



## Midrange MAGAZIN

Ausgabe April 2005

Alternative Integrationsplattform

### Was ist .Net

**Im Jahr 2001 hat Microsoft die .Net-Technologie der Öffentlichkeit vorgestellt. .Net ist eine betriebssystem- und sprachunabhängige Entwicklungsplattform. Vom Prinzip her ist .Net ähnlich wie Java konzipiert; ein wesentlicher Unterschied zu Java ist, dass man sich aussuchen kann, in welcher Sprache man seine Anwendungen entwickelt.**

Es scheint, dass Microsoft den Ansatz von Java übernommen und weitergedacht hat. Investiert wurden unglaubliche 4.000 Mannjahre. Bill Gates persönlich ist im Herbst 2001 zur Präsentation von .Net um die Welt getourt. Spätestens hier wird klar, was das Ziel der Anstrengungen war.

.Net ist Microsofts Investition zum Eintritt in das Business-Computing. Es hat bei Microsoft noch keine fehlerfreie und stabilere Anwendung als .Net gegeben.

Zum Schrecken der Windows-Entwickler war .Net wirklich neu und keine Neuverpackung alter Technologien. Microsoft hat mit .Net einen radikalen Schnitt gemacht. Das erklärt auch die zurückhaltende Reaktion der Entwickler, die vor die Tatsache gestellt wurden, sich wieder einmal – und zwar massiv – mit neuen Technologien auseinandersetzen zu müssen. Durch das .Net-Framework werden folgende Ziele erreicht:

- Vereinheitlichung des Programmiermodells
- Vereinfachung der Entwicklung
- Hohe Programmstabilität
- Unterstützt viele Sprachen
- Unterstützt XML-Web-Services
- Unterstützt alle Plattformen

#### **.Net als Integrationsplattform**

Der Compiler, in dem das Anwendungsprogramm geschrieben ist, setzt die Befehle der Sprache in die so genannte Intermediate Language (IL) um. Dieser Code wird vom Just In Time Compiler (JIT) interpretiert. Die Laufzeitumgebung Common Language Runtime (CLR) ist auf die Plattform abgestimmt.

Dieses Framework ist somit nach oben (für alle Anwendungssprachen) und nach unten (für alle Plattformen) offen. Bei der Präsentation von .Net lagen Absichtserklärungen von vielen Compiler-Herstellern vor, ihre Produkte auf .Net zu portieren. Heute gibt es für jede bekannte Sprache ein .Net-Äquivalent.

Microsoft hat auch niemals Zweifel über die Absichten mit .Net zugelassen. Es gab, auch als der Markt nur zögerlich bereit war, .Net anzunehmen, keine Alternativ- oder Ausweichstrategie. Das Ziel war, die Entwickler auf einer Plattform zu vereinen, ohne Ihnen eine Sprache aufzuzwingen. Abgesichert wurden diese Investitionen durch die Möglichkeit Anwendungen für eine unglaubliche Breite von Geräten – vom Mobiltelefon bis zum Server – in einer Umgebung entwickeln zu können.

### CTS – CommonTypeSystem

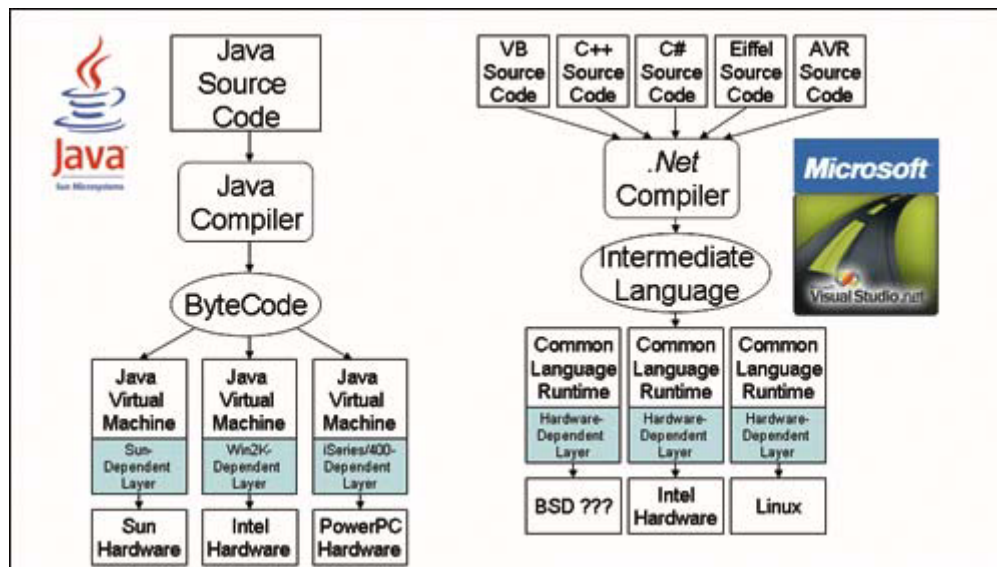
Damit die Objekte aus verschiedenen Sprachen auch zusammenarbeiten können, muss gewährleistet sein, dass die gleichen Datentypen verwendet werden. Diese Datentypen werden in der CLR (Common Language Runtime) definiert. Im CTS sind die Basis-Datentypen definiert. Darauf aufbauend kann jede Sprache seine eigenen Datentypen führen. Visual RPG verwendet z.B. den Datentyp \*packed für die gepackte Darstellung numerischer Werte.

### Datenzugriff mit ADO.Net

Natürlich wurde mit .Net auch ein neues Konzept vorgestellt, mit Daten umzugehen. Die bisher bekannten ADO (ActiveX Data Objects) haben mit ADO.Net nur den Namen gemeinsam.

Daten werden in ADO.Net in einem Data Set gehalten. Das ist eine XML-Datei, in der eine Datenbank mit Tabellen und Relationen abgebildet ist. Das Data Set enthält das Result Set einer SQL-Abfrage, die von einem „Managed Provider“ zurückgegeben wird.

Der Managed Provider stellt als .Net-Komponente die Schnittstelle zur Datenbank dar. Der Vorteil dieses Konzeptes gegen eine globale Schnittstelle wie ODBC ist, dass sich ein Managed Provider der spezifischen APIs der Datenbank bedienen kann und keine Rücksicht auf Standards nehmen muss.



Architektur Java vs. .Net

## **Web-Entwicklung mit ASP.Net**

Ist grundlegend anders als mit dem Vorgänger ASP. Es wird ein objektorientiertes Programmiermodell verwendet. Sowohl die Web-Seite als auch Ihre Elemente sind Objekte, die über Namen angesprochen werden und auf deren Eigenschaften und Methoden aus kompilierten Sprachen zugegriffen werden kann. Ein weiterer wesentlicher Unterschied sind die Erweiterungen in der Zustandsverwaltung der Web-Anwendung gegenüber ASP.

## **Web-Services mit .Net**

Werden von Microsoft als Kernpunkte von .Net betrachtet. Hier geht es darum, Dienste, die im Internet angeboten werden, zu nutzen oder selbst anzubieten. Ein Web-Service ist eine Anwendung, die in ein .Net-Projekt eingebunden werden kann wie jede andere Komponente auch. Ein gutes Beispiel sind Flugbuchungen; ein sehr bekannter Web-Service ist der von Amazon.

Die Web-Services werden in Zukunft im B2B-Bereich weiter an Bedeutung zunehmen.

Für einen .Net-Programmierer macht es keinen Unterschied, ob er eine Funktion aus einem Verweis auf eine lokal vorliegenden Bibliothek (DLL) oder einem Web-Service in sein Programm integriert. Die Kommunikation läuft über das SOAP-Protokoll wobei die Codierung und die Decodierung von .Net automatisch ausgeführt wird.

## **Mobile Einheiten und .Net**

Mit .Net CF steht ein Compact Framework zur Verfügung, mit dem Anwendungen für mobile Geräte erstellt werden können. Dieses Framework verfügt über dieselben Datentypen und weitgehend dieselben Möglichkeiten wie das Desktop-Framework. Es können Windows Forms-Programme erstellt werden, die nur wenigen Einschränkungen unterliegen.

## **Kosten**

Das .Net-Framework SDK (Software Development Kit) kostet gleichviel wie Java – nämlich nichts. Mit dem SDK bekommt man auch Visual Basic geliefert und kann mittels Notepad Anwendungen erstellen.

Allerdings kenne ich niemanden, der das im professionellen Umfeld so macht. Alle verwenden die IDE (Integrated Development Environment) – eine Entwicklungsumgebung, mit der man alle Werkzeuge bekommt um .Net-Anwendungen auf rationellem Weg zu erstellen und zu verteilen. In der integrierten Hilfe, der MSDN (Microsoft Developer Network) finden sich Beispiele und auch komplette kleine Anwendungen, die wichtige Hilfestellung bei der Entwicklung geben.

Im Gegensatz zu Java bekommt man hier alles aus einer Hand.

## **Blick in die Zukunft**

Dieses Jahr soll Version 2 von .Net kommen; mit dieser Version kommt auch ein neuer SQL-Server, der eine hochinteressante Eigenschaft mitbringen soll. Bis heute waren Trigger-Programme am SQL-Server nur als TSQL (Transact SQL) möglich. Mit der neuen Version sollen auch Managed-Stored Procedures – also Trigger geschrieben in .Net – möglich sein. Hier entstehen Eigenschaften, die uns als i5-Anwender lange Zeit bekannt sind.

## **Fazit**

Microsoft hat mit .Net eine zeitgemäße und flexible Umgebung geschaffen, in der langsam aber sicher eine Vielzahl von Anwendungen entstehen. Der Ansatz von .Net als Integrationsplattform und die Entschlossenheit von Microsoft, .Net am Markt durchzusetzen, lässt weniger die Frage aufkommen, ob man .Net verwenden soll als ob man sich langfristig dagegen wehren kann.

**Fachautor: Christian Neißl**