umstand, dass es die Dokumenterweiterungen auch für Excel 2007 und Word 2007 gibt (es stehen zum Beispiel keine Dokumenterweiterungen für PowerPoint oder Visio zur Verfügung).

## Vorlagen für Multifunktionsleistenerweiterungen

Das ist eines der Highlights der VSTO 3.0. Für die mit Office 2007 eingeführten Multifunktionsleisten (Ribbons) gibt es gleich zwei Vorlagen sowohl für Erweiterungen auf Anwendungs- als auch auf Dokumentebene. Eine basiert auf XML, die zweite auf einem wirklich komfortablen Designer. Eine neue Multifunktionsleiste lässt sich damit in wenigen Schritten und ohne, dass man etwas mit der zugrunde liegenden XML-Definition zu tun haben muss, erstellen. Mehr zu diesem Thema in Kapitel 17.

## Workflow-Vorlagen für SharePoint

Workflow, also das programmgesteuerte Umsetzen von Abläufen, die aus Teilschritten bestehen, die zeitlich nicht unmittelbar aufeinander folgen, und bei denen zwischen einzelnen Teilschritten Abhängigkeiten existieren, sodass ein Teilschritt erst dann ausgeführt werden kann, nachdem ein anderer Teilschritt abgeschlossen wurde, ist im Zusammenhang mit SharePoint ein wichtiges Thema. Für die Entwicklung solcher Vorgänge gab es bislang nur den SharePoint Workflow-Designer, der sich aber auf die Umsetzung einfacher Workflows beschränkte und in erster Linie für Anwender und nicht für Entwickler konzipiert wurde. Mit Visual Studio 2008 wird der Workflow-Designer offizieller Bestandteil von Visual Studio (bis dahin gab es nur eine Vorabversion). Die VSTO 3.0 bieten zwei Vorlagen, mit denen sich sequenzielle und auf Zustandsmaschinen basierende Workflows für SharePoint erstellen lassen.

## Die Word 2007 Content Controls

Dies ist keine direkte Eigenschaft der VSTO 3.0, sondern von Word 2007. Die Inhaltssteuerelemente (engl. »Content Controls«) wurden mit Word 2007 eingeführt und erlauben es, spezielle Eingabeelemente in ein Word-Dokument, wie zum Beispiel ein Textfeld, eine Auswahlliste oder ein Kalenderblatt, unterzubringen und es zu sperren, sodass der Anwender den Bereich nicht verändern kann. Insgesamt stehen sieben dieser Content Controls zur Verfügung. In den Bereich eines Inhaltssteuerelements lassen sich beliebige Daten »einblenden«, die zum Beispiel aus einer Datenbank stammen oder per XML-Schema dem Bereich zugeord-

[8] Die Vorlage für Publisher aus den VSTO 2005 SE ist nicht mehr dabei, was gewisse Rückschlüsse zulässt.

net werden. Die VSTO 3.0 bieten über die Visual Studio-Toolbox für Word 2007-Dokumente und -Vorlagen sieben Content Controls an, gegen die zum Beispiel eine (XML-) Datenbindung möglich ist, sodass der Inhalt einer Datenquelle automatisch im Inhaltsbereich erscheint. Mehr zu diesem Thema in Kapitel 7.

## Integration mit VBA

Die VSTO sind nicht der Nachfolger von VBA. Das ergibt sich bereits aus dem Unterschied, dass VBA eine Programmiersprache ist, die VSTO dagegen sehr viel mehr sind. Möchte man einen Vergleich ziehen, dann sind die VSTO der Nachfolger der durch VBA ermöglichten Erweiterung von Anwendungsfunktionalität. Zwischen VBA als Programmiersprache und den VSTO gibt es eine Brücke, die in beide Richtungen begangen werden kann. Vorhandene VBA-Makros lassen sich in einer VSTO-Erweiterung über die *Run*-Methode des *Application*-Objekts ausführen. Umgekehrt kann ein VBA-Makro öffentliche Funktionen in einer VSTO-Assembly aufrufen. Mehr dazu in Kapitel 4.

## Auslieferung über ClickOnce

Dies ist sicher die wichtigste Neuerung der VSTO 3.0. Sowohl Erweiterungen auf Anwendungs- als auch auf Dokumentebene lassen sich über ClickOnce sehr viel einfacher verteilen (bereitstellen), als es bei den VSTO 2.0 möglich war. Die VSTO 3.0 verzichten komplett auf Codezugriffsrichtlinien und arbeiten stattdessen mit Zertifikaten, die auch für die Signierung von VBA-Projekten benutzt werden können. Allerdings hat diese Neuerung einen kleinen schalen Beigeschmack. Sie funktioniert nur mit Office 2007 und setzt .NET 3.5 überall dort voraus, wo die Anwendung ausgeführt werden soll. Mehr zum Thema Ausliefern einer VSTO-Anwendung in Kapitel 14, das noch ausführlich auf die VSTO 2.0 und Office 2003 eingehen wird.

## .NET 3.5

Die VSTO 3.0 basieren auf dem .NET Framework 3.5 und dessen umfangreicher Klassenbibliothek, die gegenüber der Klassenbibliothek des .NET Frameworks 2.0 um so viele Bereiche und Klassen erweitert wurde, dass man sie in einem eigenen Buch beschreiben müsste. Tabelle 1.4 zählt die wichtigsten Neuerungen stichwortartig auf. Viele kleinere Neuerungen, die sich durch neue Klassen zwischen .NET 3.0 und .NET 3.5 ergeben haben, würden den Rahmen des Kapitels sprengen. Wichtig ist vor allem, dass alle diese Neuerungen bei den VSTO 3.0 genutzt werden können.

| Stichwort | Was steckt dahinter? |
|---|---|
| WCF | Die Windows Communication Foundation ermöglicht, dass ein »Host« anderen Anwendungen beliebige Funktionen über das Netzwerk zur Verfügung stellt, wobei die Details, wie zum Beispiel das zugrunde liegende Protokoll, extern konfiguriert werden. |
| WF | Die Windows Workflow Foundation ist die Workflow Engine, die auch in SharePoint eingesetzt wird und die über Klassen angesprochen wird. Workflows können in Visual Studio mit einem Designer zusammengestellt werden. |
| WPF | Die Windows Presentation Foundation sind eine neue GUI-Bibliothek als Nachfolger der WinForms, bei der eine Benutzerschnittstelle durch die XAML-Beschreibungssprache definiert wird. Die Stärken von WPF liegen unter anderem in einer auflösungsunabhängigen Darstellung basierend auf Vektorgrafik und einer Unterstützung für 3D-Grafik und Animationen. |
| CardSpaces | Mit CardSpaces wird ein Identitätsmanagement für Web- und Windows-Anwendungen möglich. |
| Packaging-API | Diese API ist als offizieller Teil der .NET-Klassenbibliothek für den Zugriff auf Open XML-Dokumente bzw. allgemein Dokumente, deren Aufbau der Open Packaging Convention entspricht, zuständig. |

**Tabelle 1.4** Wichtige Neuerungen bei .NET 3.0/3.5, die natürlich auch den VSTO 3.0 zur Verfügung stehen

## Dinge, die bei den VSTO 3.0 nicht dabei sind

Manche erfahrene Office-Entwickler werden vielleicht an dieser Stelle (erneut) ein »War das schon alles?« im Sinn haben. Ja, das war alles und es ist auch nicht gerade wenig. Nicht dabei sind unter anderem neue Objekte in der VSTO-Laufzeit und Klassen für einen vereinfachten Zugriff auf Open XML-Dokumente (mit Ausnahme des bereits erwähnten Word 2007 Content Controls)[9].

# Office-Programmierung mit .NET, aber ohne VSTO

Auch diese Variante muss in diesem Buch erwähnt werden. Die VSTO sind nicht erforderlich, um Office-Anwendungen mit Visual Basic .NET oder C# zu automatisieren. Es ist überhaupt kein Problem, Excel, Outlook oder Word von einem .NET-Programm aus anzusteuern. Das funktioniert bereits mit den »Uralt«-Versionen ab Office 97 und setzt auch keine Visual Studio-Lizenz voraus, sondern geht mit allen (kostenlosen) Express Editionen von Visual Studio, mit der Open Source-IDE SharpDevelop und theoretisch auch mit Notepad, indem die Befehle in einer Textdatei gespeichert und in der Kommandozeile mit einem Verweis auf die Office-Bibliotheken kompiliert werden. Ab Office 2003 gibt es diese PIAs (*Primary Interop Assemblies*) bei Microsoft zum Download, für Office 2000 und Office 97 müsste man sie sich mit dem Tool *Tlbimp.exe* aus dem .NET Framework SDK selbst anlegen, was schnell erledigt wäre. Da die Möglichkeiten aber jenen sehr ähnlich sind, die bereits seit vielen Jahren mit Programmiersprachen wie C++, Visual Basic 6.0 oder Delphi zur Verfügung stehen, wäre damit nicht allzu viel gewonnen. Auch Add-Ins für moderne Office-Anwendungen lassen sich theoretisch ohne die VSTO erstellen. Allerdings kommt in diesem Fall ein Modell zum Einsatz, das dem Entwickler etwas mehr »Handarbeit«, Erfahrung und Geduld abverlangt. Für die Entwicklung von Add-Ins sollten daher, sofern es keine konkreten Gründe gibt, die dagegen sprechen, die VSTO 2005 SE oder VSTO 3.0 verwendet werden.

Wer zum Beispiel mit Word- oder Excel-Dokumenten Operationen durchführen, auf deren Inhalte zugreifen, neue Dokumente anlegen oder Outlook-Ablagen durchsuchen, per Outlook E-Mails versenden oder vielleicht neue Kontakte oder Termine in Outlook anlegen möchte, benötigt dazu nicht die VSTO. Auch das Anlegen von Office 2007-Dokumenten auf der Basis von Open XML hat mit den VSTO (die dafür auch nichts beisteuern) nichts zu tun. Mehr zu diesem Thema in Kapitel 5.

Auch für sämtliche Beispiele in den »XML-Kapiteln« dieses Buches sind die VSTO ebenfalls nicht notwendig (die ließen sich mit wenig Aufwand auch auf VBA umstellen). Es ist daher sehr wichtig, die Einsatzbereiche der VSTO zu kennen. Diese sind Office-Erweiterungen auf Anwendungs- und Dokumentebene, die mit dem Start der Anwendung oder dem Laden des Dokuments aktiv werden, sowie die Anbindung an moderne Server-Anwendungen auf der Basis der Windows SharePoint Services 3.0 und Microsoft Office Server 2007.

# VSTO und VBA

Die VSTO mit VBA zu vergleichen ist wie der sprichwörtliche Vergleich zwischen Äpfeln und Birnen, es geht nicht bzw. ein solcher Vergleich lässt sich auf keine für alle befriedigende Formel bringen. VBA ist die Makrosprache aller Office-Anwendungen und in erster Linie für Makros und kleinere Erweiterungen gedacht (wenngleich einen niemand davon abhält, größere Erweiterungen zu entwickeln). Das wichtigste Merkmal von VBA

[9] Eine bekannte Lebensweisheit besagt: Es gibt (fast) immer eine nächste Version. Bei den VSTO wird dies mit Sicherheit der Fall sein. Ende Januar ergab eine Suche nach »VSTO 4.0« noch exakt 0 Treffer. Spätestens im November 2008 könnte aber eine erste »Alpha« verfügbar sein.

Microsoft Office 2007-Programmierung – Das Entwicklerbuch, © 2008 Microsoft Press Deutschland, ISBN 978-3-86645-415-6

ist, dass VBA-Erweiterungen immer Teil des Dokuments sind (bei Office 2007 gibt es dafür mit *.Docm*, *.Xlsm* und *.Pptm* eigene Dokumenttypen, die sich von den regulären Dokumenttypen durch den Umstand unterscheiden, dass hier VBA-Makros enthalten sind). VSTO-Erweiterungen liegen dagegen grundsätzlich extern vor. Beide Ansätze haben ihre kleineren Vor- und Nachteile, wobei die Tatsache, dass bei den VSTO der »aktive Code« nicht Teil des Dokuments ist, gerade unter dem Aspekt der Sicherheit eine hohe Attraktivität aufweist. Für die VSTO sprechen ganz einfach die sehr viel größeren Möglichkeiten, was die Integration mit anderen Systemen, etwa einer Datenbank, einem SharePoint-Portal und dessen Listen und auch »einer SAP«, angeht. Vieles von dem wäre mit VBA sehr umständlich oder gar nicht möglich. Der größte Vorteil der VSTO liegt darin, dass es hier um richtige Software-Entwicklung geht und nicht um Makroprogrammierung. Ein VSTO-Projekt basiert auf Quelltext, es kann im Team entwickelt werden, es ist eine Wiederverwendung von Quellen möglich, es kann metrischen Analysen unterworfen werden (wie viel Quellcode wurde in wie vielen Personentagen entwickelt?), es stehen im Rahmen der .NET-Klassenbibliothek moderne Mechanismen wie Tracing und Logging zur Verfügung und einiges mehr. Und mit Visual Studio existiert eine moderne, robuste und überaus komfortable Entwicklungsumgebung, die sowohl Profis als auch »Einsteigern« sehr viel zu bieten hat.

Gerüchte über das bevorstehende »Ableben« von VBA sind (wie so oft) anscheinend maßlos übertrieben. VBA als Programmiersprache mag inzwischen im Vergleich zu modernen Sprachen wie Visual Basic.NET und C# altbacken erscheinen, aber es geht hier nicht um einen Vergleich der Sprachspezifikationen. Für Excel- und Access-Anwender ist VBA nicht irgendeine Sprache, sondern der Mittel zum Zweck. Und das soll wenn möglich »für immer« so bleiben. Das verhält sich ein wenig so wie ein Wagen mit Formel-1-Motor, der einem Bauern, der seine Felder bestellen will, zunächst keine Vorteile bringt. Dass die Landwirtschaft aber Hightech mehr als aufgeschlossen ist, macht der Umstand deutlich, dass es kaum noch moderne Traktoren ohne Satellitennavigation geben dürfte und Satelliten zum Beispiel eingesetzt werden, um Felder aufzuspüren, bei denen »nicht lizenziertes Saatgut« eingesetzt wird. Bevor der Autor aber zu sehr vom Thema abschweift, bei Access und Excel ist .NET nicht sehr viel mehr als ein Schlagwort. Solange die VSTO nicht ganz konkrete Vorteile für das Arbeiten mit Excel bieten können, werden sie nicht viel mehr als eine theoretische Option sein.

Angeblich soll »Excel 14« die alten Excel-Makros nicht mehr ausführen können und die Möglichkeiten des Makrorekorders werden insofern reduziert, dass nicht mehr alle Vorgänge aufgezeichnet werden können. Gleichzeitig gibt es Gerüchte, dass der VBA-Editor mit »Excel 14« verbessert werden soll.

Ähnlich sieht es bei Access aus, bei dem die Anwender VBA als ein Bindeglied zwischen Datenbank und den Formularen nutzen und .NET und die VSTO hier keinerlei Vorteile bieten können[10].

Bei Word spielt VBA in erster Linie bei Vorlagen eine Rolle, sodass dem Anwender nach dem Anlegen eines Dokuments bestimmte Zusatzfunktionen über die Befehlsleisten angeboten werden können. Auch wenn es grundsätzlich kein Problem ist, diese Funktionalitäten mit Visual Studio im Rahmen eines VSTO-Add-Ins zu implementieren, würde dies keine Vorteile bringen und sogar den »Nachteil« aufweisen, dass die Erweiterung installiert werden muss, während sie früher einfach Teil der Word-Vorlage war. Das wäre bei bestimmten Anwenderkreisen nicht machbar[11].

Was in den letzten Absätzen beschrieben wurde, war jedoch die Sichtweise auf die Anwendung. Richtet man den Blick auf das Unternehmen, tritt die einzelne Anwendung in den Hintergrund. In einer Unternehmensperspektive sind Anwendungen wie Excel oder Word nur »Oberflächen« für den Zugriff auf Unternehmensdaten, mit denen man völlig unabhängig davon noch viele andere Dinge machen kann. Ein Mitarbeiter benutzt Excel auf

[10] Ein Microsoft-Mitarbeiter hat im VSTO-Forum allerdings angedeutet, dass Access-Unterstützung in einer »future version« der VSTO geplant sei.

[11] Zum Beispiel bei Autoren von IT-Fachbüchern. Hier wird es die gute, alte Word-Vorlage, die auch mit Word 97 funktioniert, noch in zehn Jahren geben.

zwei vollkommen verschiedene Arten. Zum einen traditionell, indem er oder sie Arbeitsmappen lädt oder neu anlegt und lokale Daten bearbeitet (hier wird VBA auch in Zukunft die gleiche Bedeutung haben wie seit Jahren). Zum anderen aber, indem Excel Daten in Arbeitsmappen anzeigt, die aus Unternehmensdatenbanken stammen. Hier ist VBA uninteressant, denn die Anbindung muss mit den VSTO und .NET vorgenommen werden.

In diesem Buch geht es vorwiegend um diesen modernen Ansatz. Natürlich kann .NET auch die Rolle von VBA in Bezug auf das Automatisieren und Erweitern von Office-Anwendungen übernehmen. Hier kommt es primär auf die Bereitschaft der Anwender an, den Schritt von VBA zu .NET zu machen und die nicht gerade geringe Lernhürde zu überwinden. Dazu sollen die folgenden Kapitel in diesem Buch behilflich sein.

## Was ist mit Access?

Microsoft Access ist eine Anwendung, die nur offiziell zum Microsoft Office-Paket gehört, ansonsten aber in vielerlei Hinsicht eigene Wege geht und eine »Welt« für sich ist. Mit den VSTO gibt es insofern eine Verbindung, dass die Access 2003 Developer Extensions und Runtime noch Teil der VSTO 1.0 waren, inzwischen sind diese optionalen Komponenten ein freier Download. Zwischen Access und VSTO existieren darüber hinaus keine Berührungspunkte. Das soll nicht bedeuten, dass bei Access keine Weiterentwicklung stattgefunden hätte. Auch Access 2007 hat die neuen Multifunktionsleisten erhalten, die sich per VBA problemlos erweitern lassen. Informationen dazu bietet zum Beispiel das *Office Developer Center* unter *http://msdn2.microsoft.com/en-us/office/aa905409.aspx*. Da es aber auch bei Access 2007 keine Brücke zu .NET gibt, wird Access in diesem Buch nicht behandelt. Mit einer Ausnahme: In Kapitel 12 wird gezeigt, wie sich Datenbankinhalte in einer VSTO-Anwendung darstellen lassen. Dabei geht es natürlich auch um Access-Datenbanken.

Als vorläufiges Fazit lässt sich Folgendes festhalten: VBA wird zwar von Microsoft seit vielen Jahren nicht mehr offiziell weiterentwickelt[12], aber mit an Sicherheit grenzender Wahrscheinlich auf absehbare Zeit ein fester Teil des Office-Pakets bleiben. Es gibt »Millionen VBA-Makros« in der ganzen Welt, die es auch noch in ein paar Jahren geben wird. Es existieren »Hunderttausende« von Anwendungen, die auf Access basieren, und die in sehr kleinen wie in sehr großen Unternehmen wichtige Aufgaben übernehmen[13]. Diese Anwendungen können (und sollen auch) nicht durch die VSTO ersetzt werden. Weder heute noch in einigen Jahren. Die einschlägigen Internetforen zu Excel, Word, PowerPoint und Outlook sind voller Einträge von Menschen, die Rat zu Themen rund um VBA-Makros suchen, daran wird sich vermutlich auch in Zukunft nichts ändern. Auch Access 2007 enthält keine .NET-Anbindung. Trotzdem wird Access auch in Zukunft noch eingesetzt werden (zum Beispiel als Reportgenerator auf der Grundlage von SharePoint-Inhalten), wenngleich sicherlich mit fallender Tendenz, auch wenn es weder von Microsoft noch von einer anderen Firma eine direkte Alternative gibt, die auf .NET basiert. Die Schlussfolgerung aus diesem Vergleich ist, dass VBA und VSTO in den nächsten Jahren parallel existieren werden. Die Zeit läuft gegen VBA (und damit auch »gegen« Menschen, die mit VBA-Know-how ihren Lebensunterhalt verdienen), insbesondere im Unternehmen sollten auch kleinere Projekte nicht mehr mit VBA begonnen werden. Dafür gibt es einige Gründe, die weniger etwas mit Features und Funktionalität, sondern mit IT im Allgemeinen und einem besonnenen und zukunftsorientierten Handeln zu tun haben:

[12] Die Tatsache, dass es ein VBA 6.1, 6.2, 6.3 und 6.4 gab bzw. gibt, hat nichts zu bedeuten, da funktional nie etwas hinzugefügt wurde. Angeblich hatte sich mit den Versionssprüngen das Marketing »einen Scherz erlaubt« bzw. man hatte wohl das Gefühl, dass sich eine Versionsnummer 6.0 im VBA-Editor über die Office-Versionen hinweg nicht gut machen würde.

[13] Die Autoren hoffen, dass diese »Mengenangaben« nicht unprofessionell klingen, aber es ist sehr schwierig bis unmöglich, halbwegs genaue Zahlen über die Verbreitung von VBA zu gewinnen bzw. diese Verbreitung abzuschätzen. Die von Microsoft früher gerne zitierte Zahl von 2 bis 3 Millionen VBA-Programmierern dürfte ein wenig zu optimistisch sein.

- VSTO-Lösungen sind im Allgemeinen sicherer, da VBA-Projekte nur selten digital signiert werden und beim Programmieren keine sicheren Techniken (Stichwort: Zugangsdaten für Datenbanken werden häufig sträflicherweise direkt im Quelltext hinterlegt) angewendet werden.
- Es ist sinnvoll, die Inhouse-Entwicklung auf eine einheitliche Grundlage zu stellen.
- Der »VBA-Individualprogrammierer«, der als Einziger weiß, wie eine Inhouse-Lösung funktioniert, auf die ganze Abteilungen angewiesen sind, ist ein Relikt der 90er-Jahre und für ein Unternehmen heutzutage ein gewisses »Risiko«. .NET-Know-how ist mittlerweile breiter gestreut als VBA-Know-how. Wer sich mit .NET auskennt, verfügt als Entwickler über eine solide Grundausbildung und ist in der Lage, über den sprichwörtlichen Tellerrand hinauszuschauen.
- Für .NET und damit für die VSTO gibt es eine Fülle von Werkzeugen, Erweiterungen und »Best Practices«.
- Die UserForms bei VBA sind auf dem Stand von Office 97. Moderne Benutzeroberflächen lassen sich nur mit den VSTO realisieren.
- Entwickler sind im Allgemeinen sehr viel zufriedener, wenn sie mit modernen Techniken arbeiten können. Auch wenn es bedeutet, dass sie nach einem Umstieg eine Weile unproduktiv sind, weil sie sich an die neue Umgebung gewöhnen müssen (und damit vielleicht eine Weile auch unzufriedener sind).
- .NET ist keine »new technology« mehr, die gerade erst freigegeben wurde und erst einmal in Ruhe reifen sollte. Das .NET Framework 1.0 wurde offiziell im Januar 2002 freigegeben, .NET ist als Technologie für Software-Entwickler über sechs Jahre alt und damit mit Sicherheit ausgereift. Auch die VSTO liegen inzwischen immerhin in der vierten Version vor.

Diese Argumente sprechen für sich. Mehr soll in diesem Buch zum Thema »VBA versus VSTO« auch nicht an Worten verloren werden. Die Zeit der »Glaubenskriege« zwischen Entwicklern im Speziellen und IT-Leuten im Allgemeinen ist ein weiteres Relikt der 90er-Jahre und damit im Jahre 2008 (weitestgehend) vorbei. Das Motto heißt friedliche Koexistenz. Linux mit Windows über Virtualisierung und überzeugenden und sympathischen Varianten wie Ubuntu Linux, C# mit Visual Basic, indem beide Sprachen vollkommen gleichberechtigt sind, und VBA mit VSTO (sofern es hier jemals eine Kluft gegeben hat[14]), indem Microsoft bei den VSTO 3.0 gewisse Brücken baut, die in Kapitel 4 vorgestellt werden.

# Auf den Weg zu den OBAs

Das Kürzel *OBA* steht bei Microsoft (seit einiger Zeit) für *Office Business Application*. Dies ist eine relativ neue Kategorie von Geschäftsanwendungen, die in den letzten Jahren durch das Zusammenwachsen der im Unternehmensalltag eingesetzten Systeme entstanden ist. Wie es bei solchen marketinggeprägten Schlagworten üblich ist[15], gibt es keine »harten Kriterien«, die festlegen, wann eine Anwendung eine OBA ist. Charakteristisch für eine OBA ist, dass moderne Office-Anwendungen auf der Anwenderseite im Mittelpunkt stehen, und dass die Anbindung an sogenannte *Line Of Business-Anwendungen* (LOB, ein weiteres Schlagwort der Branche) über moderne Schnittstellen wie zum Beispiel den *Business Data Catalog* (BDC) eines *Microsoft Office SharePoint Servers 2007* vorgenommen wird. Nachdem es unter anderem mit VBA und dem inzwischen ausrangier-

[14] Und wenn, dann basierte diese auf Missverständnissen wie der »Ankündigung«, dass ein »Office 14« kein VBA mehr enthalten würde.

[15] Microsofts Business Division, zu der auch Office System gehört, trug 2007 zu mehr als einem Drittel des Gesamtumsatzes des Konzerns von 51 Milliarden US-Dollar bei. Und das soll mit Sicherheit auch in Zukunft so bleiben.

ten *Information Bridge Framework* (IBF) in der Vergangenheit für Unternehmen mehrere Alternativen gab, OBAs zu entwickeln, hat Microsoft die VSTO (in der aktuellen Version 3.0) zu dem Werkzeug für die OBA-Entwicklung auserkoren.

Es ist wichtig zu verstehen, dass OBA voraussichtlich nicht ein weiteres dieser Kürzel ist, an die sich nächstes Jahr niemand mehr erinnern kann. Dahinter steckt ein Architekturkonzept für moderne Unternehmensanwendungen, das durch Abbildung 1.9 nur ansatzweise wiedergegeben werden kann. Einsatzbereiche für OBAs sind daher auch weniger die Abteilungsebene, sondern ganze Unternehmensbereiche, wie Materialbeschaffung, Logistik, Kostenrechnung, Angebotserstellung und überall dort, wo mit Office-Anwendungen auf der Grundlage von Daten, die sich an verschiedenen Orten im Unternehmen befinden, komplexere Entscheidungen getroffen werden müssen. Zu einer modernen OBA gehört auch, dass sich die beteiligten Personen über moderne Kommunikationsmittel wie *Instant Messaging* austauschen, um in einem Entscheidungsprozess zu einem Ergebnis zu kommen, und diese Form der Verständigung zum Beispiel in einen Workflow einbezogen wird. Dass es Microsoft mit dem Thema ernst meint, beweisen nicht nur die zahlreichen Vorträge und Ankündigungen auf Branchenkongressen, sondern auch der Umstand, dass es bereits eine Reihe von *Reference Architecture Packs* (RAPs genannt) gibt, die an einem Praxisbeispiel die Architektur einer OBA veranschaulichen sollen. Die VSTO und Visual Studio sind natürlich nur das Grundwerkzeug, mit dem sich die Anbindung eines Office-Frontends an eine LOB oder ein Workflow für ein SharePoint-Portal, aber auch Erweiterungen für SharePoint und BizTalk Server erstellen lassen. Das zugrunde liegende Architekturmodell gibt VSTO nicht vor und beantwortet auch nicht die Frage, wie man an die Daten einer LOB konkret herankommt. Das Thema OBA soll daher in diesem Buch auch nicht weiter vertieft werden. Lediglich ein Buchtipp zum Schluss. Wer mehr über die Möglichkeiten der OBA an Beispielen aus der Praxis erfahren möchte, dem sei das Buch »6 Microsoft Office Business Applications for Office SharePoint Server 2007« (erschienen bei Microsoft Press) empfohlen. Die Adresse vom OBA Teamblog lautet *http://blogs.msdn.com/oba/default.aspx*.

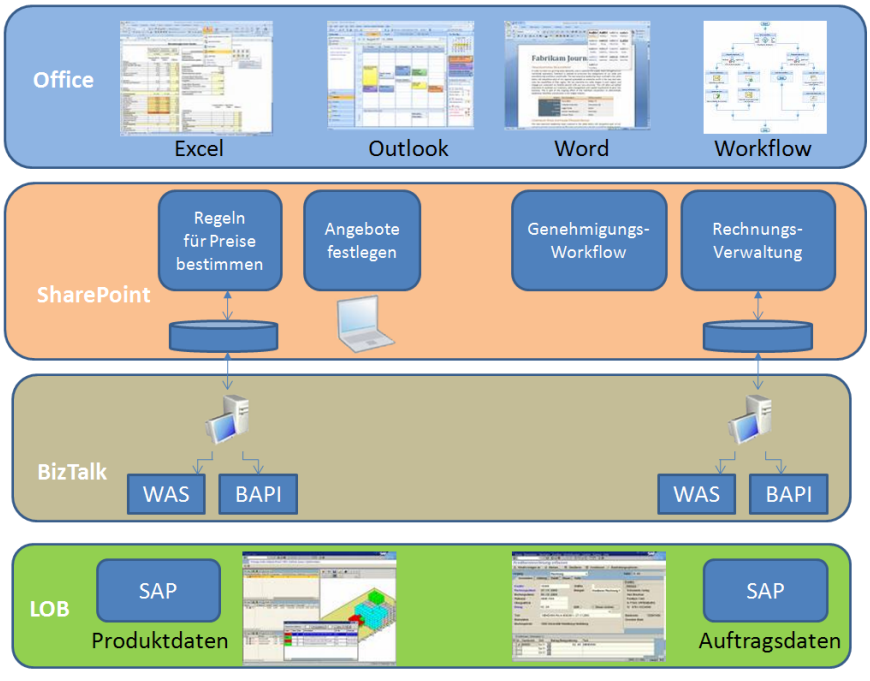


**Abbildung 1.9** Eine OBA integriert nicht nur Daten, sondern auch Prozesse mit einem »Office-Frontend«

Auch der Begriff VSTO soll sich nach dem Willen noch Microsoft nicht ewig halten, zumal es mit Visual Studio 2008 die VSTO als eigenständiges Produkt nicht mehr gibt. In Zukunft wird es wohl eher »Office-Integration« heißen.

# Wo gibt es Hilfe?

Für die VSTO gibt es (natürlich) eine ordentliche Dokumentation, in der alle wichtigen Themen ausführlich behandelt werden. Sie ist unter Umständen nicht so einfach zu finden, sodass es am einfachsten sein könnte, unter der Adresse *http://msdn2.microsoft.com/de-de/library/d2tx7z6d.aspx* ins *World Wide Web* zu gehen und die einzelnen Kapitel eventuell auszudrucken.

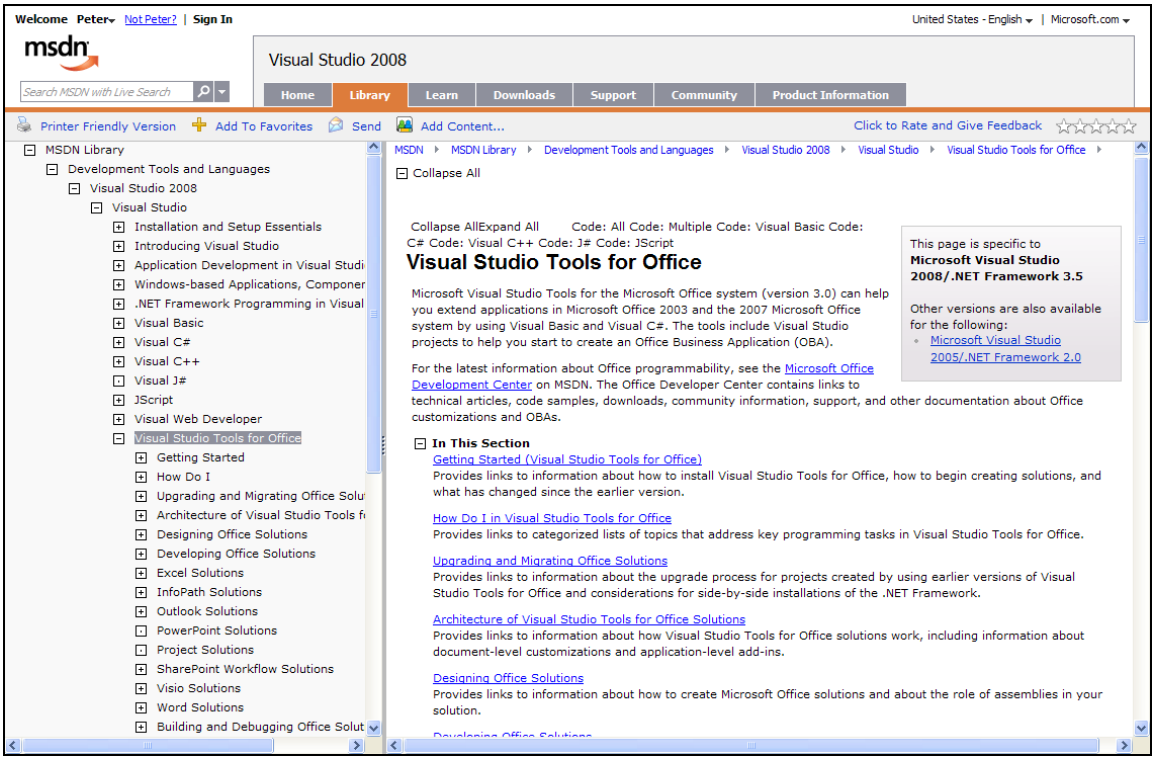


**Abbildung 1.10** Der Ersatz für das Handbuch – die Online-Dokumentation der VSTO im MSDN-Angebot (hier noch in der englischsprachigen Version)

Die Dokumentation ist zwar eine solide Grundlage, um einen ersten Überblick zu erhalten, löst im Allgemeinen aber keine Probleme, die in der Praxis auftreten. Die erste Anlaufstelle für derartige Probleme ist das *VSTO-Forum* auf MSDN. In der Newsgroup *microsoft.public.vsnet.vstools.office* ist dagegen nicht viel los. Wer heutzutage Probleme mit den VSTO hat, auf Fragen Antworten sucht oder einfach einen Kommentar loswerden möchte, wendet sich dazu an das MSDN-Forum unter der allgemeinen Adresse *http://forums.microsoft.com/msdn*. Nach dem Motto »wer nicht fragt, bleibt dumm« sollten Sie es ruhig einmal mit einer Frage (allerdings wenn möglich auf Englisch) versuchen.

Hier ein Beispiel:

*Hi all,*

*I have to write a book about VSTO (in German). Does anybody knows how this goes?*

*Thank you,*

*Peter*

Auch wenn das Beispiel nicht ganz ernst gemeint ist, es soll deutlich machen, dass der Stil in einem Forum stets »casual«, also ungezwungen höflich und vor allem ohne Floskeln ist. Ein Standardschema für ein »Posting« lautet »Allgemeine Anrede, Frage, Danke, Gruß«. Im Zweifelsfall eher sachlich und ohne sprachliche Schnörkel oder betont »lockere« Elemente. Das Problem sollte so knapp und so präzise wie möglich geschildert werden. Das erhöht die Antwortchancen im Allgemeinen am meisten. Erwarten Sie aber nicht sofort eine Antwort, manchmal kann es ein wenig dauern. Und wenn es einmal dringender sein sollte, müssten Sie die Frage unter Umständen erneut »posten«. Bedanken muss man sich für eine Antwort im Allgemeinen nicht, da dies nicht erwartet wird. Es sei denn, jemand hat sich für eine Antwort besonders viel Mühe gegeben. Möchten Sie über alle Aktivitäten im Forum auf dem Laufenden bleiben? Dann abonnieren Sie den RSS-Feed, indem Sie entweder auf den orangefarbenen Button oder den Link *Subscribe to RSS* klicken. Je nach ausgewählter Variante wird Ihnen der RSS-Feed als Lesezeichen oder direkt in Outlook (ab Version 2007) oder im Newsreader Ihrer Wahl angezeigt.

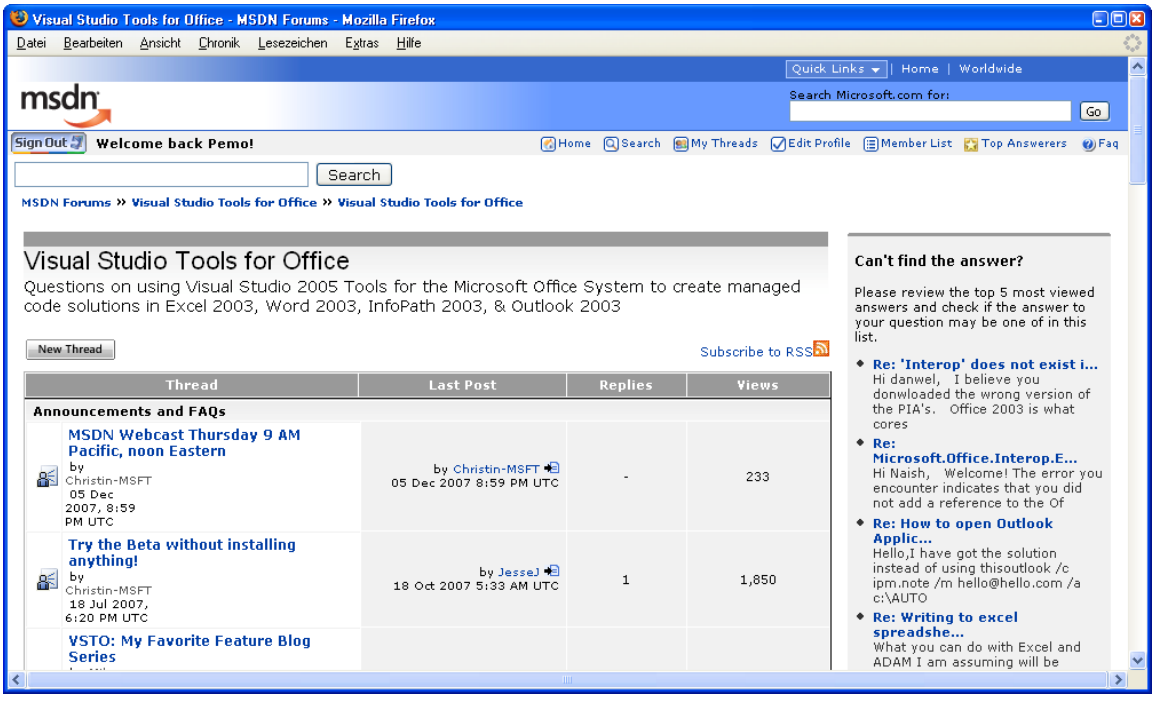


**Abbildung 1.11** Hier wird Ihnen unter Umständen geholfen – fragen kostet nichts

# Zusammenfassung

Mit den aktuellen VSTO 3.0 wird eine Entwicklung vorläufig abgeschlossen, die vor einigen Jahren mit den VSTO 1.0 als ein erster Versuch begann, Office mit .NET zu »verheiraten«. Die VSTO 3.0 bringen nicht nur Vorlagen für Add-Ins für alle wichtigen Office 2003/2007-Anwendungen, sondern vereinfachen durch ClickOnce die Verteilung von VSTO-Anwendungen. Auch wenn es bei den VSTO nicht so sehr darum geht, VBA abzulösen, und sich VBA mit dem VBA-Editor, dem Makrorekorder sowie dem Direktfenster zum schnellen Ausprobieren von Befehlen großer Beliebtheit erfreut und weit verbreitet ist – man denke nur an die Millionen im Einsatz befindlichen VBA-Makros, die in den nächsten Jahren nicht verschwinden werden –, sollten moderne Office-Erweiterungen auf der Basis der VSTO geplant und umgesetzt werden. Dass es nicht eine Entweder-oder-Entscheidung sein muss, macht der Umstand deutlich, dass sich die öffentlichen Funktionen von VSTO-Komponenten direkt in VBA ansprechen lassen.

### Wie geht es in diesem Buch weiter?

Im nächsten Kapitel werden die VSTO 3.0 an einem kleinen Beispiel vorgestellt. Dies ist für viele VBA-Programmierer unter Umständen der erste Kontakt mit Visual Studio und dem »neuen« .NET Framework. In den Kapiteln 3 und 4 werden Visual Studio 2008 als Nachfolger des VBA-Editors und Visual Basic (.NET) als Nachfolger von VBA vorgestellt.

Microsoft Office 2007-Programmierung – Das Entwicklerbuch, © 2008 Microsoft Press Deutschland, ISBN 978-3-86645-415-6

Microsoft Office 2007-Programmierung – Das Entwicklerbuch, © 2008 Microsoft Press Deutschland, ISBN 978-3-86645-415-6