



SAP in SharePoint

Vier Wege zur Enterprise-Application-Integration

von Frank Fischer

Wie bereits im Überblicksartikel zu SharePoint beschrieben, stellen SharePoint Technologies ein Portal für die Zusammenarbeit innerhalb von Unternehmen dar. Ein wesentlicher Aspekt hierbei ist die Integration weiterer Line-of-Business-Applikationen. In Deutschland sind das häufig mySAP-Systeme, wobei dieser Beitrag explizit so allgemein gehalten wurde, dass man die gezeigten Ansätze auch auf andere Systeme anwenden kann.

Zunächst stellt sich die Frage, wozu man überhaupt bestehende Anwendungen in ein Portal integrieren will? In den seltensten Fällen wird man die gesamte Funktionalität einer Applikation in einem Portal nachbilden, sondern sich von den möglichen Benutzern leiten lassen. Zumeist gibt es in Applikationen einige Funktionen, die von vielen Benutzern verwendet werden, aber normalerweise tief in den Menühierarchien verborgen liegen. Man macht es dann mit einem Portal seiner IT-Abteilung (die nicht eine Vielzahl von Benutzern ohne rechte Erfahrung auf eine Applikation loslassen muss) und dem Benutzer (der seine gesuchte Funktion jetzt offen liegen hat) einfacher.

Ein anderer Wunsch könnte sein, Daten oder Auswertungen von Daten aus solchen Applikationen dem Benutzer einfach zugreifbar zu machen. Beliebtes Beispiel ist der Zugriff auf die Arbeitszeit-Datenbank, welche dem Arbeitnehmer in seinem Portal bequem eine Übersicht über die derzeitige Summe der Überstunden gibt.

kurz & bündig

Die vier theoretischen Wege, bestehende Enterprise-Applikationen in SharePoint einzubinden, werden aufgezeigt. In der Praxis bewährt sich meist eine Mischung aus

- HTML-Capturing
- Daten-Capturing
- WSRP
- Punkt-zu-Punkt-Verbindung

Die vier Möglichkeiten

Die Übersicht in Abbildung 1 zeigt die vier theoretischen Wege, eine bestehende Enterprise-Applikation in ein SharePoint-Portal einzubinden, die in diesem Artikel beleuchtet werden. Hierbei soll ein wesentliches Augenmerk auf Vor- und Nachteilen sowie typischen Einsatzszenarien der unterschiedlichen Lösungen gelegt werden.

Das Feld der Lösungsaspiranten in Abbildung 1 ist in vier Quadranten eingeteilt. Die Lösungen auf der linken Seite bedürfen mehr oder weniger großer Programmieranstrengungen, wogegen die rechte Seite auf mehr oder weniger Infrastruktur setzt.

Programmatisch I: Web Capture

Der grundlegende Ansatz bei Web Capture ist es, eine schon vorhandene Web-Oberfläche einer Applikation zu verwenden, um diese in das Portal einzubinden. Hierbei gehen wir mit dem Begriff „Web-Oberfläche“ etwas großzügiger um und erlauben neben einer Benutzeroberfläche auch XML-Daten und Web Services. Aber der Reihe nach.

Bietet eine Applikation eine Web-Benutzeroberfläche, ist es ein Leichtes, diese mit einem Page Viewer oder Web Capture WebPart in eigene Seiten einzubinden. Diese beiden WebParts sind bereits bei Windows SharePoint Services oder Microsoft SharePoint Portal Server 2003 im Lieferumfang. Um diese zu verwenden, muss man nur die entsprechende URL angeben und los geht es. Hierbei muss man sich be-

wusst machen, dass SharePoint keine Ahnung über Daten oder innere Funktion der dargestellten Applikation hat. Es handelt sich also um eine pure Anzeigefunktion (was man offensichtlich schon am wahrscheinlich unterschiedlichen Design des Inhalts des Web Capturing WebParts und seiner Umgebung – dem Portal – feststellen wird). Das WebPart zieht hierfür das HTML von der angegebenen Quelle und fügt es in die Portal-Seite ein.

Von Vorteil ist es, wenn die Ziel-Applikation Windows-integrierte Authentifizierung verwendet oder sich mit dem Single-Sign-on-Service von SharePoint Portal Server anbinden lässt. Damit entfällt das häufig lästige Anmelden für den Benutzer.

Damit ist das Web Capturing aber noch nicht am Ende. Eine weitere Möglichkeit eröffnet sich, wenn man per HTTP Zugriff auf Daten der Applikation erhalten kann. Dies könnte im einfachsten Fall eine XML-Datei sein, kann aber bis hin zur dynamischen Datenbankabfrage gehen. Der

HTML-Capturing

Vorteile

- Sehr schnell konfiguriert
- Verwendet bereits existierende Oberfläche
- Wenig oder gar kein Programmieren notwendig

Nachteile

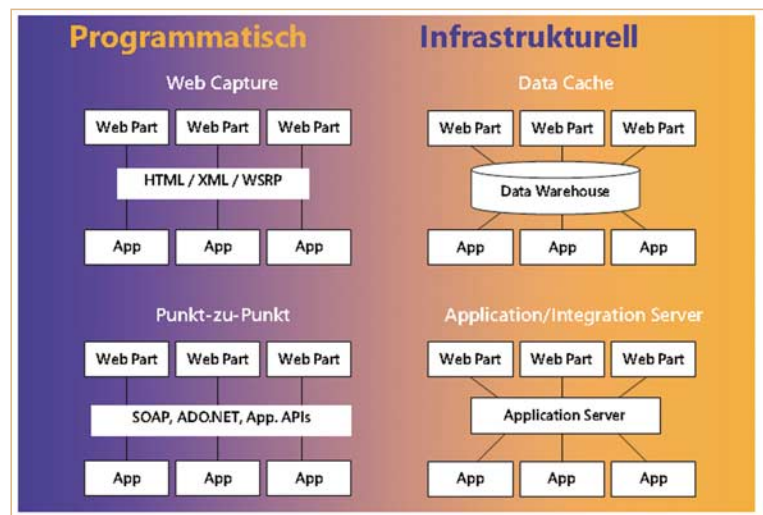
- Die Applikation muss eine HTML-Oberfläche haben.
- Transformation oder alternative Darstellung nicht möglich

Ansatz ist, dass ein SharePoint WebPart (zumeist das List View WebPart) zur Darstellungszeit diese Daten lädt und mittels einer XSLT-Transformation in HTML wandelt. Hierbei sind einfache Aggregationen und auch die Zusammenfassung verschiedener Datenquellen denkbar. Um die Erzeugung der entsprechenden Transformation kommt man allerdings nicht herum, so dass sich FrontPage 2003 als Editor für XSLT anbietet. FrontPage bietet die Möglichkeit, aus vorhanden Metadaten zu einer Datenquelle automatisch eine Transformation zu erzeugen, die sich dann auch sehr einfach weiterbearbeiten lässt.

Die Anforderung, verschiedene Portale miteinander zu verbinden, hat zur Entwicklung eines eigenen Standards in dieser Richtung geführt: „Web Services for Remote Portlets“ (WSRP) [2]. Wie der Name schon andeutet, kommt dies ursprünglich nicht aus dem Hause Microsoft, jedoch hat Microsoft in späteren Phasen sich an der Ausarbeitung beteiligt. Der Standard wurde bei OASIS eingereicht. Was steckt aber nun dahinter?

Die Idee ist, Portal A schnell und sauber mithilfe von Web Services in Portal B einzubinden. Hierbei sind keine Annahmen über die darunter liegende Technologie gesetzt, will heißen, WSRP ist plattform-

Abb. 1: Quadrant der möglichen Lösungen



unabhängig. Es fehlt in der Liste der beteiligten Firmen und Konsortien eigentlich niemand, der von Gewicht im Bereich der Web-Portale ist. Leider ist der Standard an sich noch recht jung und so mangelt es noch an konkreten Umsetzungen, was sich jedoch in naher Zukunft erledigen sollte. Der Ansatz für sich ist viel versprechend. Abbildung 1 stammt aus der Dokumentation der Arbeitsgruppe bei OASIS.

Die grundlegende Idee wird hieran deutlich: Ein oder mehrere WSRP Producer erzeugen Fragmente von Mark-Up (HTML oder XML) und stellen diese über SOAP zum Abruf zur Verfügung. Hierbei kann das Producer Portal den enthaltenen Portlets Hilfestellung beim Anbieten von WSRP bieten.

Aufgabe des Consumer Portals ist es nun, die Fragmente zu sammeln und dem Endverbraucher zur Verfügung zu stellen. Hierbei sind unterschiedliche Anzeigeformate denkbar wie einfaches HTML, aber auch WML, VoiceXML oder SALT. Das Consumer Portal kann allerdings noch

weitere Dienstleistungen erbringen wie zum Beispiel Caching. Sollte also WSRP in naher Zukunft Realität werden, hätte es folgende Vor- und Nachteile (siehe Kasten).

Für SharePoint gibt es bereits eine Implementierung des WSRP-Standards, die bei GotDotNet heruntergeladen werden kann. SharePoint von sich aus besitzt keine direkte Implementierung, denn auch hier gilt, dass dieser Standard später als das eigentliche Produkt kam. Für die nächste Version gibt es noch keine verlässliche Aussage.

Geht es um die Einbindung von SAP-Portalen, sollte man sich unbedingt auch das SAP iView WebPart Toolkit ansehen

Programmatisch II: Punkt-zu-Punkt
Der Ansatz, der hinter der Idee „Punkt-zu-Punkt“ steht, lässt sich kurz so umschreiben: Angenommen, unsere einzubindende Applikation bietet einen wie auch immer gearteten API. Dies kann im Fall von SAP zum Beispiel BAPI sein – Siebel dagegen bietet ein COM-basierendes Komponen-

Daten-Capturing

Vorteile

- Immer noch schnell konfiguriert
- Wenig Programmieren notwendig
- Passt sich in die Oberfläche und das Design des übrigen Portals ein

Nachteile

- Die Applikation muss Daten per HTTP zugreifbar machen.
- Nur einfache Aggregationen sind möglich.

WSRP

Vorteile

- Schnell und einfach
- Wenig bis mittlerer Programmieraufwand
- Plattform-agnostisch

Nachteile

- Einzubindende Applikation muss Portal sein und/oder WSRP-Dienste anbieten.

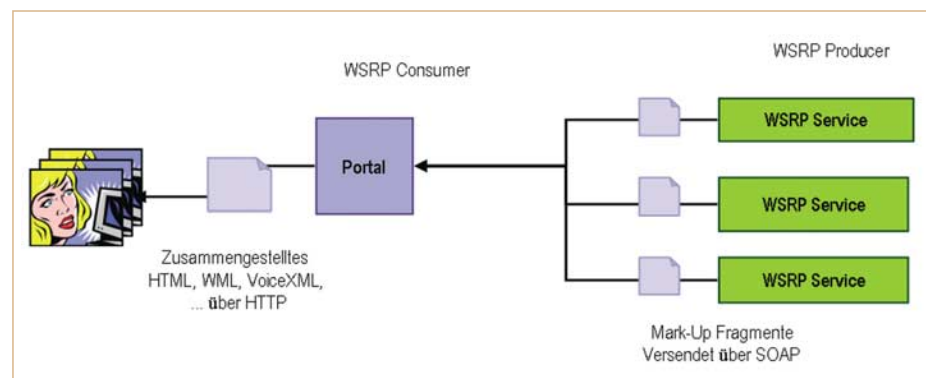


Abb. 2: Überblick zu WSRP

tenmodell. Weiterhin sollte der API natürlich den Umfang an Funktionen bieten, den man im Portal darstellen möchte.

Wie im SharePoint-Überblicksartikel schon gezeigt, ist es sehr einfach, zumindest ein rudimentäres WebPart zu schreiben und damit ein Interface für die entsprechende Applikation zu erzeugen. Dazu ist es dann notwendig, dass das WebPart die API der Applikation aufruft.

Konkret kann man das mit dem .NET Connector, der von SAP mittlerweile in der Version 2.0 zum Download angeboten wird. Dieser Connector erweitert Visual Studio .NET und ermöglicht das Erstellen des Proxy-Codes mittels der Anlage einiger Komponenten im Designer. Leitet man nun noch, wie im Überblicksartikel gezeigt, von der richtigen Klasse ab und überlädt

eine Funktion, ist das SAP WebPart schon fertig.

Bevor man sich jedoch mit der eigenen Herstellung beschäftigt, sollte man sich schlau machen, ob es derartige WebParts nicht schon fertig irgendwo gibt. Gerade bei den Marktführern wird man durchaus sehr schnell fündig; vielleicht nicht bei ihnen direkt, aber dann bei einem ihrer Partner.

Der Punkt-zu-Punkt Ansatz bietet viele Vorteile, die weiter unten aufgelistet sind. Man muss sich jedoch unbedingt vorab die Frage stellen, wie man den Benutzer auf die angesprochene Applikation abbildet. Fein raus ist man bei Windows-integrierter Sicherheit, da man hier Kerberos für sich arbeiten lassen kann. Eine Möglichkeit sind Applikationsbenutzer (also ein Account für viele Benutzer), denn deren Name und Passwort kann man zentral speichern (oder den Worker-Process des Portals unter diesem Account laufen lassen und wieder Windows-integrierte Sicherheit verwenden). Soll oder muss ein individueller Benutzer sich identifizieren, bleibt neben den typischen Fenstern mit der Aufforderung zur Anmeldung zu Anfang und dem Speichern eines Cookies, noch das Single-Sign-On im SharePoint-Sinne, auf das später noch eingegangen wird.

Zu dieser Liste noch ein paar erläuternde Worte: Die maximale Kontrolle und mögliche Wiederverwendung kommt aus der schlichten Tatsache, dass der jeweilige Programmierer wahrscheinlich sehr gut weiß, welche Form von Daten sie oder er abfragt. Da SharePoint zudem einen eigenen Caching-Mechanismus zur Verfügung stellt, kann man hier mit etwas Geschick sehr viel Last von den Servern der Applikation nehmen. Es ergeben sich auch wenige Abhängigkeiten von anderen Systemen oder Servern, da nur die Applikations- und die SharePoint-Server im Spiel sind. Damit reduziert sich der Aufwand bei Installation und Betrieb. Man muss sich aber im Klaren sein, dass man mehr oder weniger plötzlich eine Web-Oberfläche zu einem System baut, was auf so etwas nicht direkt vorbereitet sein muss. Die Frage nach Transaktionshandling und auch Multi-User-Fähigkeit des Back-End-Systems muss gestellt und ausreichend beantwortet werden. Wer glaubt, dass Enterprise-Systeme per se transaktionsorientiert arbeiten, sollte noch mal genauer hinse-

hen. Ebenfalls muss man sehr vorsichtig sein, wie viele Transaktionen über eine Connection gleichzeitig offen gehalten werden können, wie viele Connections im Pool sein können etc. Bitte auch daran denken, dass selbst ein kleines Balkendiagramm mit den aktuellen Firmenzahlen, wenn es bei jedem Aufruf des Portals erneut in der Back-End-Applikation generiert wird, irgendwann jeden Server überlasten kann.

Infrastrukturell I: Data Cache

Der letzte Satz des vorigen Abschnitts umschreibt ein Dilemma, in das man schnell laufen kann. Das magische Wort heißt *Caching*. Man kann nun in einem Fall die Ergebnisse eines Aufrufs zwischenspeichern, man kann aber auch einen Schritt weiter gehen und Zwischenergebnisse, Teile oder sogar die gesamte Datenbasis in Form eines Caches anlegen.

Bei diesem Vorschlag erntet man zunächst Gegenwehr aus zweierlei Gründen: Erstens muss man dafür Datenbank-Systeme oder Speicher vorhalten, was nach Kosten klingt, oder man hat Angst, dass man plötzlich mehrere Datenmaster hat und in die typischen Replikationsproblematiken steuert. Beides sind valide Bedenken und man sollte sie bei dieser Lösung nicht aus den Augen lassen. Sieht man sich den Einwurf der zusätzlichen Systeme an, muss man abwägen, inwieweit das bestehende System die zu erwartende zusätzliche Last tragen kann. Wenn Sie zum Beispiel ein Host-basierendes System haben, das Sie nun zusätzlich über ein Portal abfragbar machen wollen, können die Kosten, den Host für die Zukunft fit zu machen, einen notwendigen Server von der Stange schnell übersteigen. Falls das Angebot im Portal nicht lebensnotwendig ist und der Dienst auch mal ausfallen darf, tut es eigentlich ein einfacher Server, der nicht einmal ein Backup benötigt. Und an der Software ihres Hosts müssen sie auch nichts ändern. Klingt doch plötzlich gar nicht mehr so abwegig, oder?

bleibt die Frage nach dem Daten-Master. Und diese muss man für sich selbst klar beantworten. Zumeist wird diese Variante dann verwendet, wenn man zum Beispiel einen relativ statischen Datenbestand hat und diesen über das Portal zugreifbar machen möchte. Änderungen finden dann nicht über das Portal statt (vielleicht weil die meisten Nutzer eben nur lesen wollen).

Punkt-zu-Punkt-Verbindung

Vorteile

- Zugriff auf Applikationen, wie von deren Programmierer vorgesehen
- Maximale Kontrolle und Wiederverwendung von Daten möglich
- Keine weitere Middleware-Infrastruktur notwendig
- Wenig weitere Abhängigkeiten

Nachteile

- Jeweilige APIs müssen anprogrammiert und eine Oberfläche erstellt werden.
- Transaktionen und konkurrierende Zugriffe müssen von der Applikation abgebildet werden.
- Unter Umständen extreme Last auf Back-End-Systeme

Data Cache

Vorteile

- Wenig Last auf der Enterprise Applikation
- Sehr einfache Möglichkeit, Daten aus unterschiedlichen Quellen zu verbinden und/oder vorauszuwerten

Nachteile

- Eine Applikation muss die Daten in den Zwischenspeicher überführen und auf Konsistenz achten.
- Zurückschreiben steigert die Komplexität extrem.
- Rechtestruktur muss ebenfalls abgebildet werden

Damit bleibt der Datenmaster, wo er ist, es wird eine regelmäßige Replikation in Form einer Kopie auf den Data Cache durchgeführt und damit hat man zumeist am wenigsten Probleme.

Bleibt die Frage nach der Größe des Cache. Ein Kunde hat mir in diesem Zusammenhang einmal gesagt, dass sein Unternehmen über 3 Millionen Kunden hat und dies sich damit von selbst verbietet. Ok, gehen wir also einmal von 3 Millionen Datensätzen und – sagen wir – 1 Kilobyte Daten pro Eintrag aus. Mit ein wenig Mathematik sieht man leicht, dass sogar das Laptop-Sonderangebot des Discounters an der Ecke damit wenige Probleme haben wird. Und vielleicht lässt die neue Datenbank noch weitere Innovationen zu, man denke nur an das Information Bridge Framework für Microsoft Office. Also: Lassen Sie sich nicht durch diese Befürchtungen schrecken, sondern klären Sie nüchtern die genannten Fragestellungen.

Welche Tools benötigt man für diese Lösungsform? Zunächst einmal eine Datenbank, die die Daten zwischenspeichert. Bietet diese dann auch noch einen Agenten zur Datenreplikation, umso besser. Denkt man an Microsoft, fällt hier natürlich der

SQL Server zusammen mit dem DTS ins Auge. Das Darstellen der Daten lässt sich am einfachsten mit FrontPage 2003 bewerkstelligen.

Eine weitere Möglichkeit wäre, die Daten in eine SharePoint-Liste direkt zu speichern. Hierzu muss man das Objektmodell oder die Web Services von SharePoint bemühen (ja, das Datenbank-Schema von SharePoint ist im SDK beschrieben, aber dort steht auch explizit, dass sich dieses in zukünftigen Versionen ändern kann). Wirklich komplex ist dieser Ansatz auch nicht, bedarf aber schon eines gewissen Programmieraufwandes. Vorteil ist dann aber, dass der Benutzer die ganze Palette der Möglichkeiten hat (z.B. unterschiedliche Ansichten, Sortierungen usw.).

Infrastrukturell II: Application Integration Server

Funktionalität kann SharePoint Portalen durch WebParts auch mittels eines Application Servers bereitgestellt werden. Dies kann mehr oder wenig Infrastruktur und Programmierung erfordern. Zumeist sagt die Regel: je mehr Infrastruktur, desto weniger Programmierung.

Im einfachsten Fall verwendet man Windows Server und den darin enthaltenen Application-Server-Dienst. Damit stehen z.B. Transaktionsüberwachung und Message-Queueing zur Verfügung. Dieser von der Infrastruktur her weniger anspruchsvolle Ansatz belastet dann aber eher den Programmierer, weil hier mehr von seiner Seite gefordert wird und entspricht in weiten Teilen dem Punkt-zu-Punkt-Ansatz, mit dem Unterschied, dass hier die Transaktionsverwaltung von Windows durchgeführt wird.

Natürlich gibt es weitere Produkte, die die Erstellung und den Betrieb solcher Integrationen erleichtern. Von Seiten Microsofts fallen mindestens zwei ein: Zunächst einmal BizTalk Server, der von Hause aus eine Menge an Connectoren mitbringt und das Abbilden von Business-Prozessen sehr vereinfacht.

Im Bereich Punkt-zu-Punkt hatten wir bereits über eine Integration von zum Beispiel SAP über den .NET Connector und BAPI Aufrufen gesprochen. Der BizTalk Server bietet hierzu eine Alternative. So stellt der Microsoft BizTalk Adapter für die mySAP Business Suite in der Version 2.0 neben BAPI auch iDoc und RFC für Ein- und

Jetzt abonnieren und Vorteile sichern!

Riesen-Poster!



inklusive Jahres-CD



128 MB
USB-Stick!

Ihre Abo-Vorteile:

- Sie erhalten den 128 MB dot.net-USB-Stick*
- Sie sparen rund 10% gegenüber dem Einzelverkauf
- Sie erhalten das **Riesen-Poster** „.NET Framework 2.0“
- Mit der **Jahres-CD** erhalten Sie alle Ausgaben des vergangenen Jahres noch gratis obendrauf
- Sie verpassen keine Ausgabe
- Jede Ausgabe inklusive **Heft-CD** mit vielen Tools

* Nur solange der Vorrat reicht! Memory Stick wird nach Ausgleich der Abonnementrechnung zugesendet.

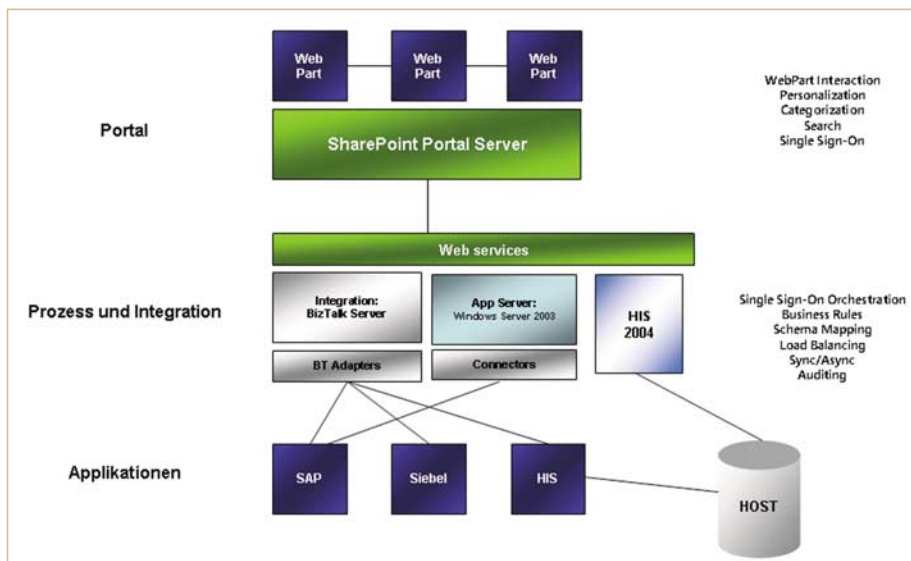


Abb. 3: Architekturvorschlag für die Einbindungen von Applikationen in SharePoint-Portale

Ausgabe zur Verfügung. Man kann hiermit ohne Programmieraufwand auf Daten eines SAP R/3 4.x oder R/3 6.20 (Enterprise) Systems zugreifen und diese in beliebiger Form darstellen oder weiterverwenden.

Will man einfache Abfragen und Darstellung von Ergebnissen, ist man wahrscheinlich mit dem .NET Connector und somit dem BAPI-Ansatz besser bedient. iDoc hingegen zeigt Vorteile beim Starten bestimmter Prozesse, deren gesamte Informationen man in ein Dokument packen kann und dann losschickt. Man sollte sich daher zunächst über Datenmenge und Aufgabenstellung im Klaren sein und dann entscheiden, welche Mittel man einsetzt.

Ein weiterer Kandidat ist der Host Integration Server oder HIS, der in der Version 2004 bereits Schnittstellen für Web Services anbietet und somit die Integration von Hosts in das Gesamtszenario erleichtert. In

Abbildung 3 ist ein Überblick einer möglichen Architektur gegeben und Sie sehen, welche wichtige Rolle Web Services als Mittler zwischen den Welten spielen.

Single Sign On

Bei der Integration bestehender Anwendung trifft man sehr schnell auf das Problem der Benutzerverwaltung. Meist bringen Enterprise Applikationen eine eigene Anwenderverwaltung mit Rollen- oder Gruppenkonzept mit. Für einen Benutzer ist es sehr mühsam, beim Starten eines Portalfensters zunächst einmal alle eingeblendeten Applikationen mit dem jeweiligen Benutzernamen und Passwort zu versehen. Erzwingt man dies, geht sehr viel des ursprünglichen Vorteils des Portals verloren.

SharePoint stellt daher einen zentralen Dienst zur Verfügung, den zum Beispiel BizTalk Server nutzt, um applikationsspezifisch den Benutzernamen und weitere Information zur Verfügung zu stellen. Bei diesem Single Sign On (SSO) handelt es sich mehr um einen gesicherten Speicherbereich als um ein SSO im eigentlichen Sinne, wie es zum Beispiel von Active Directory mit Hilfe von Kerberos zur Verfügung gestellt wird.

Der Administrator legt im SharePoint Portal Server eine Applikation an und definiert dabei, welche Information ein WebPart bräuchte, um mit dieser Applikation im Rechtekontext des Benutzers interagieren zu dürfen. Im einfachsten Fall sind dies Benutzername und Passwort, bei manchen

Systemen wird aber auch noch Workstation-ID oder Ähnliches gebraucht.

Ein WebPart kann nun während seiner Darstellungsphase an SharePoint Portal Server die Frage nach den Daten des aktuellen Benutzers für die entsprechende Applikation stellen. Besitzt SharePoint bereits diese Daten, werden sie zur Verfügung gestellt, falls nicht, wird der Benutzer nach diesen Daten gefragt und die Informationen in einem verschlüsselten Speicher für die Zukunft gehalten.

Es gibt mittlerweile eine Anzahl von WebParts, die sich dieses Speichers bedienen, um unterschiedliche Funktionalitäten abzubilden. Die einfachste Funktion ist ein Web Capturing mit Basic-Authentifizierung.

Die Methodik spaltet die Welt in grob zwei Lager: Für die einen ist dies endlich ein Mechanismus, um der Flut von verschiedenen Passwörtern Herr zu werden, für die anderen wird das Sicherheitskonzept vieler Applikationen damit unterlaufen. Es ist auf jeden Fall eine mögliche Option und man muss selbst entscheiden, wie und wo man sie einsetzen möchte.

Dieser Dienst ist übrigens auf SharePoint Portal Server 2003 beschränkt und wird in Windows SharePoint Services nicht angeboten.

Fazit

SharePoint ist eine offene Plattform und bietet viele Möglichkeiten, bestehende Applikationen einzubinden. Welche der vier vorgeschlagenen, grundlegenden Möglichkeiten man wählt, ist entsprechend dem Szenario abzuwägen, das sich aus der bestehenden Applikation heraus ergibt.

Dieser Artikel sollte hierzu Ansätze und Gedankengänge liefern. Meist wird man eine Mischung aus einigen der vier Ansätze wählen. Es ist jedoch wichtig, sich von vorneherein über die genannten Vor- und Nachteile im Klaren zu sein und entsprechende Maßnahmen vorzusehen. ●

● Links & Literatur

- [1] WebPart-Verzeichnis: www.microsoft.com/sharepoint/downloads/components/
- [2] WSRP Standard Homepage: www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp
- [3] Download der iView- und WSRP-WebParts: www.gotdotnet.com/team/sharepoint/
- [4] Homepage zu BizTalk Server: www.microsoft.com/biztalk/

Application Integration Server

Vorteile

- Standard-Vorgehen bei Integration von Applikationen (fertige Connectoren etc.)
- Integrationsserver übernimmt Zugriff
- Mehrwerte wie Transaktionshandling, Datentransformation

Nachteile

- Komplexe Infrastruktur muss erstellt und gepflegt werden.
- Meist nicht optimal für große Datenmengen