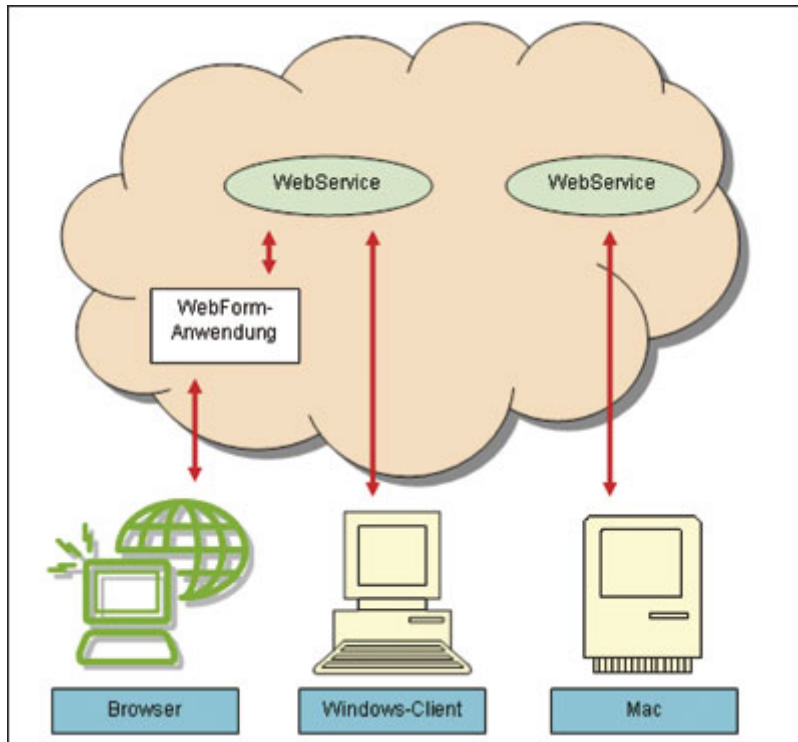


Service Oriented Architecture einfach nutzen

RPG in der modernen Software-Architektur

Die technologische Entwicklung hat uns ein neues Konzept beschert: SOA – Service Oriented Architecture. Dabei handelt es sich um Technologien rund um Web-Services. Ein Web-Service ist ein Dienst, der Programmen die Nutzung von Ressourcen im Web ermöglicht. Allgemein könnte man sagen, dass ein Web-Service für Computer dasselbe ist wie eine Web-Page für Menschen.

Der Computer (das Programm) orientiert sich an der Dienstbeschreibung wie der Mensch an der Darstellung einer Homepage und nutzt die Angebote (Funktionen, die man Web-Methoden nennt) wie wir Menschen, wenn wir eine Bestellung machen oder uns Daten herunterladen. Die Richtlinien für die Technologie werden vom W3C festgelegt, um sicherzustellen, dass SOA plattformneutral bleibt.



Durch Standardisierung können Webservices von jedem Cluster verwendet werden, unabhängig von Betriebssystem und verwendeter Sprache.

Der Web-Service kapselt Logik; dem Nutzer ist unbekannt, was hinter dem Web-Service steckt. Ob es sich um einen simplen Datenzugriff oder eine komplexe Anwendung handelt, ist nach außen hin nicht sichtbar.

Die Web-Methode kann Parameter übernehmen und gibt dem aufrufenden Programm eine Klasse zurück. Die Informationen, die vom Programm gebraucht werden, um mit der Klasse umgehen zu können, sind in der Dienstbeschreibung (WSDC) hinterlegt.

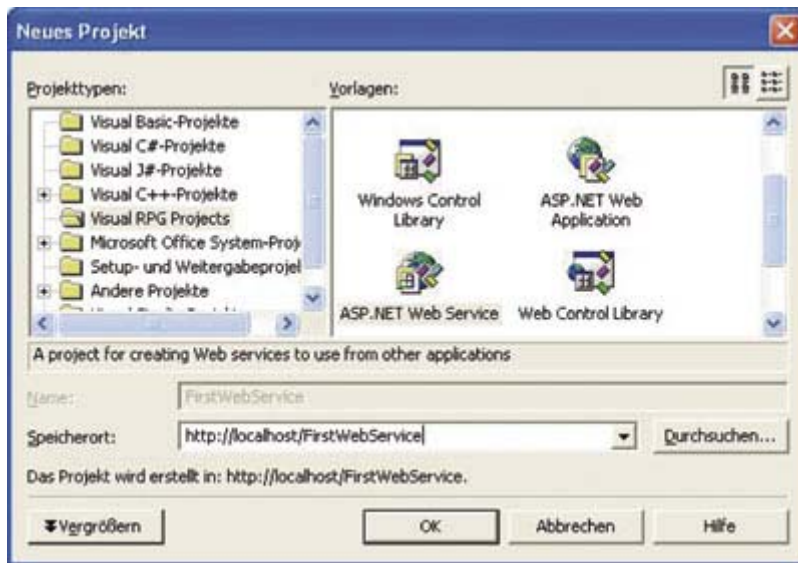
Das hört sich nach jeder Menge Verwaltung und Technologie an. Für einen DotNet-Programmierer stellt sich der Web-Service letztendlich als Batch-Programm dar. Das Handling der Dienstbeschreibung wird vollständig von VisualStudio übernommen.

Die Aufgabe des Programmiers, um einen Web-Service anzubieten, ist vor allem die Beschreibung der Klasse sowie die Programmierung der Web-Methode.

Um einen Web-Service zu nutzen, genügt es, eine Web-Referenz zum Projekt hinzuzufügen und die Methoden zu verwenden. Die Verwaltung wird komplett von VisualStudio übernommen.

Erstellen des Web-Service

Wir starten ein neues Projekt vom Typ RPG.NET WebService mit dem Namen FirstWebService in VisualStudio2003. Über die Projekt-Templates wird von VisualRPG.NET der Programmrahmen für den Web-Service generiert. Wir deklarieren die Klasse, die wir zurückgeben wollen, und schreiben den Code, um die Daten abzurufen.



Anlage des Projekts in Visual-Studio

Die Web-Methode bekommt einen Parameter mit der Kundennummer übergeben und gibt dafür eine Klasse (=Adressdaten) zurück.

Schauen wir uns den RPG-Code an, dann finden wir ein CHAIN, mit dem der Kundenstammsatz eingelesen wird, sowie die Klasse „KundenDetails“, die befüllt werden muss. Bei EOF wird ein leeres Objekt zurückgegeben.

```
// Einlesen der Kundendetails

BegFunc LeseKundenDetails Type(KundenDetails) Access(*Public)
Attributes(WebMethod())

    DclSrParm        KundenNr        Type(*integer4)

    objKNDDetail = *New KundenDetails() //
instanzieren des Objekts

    CHAIN KundenStamm Key(KundenNr) Err(*extended)

    If %found()

        // Attribute des Objekts füllen

        With objKNDDetail

            .Nummer        = CMCustno

            .Name           = %Trim( CMName)

            .Adresse        = %Trim( CMAddr1 )

            .Ort            = %Trim( CMCity)

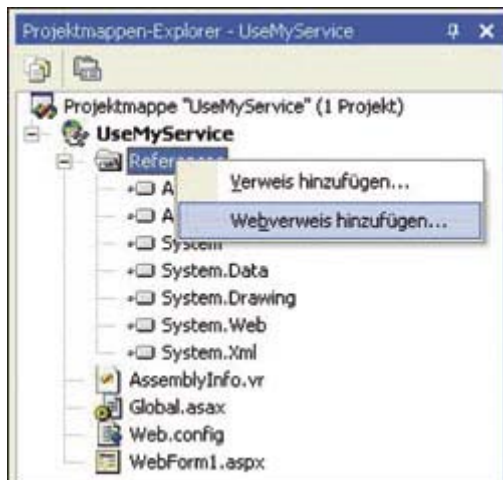
        EndWith

    Endif

    LeaveSr objKNDDetail

EndFunc
```

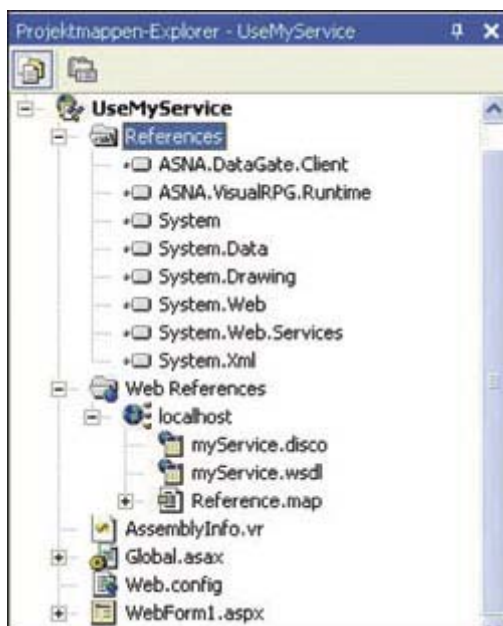
Zum Test rufen wir die URL aus dem Browser auf. Wir übergeben die von der Web-Methode geforderte Kundennummer und bekommen ein XML zurück.



Webservice aus einem Client-Projekt heraus referenzieren

Nutzen eines Web-Service

Genauso einfach wie er erstellt wird, ist der Service auch zu nutzen. Wir legen ein Projekt an und fügen eine Web-Referenz hinzu. Bei diesem Schritt sucht VisualStudio nach verfügbaren Web-Services und stellt die enthaltenen Methoden und Klassen automatisch zur Verfügung.



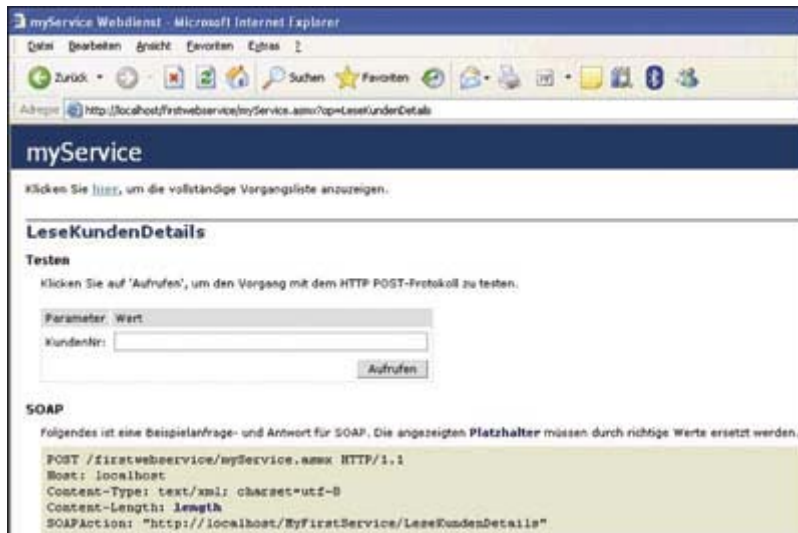
Die Webreferenz im Client-Projekt

In dieser Web-Form wird der Web-Service genutzt. Sie sehen das Ergebnis nach dem Aufruf des Kunden über die Kundennummer.

Der Nutzen

Aus dieser kurzen Beschreibung geht hervor, dass Unternehmen mit SOA eine gute Möglichkeit haben, ihre IT für die Außenwelt zu öffnen.

Derzeit geht es hauptsächlich darum, mit anderen Daten auszutauschen oder Funktionen anzubieten, die auf eigenen Daten oder Programmen basieren. Anders als beim Daten-Upload können im Web-Service einfach Prüfungen gemacht und Resultate zurückgegeben werden.



Dialog zum Test eines Webservices, hier wird die Kundennummer eingegeben, das Ergebnis sehen Sie unten

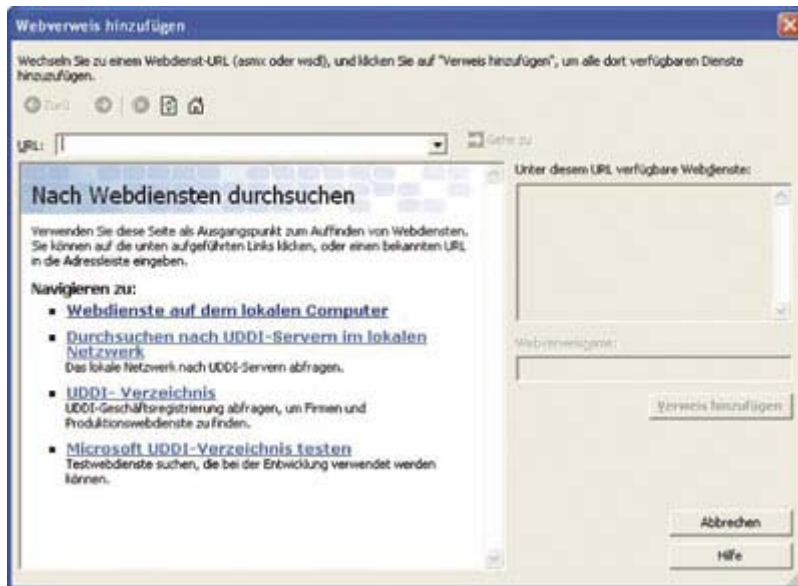
Von dieser praktischen Anwendung abgesehen hat das Konzept der standardisierten Schnittstelle, die in XML beschrieben wird, sicher längerfristige Bedeutung. Ein weiterer wichtiger Punkt ist, dass die Logik am Server liegt und ideal gekapselt ist. Die Logik selbst ist einfach – ähnlich einem Batch-Programm zu erstellen.



Das Ergebnis des Testaufrufs. Die Klasse als XML

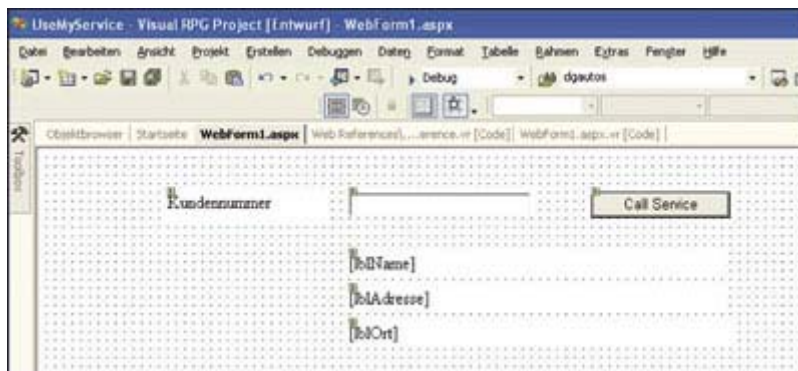
Die Zukunft

Microsoft als führendes Mitglied im W3C hat sich stark für Web-Services engagiert. Zudem kommt nächstes Jahr das neue Betriebssystem VISTA (=Durchblick) auf den Markt, das dienstbasierend sein soll. Mit INDIGO wurde für 2006 auch eine neue Plattform für Anwendungskommunikation angekündigt.



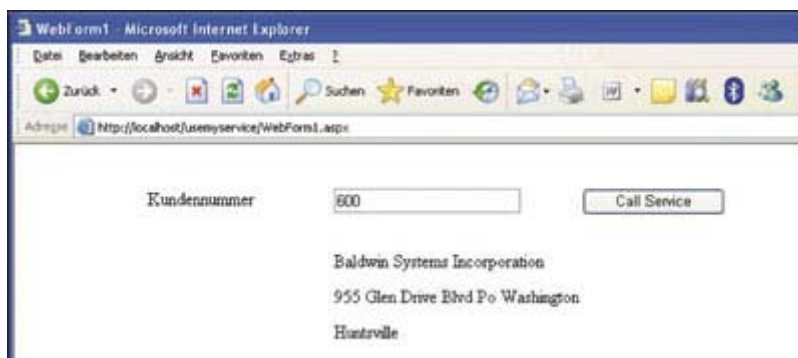
Übersicht über alle verfügbaren Webservices, lokal wie auch im Internet

Wir können davon ausgehen, dass sich SOA als Anwendungsmodell neben dem Web oder Client/Server durchsetzen wird. Ebenso sicher erscheint, dass sich im Client-Bereich sehr viel bewegen wird. Mit InfoPath stellt Microsoft bereits heute ein Standard-Tool zur Verfügung, das in der Lage ist, Client-Programme für Datenbankdialog oder Web-Services per Drag&Drop zu erstellen. Diese Entwicklung hat gerade erst Ihren Anfang genommen und wird früher oder später die derzeit aktuellen Client-Konzepte in Frage stellen.



Die Entwurfsmaske des Webprogramms als Nutzer des Webservices

Fazit: Es macht durchaus Sinn, Web-Services bei der Konzeption von Anwendungen mit einzubeziehen.



Das Ergebnis

