Vorlesung
Software aus Komponenten

3. Komponenten-Modelle

apl. Prof. Dr. Hans-Gert Gräbe
Wintersemester 2007/08

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Server-Provider (Server-Anbieter)

- stellt Plattform zur Verfügung
- Netzwerkanbindung, Skalierungsfunktion
- Prozess- und Ressourcenmanagement

EJB-Kontext-Provider (Container-Anbieter)

- setzt auf Plattform des Server-Providers auf
- Benutzerverwaltung, Transaktionsmanagement, Persistenz
- Herstelllung der Installationswerkzeuge
- Implementierung der EJB-Kontextfunktionalität
- Container- und Server-Provider oft identisch

  bessere Performance und Wartbarkeit

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Bean-Provider (Komponenten-Entwickler)

- realisiert geforderte Anwendungslogik (Geschäftslogik)
- keine grundlegenden Funktionalitäten (Persistenz, Netzwerk, etc.)
- benötigt Fachwissen über Anwendungsbereich
- Konzentration auf inhaltliche Problemstellung

Application-Assembler (Monteur)

- verbindet Komponenten zu einer Anwendung
- Clients
- Verwendung von Basiskomponenten
- Nutzung wiederverwendbarer Komponenten von Drittanbietern
- oftmals Bean-Provider und Application-Assembler in einem Haus

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Bean-Deployer (Installation)

- installiert Komponenten der Anwendung auf Zielsystem
- verwendet Werkzeuge des Container-Providers
- nutzt Deployment-Deskriptor
- konfiguriert EJBs
- generiert Stummel- und Skelettklassen
- legt Benutzerdatenbank an
- benötigt detailliertes Wissen über Server und Container

Systemadministrator

- verantwortlich für reibungslosen Ablauf
- Benutzerverwaltung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Java-Basisdienste für Komponenten

Grundlegende Dienste

Java core reflection erlaubt zur Laufzeit

- Inspektion von Klassen und Schnittstellen nach Attributen und Methoden
- Konstruktion neuer Klasseninstanzen und Felder
- Zugriff und Modifikation von Attributen in Verbundobjekten oder Feldern
- Aufruf von Methoden von Objekten und Klassen

- **java.lang.reflect** als eigene Klasse hierfür
  - einige Funktionalität historisch in der (finalen) Klasse **java.lang**.

Java GuI-Klassensammlungen AWT und Swing

- delegierende Ereignisbehandlung
- Datentransfer und Zwischenablage wird unterstützt, drag and drop
- Java 2D rendering, eng damit zusammen Java printing model
- Internationalisierung
- pluggable look and feel, Palette von Standard-Komponenten

Objektserialisierung

- standardisiertes Kodierungsschema für Serialisierung
- Klasse muss dafür Interface **java.io.Serializable** implementieren
- Objektserialisierung ist sicherheitskritisch
- Mechanismen zu Serialisierung und Deserialisierung ganzer Objekt-Webs
  - nicht zu serialisierende Attribute können als transient markiert werden
    - Bsp: große Cache-Strukturen
  - private Methoden **readObject** und **writeObject** werden statt Default genommen, wenn durch Reflektion gefunden
  - Mehrfachreferenzen auf ein Objekt werden rekonstruiert
- unterstützt einfaches Versionierungsschema:
  - 64-bit-hash-Code (Serial version UID = SUID) wird über die Signatur der Klasse berechnet und kann während **readObject** ausgewertet werden.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Ferne Objekte und RMI

- Auf ferne Objekte kann nie direkt zugegriffen werden, sondern nur über Interface **java.rmi.Remote**. Ressourcenbindung über Namensdienst.
- Remotezugriff kann immer fehlschlagen:
  Exception **java.rmi.RemoteException**
- Parameterübergabe:
  - Referenz, wenn Parameterwert selbst vom Remote-Typ ist
  - Durch Marshalling übergebene Kopie, wenn Parameterwert lokal im aufrufenden Kontext
  - Übergabe nicht serialisierbarer Objekte führt zu Laufzeitausnahme
- Unterstützt verteiltes Garbage Collection
  - durch genaue Buchführung über Remote-Referenzen
  - basiert auf Arbeit [Birrel 1993] über Network Objects
  - eines der bedeutendsten Features von Java RMI
- Konflikt mit Begriff der Objektidentität im Java-Standard
  - Referenzen auf ein fernes Objekt sind Java Referenzen auf das lokale Proxy des fernen Objekts
  - Backcall erzeugt ein lokales Proxy im ObjectHome, neben der eigentlichen Java-Referenz

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Weitere Java-Dienste für Komponenten

JNDI: Java Naming and Directory Service

Aufgabe: Dienste über Namen bzw. Attribute finden

- Interface **Context** macht Namenskontext verfügbar
- Methode **lookup** findet Objekte über ihren Namen
- Interface **DirContext** erweitert **Context** zur Suche über Attributwerte
- Unterstützung von Kontexthierarchien, die rekursiv durchsucht werden.

JMS: Java Messaging Service

Aufgabe: Unterstützung asynchroner datengetriebener Kompositionsmodelle

- Standardisiert Java-Zugriff auf vorhandenes Nachrichtensystem, implementiert keins selbst.

JDBC: Java database connectivity

Aufgabe: Einheitlicher Zugriff auf Datenbanken über entsprechende Treiber

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

JTA: Java Transaction API

JTS: Java Transaction Service

Aufgabe: Unterstützung von Transaktionskonzepten

JCA: J2EE Connector Architecture (seit J2EE 1.3)

- Einheitliches Konzept des Ressourcen-Adapters, über welches externe Ressourcen aus einer EIS (enterprise information structure) in einen J2EE-Applikationsserver eingebunden werden können
- Definition eines entsprechenden **JCA common client interface** (CCI)
- Einsatz innerhalb von Enterprise Application Integration Frameworks

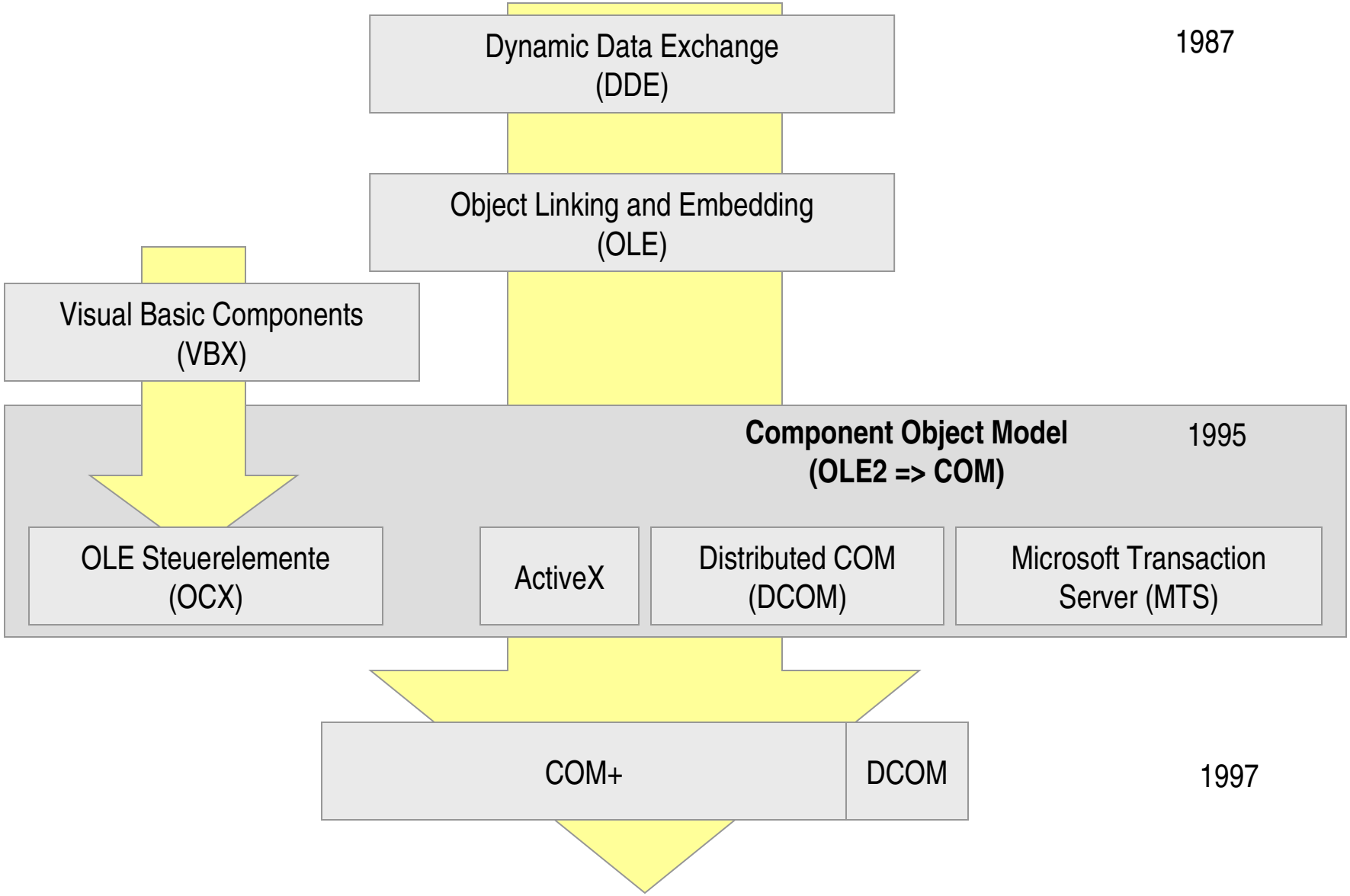
Java und XML: Java unterstützt mit entsprechenden Klassen

- XML-Dokumente (DOM)
- XML-Streaming (SAX)
- XML-Binding (JAXP)
- XML-Messaging (JAXM)
- XML-Processing (JAXP)
- XML-Registries (JAXR)

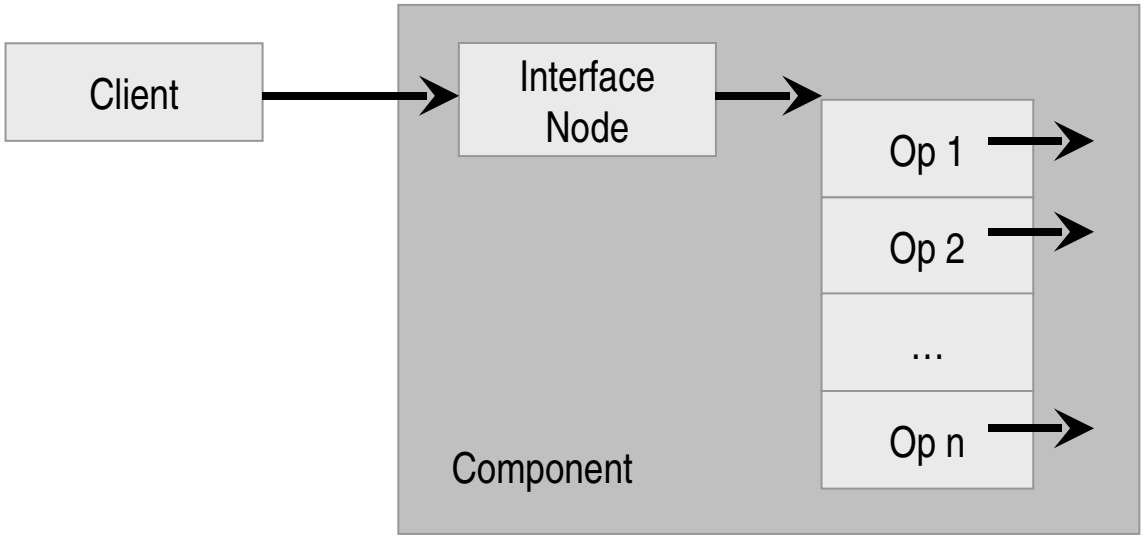
Java und CORBA:

- Java wichtigste CORBA-Referenzimplementierung
- Koexistenz in fast allen Applikationsserver-Produkten
- Zugriff auf CORBAservices über Java-spezifische Zugriffs-Schnittstellen sowie weitere Konzepte (POA, Namensdienst) seit Java 1.4
- RMI-over-IIOP als eingeschränkte RMI-Version seit 1999
  - RMI nutzt spezifisches proprietäres Protokoll
  - keine Unterstützung des verteilten Garbage Collection, so dass explizites Lebenszyklus-Management erforderlich ist
    - dafür existieren aber CORBAservices

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

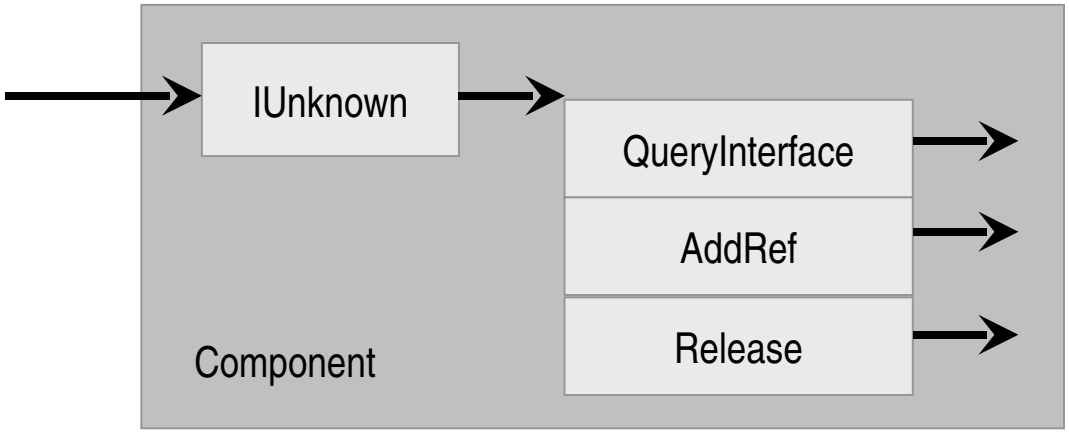


UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme



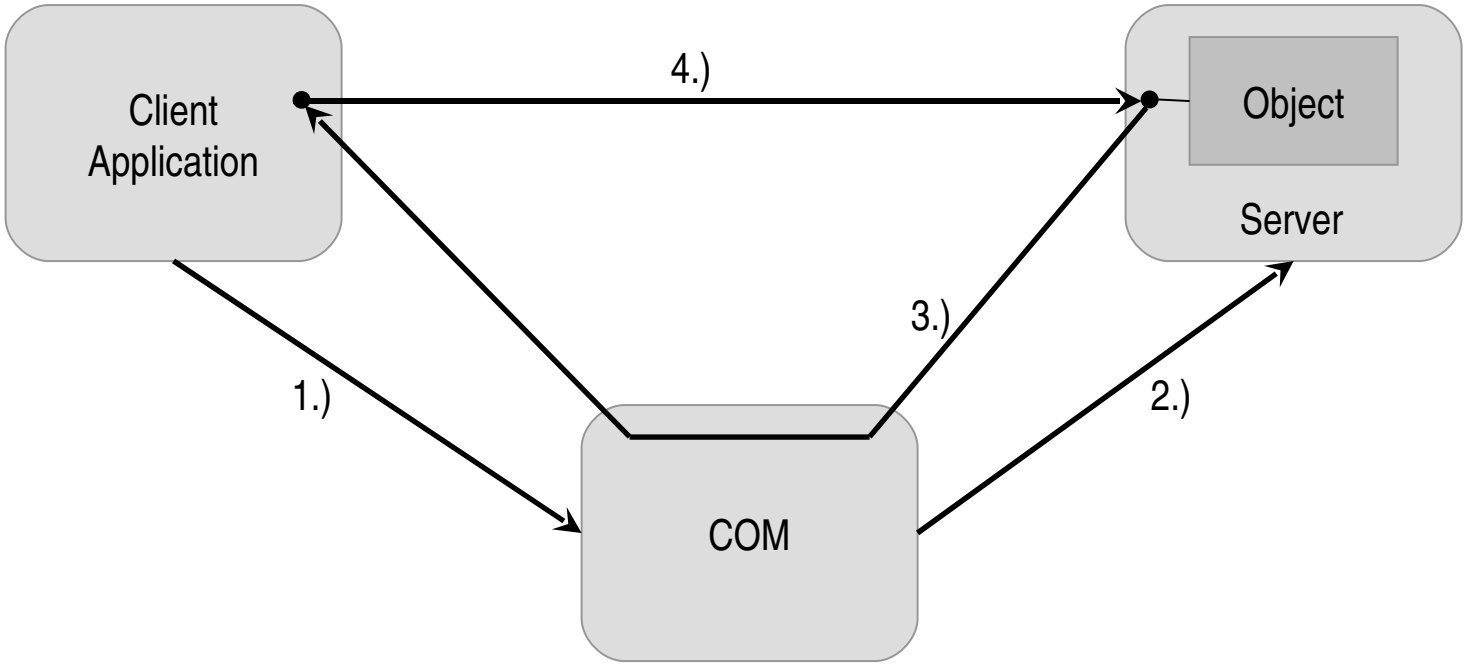
- COM ist binärer Standard
- Schnittstelle ist zentraler Punkt von COM
- Schnittstellenvererbung
- Interface Node
  - Methoden eines Objekts => **"this"**-Parameter wird an jede Methode weitergereicht
  - Komponente kann mehrere Schnittstellen implementieren

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme



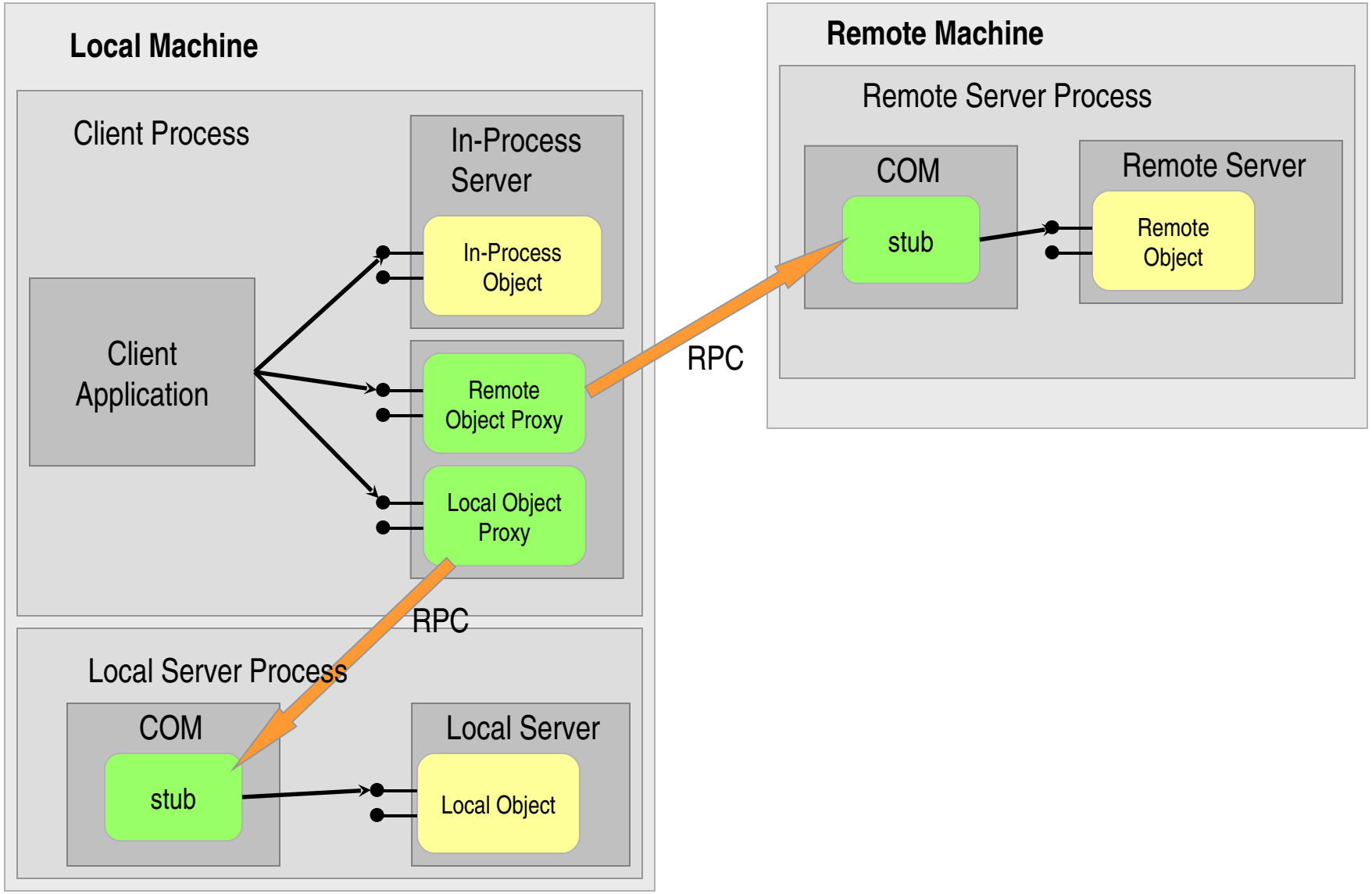
- Schnittstelle IUnknown **muss** von jeder Komponente implementiert werden
  - jede andere Schnittstelle erbt von IUnknown
- QueryInterface
  - erste Methode jedes COM-Objektes
  - gibt Zeiger auf angefordertes Interface zurück
  - benutzt Interface Identifier (IID)
  - erlaubt (dynamische) Introspektion der Schnittstelle
- AddRef, Release
  - Verwaltung des Lebenszyklus
  - Referenzzähler

Erstellen eines COM-Objektes



1. Aufruf der Funktion "CreateObject"
2. COM lokalisiert/erzeugt den Server
3. Serverprozess erzeugt das Objekt und gibt einen Schnittstellenzeiger zurück
4. Client kommuniziert über den Schnittstellenzeiger mit dem Serverobjekt

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme



- Typinformationen (Type Information)
  - Laufzeitzugriff auf Typinformationen des COM Objektes
  - wird vom Microsoft IDL Compiler generiert und in einer Typbibliothek gespeichert
  - COM-Schnittstellen zum Interagieren mit dieser Bibliothek

- Strukturiertes Speichermodell (Structured Storage and Persistence)
  - von COM unterstützte Methode zum Speichern von Daten
  - Speicherung erfolgt analog des Dateisystems in einer Datei

- Moniker
  - Dienst, welcher ein einzelnes Objekt in einem genau definierten Zustand erzeugen und initialisieren kann
  - für Clients, die mit exakt dem gleichen Objekt weiterarbeiten müssen
  - Moniker kann an gesamtes Objekt oder an einen Teil gebunden werden

COM Dienste

- Einheitlicher Datenaustausch (Uniform Data Transfer)
  - Datentransfer zwischen COM-Objekten
  - Benachrichtigung von Datenänderungen einer Quelle (Datenobjekt) und einem Datenkonsumenten
- Verbindbare Objekte (Connectable Objects)
  - zur Ereignisverarbeitung
  - Objekt definiert ein Interface, welches für das Ereignis genutzt werden soll
  - Client implementiert dieses Interface
- COM+ Ereignisdienst (Event Service)
  - Asynchrone Kommunikation zwischen Komponenten
  - Empfänger abonniert Ereignis beim Dienst
  - Sender schickt Ereignis zum Dienst, ohne den/die Empfäner zu kennen
- COM+ Nachrichtenwarteschlange (Message Queuing)
  - Garantierte Auslieferung auch bei Nichterreichbarkeit des Empfängers
  - Für Systeme, deren Verfügbarkeit nicht immer gewährleistet ist

Was ist .NET?

"... komplette Neudefinition der Art, wie Microsoft in Zukunft Geschäfte machen will ... und wie Software entwickelt werden soll."

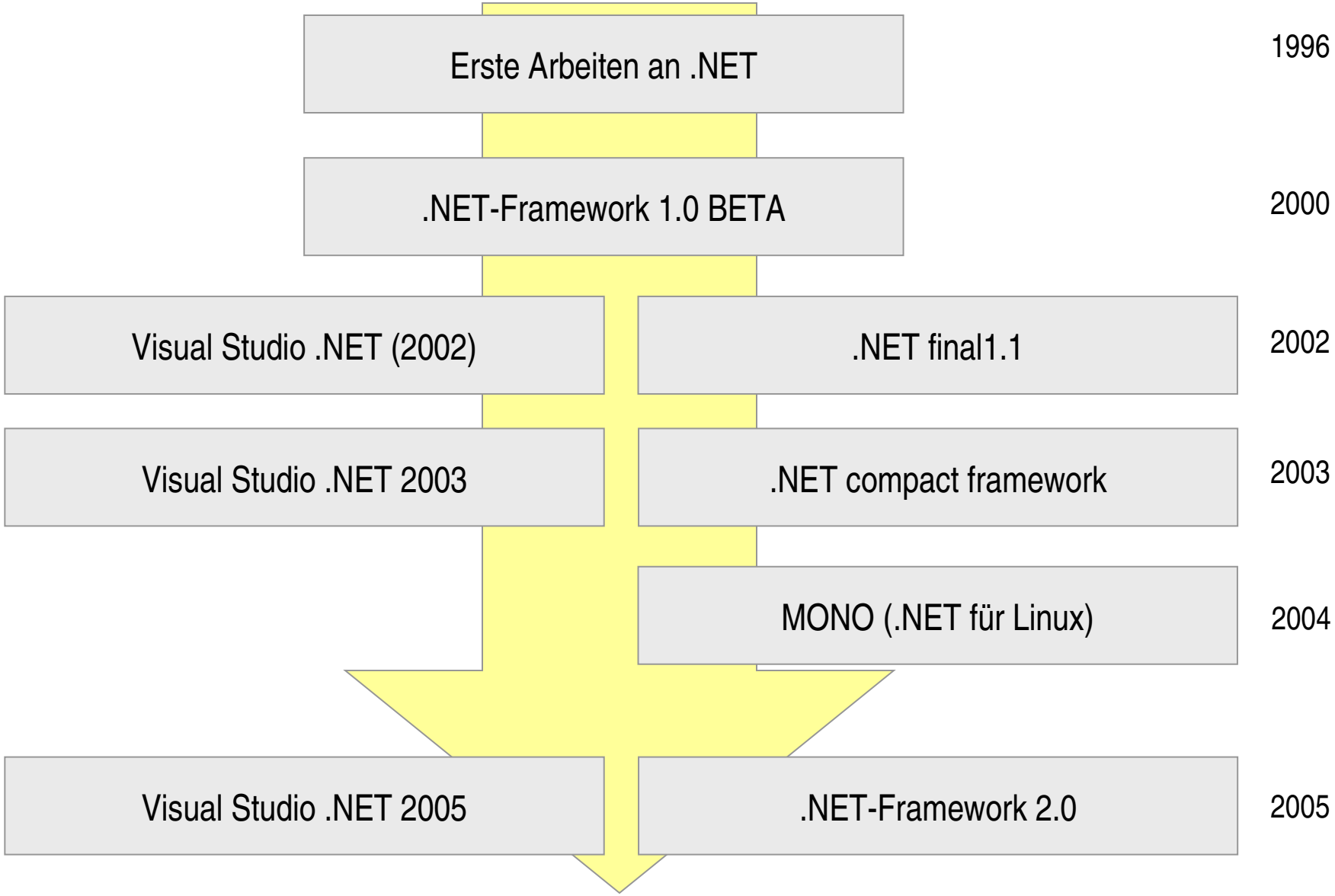
Westphal, 2002

- Plattform soll bisherige Vorgehensweisen der Windows-Programmierung sowie die veraltete COM-Technologie ersetzen, flexibel auf Betriebssystem- und Basisfunktionen zugreifen und Austausch zwischen Programmen unterstützen.
- Ausgerichtet auf den Einsatz auf verschiedenen Hardware-Plattformen bis hin zu Handys und PDAs
- Ziele
  - Sicherheit
  - Plattformunabhängigkeit
  - Interoperabilität
  - Homogenität

Vorgeschichte:

- Rechtsstreit zwischen Sun und Microsoft um Java
  - Microsoft erweitert Java nach eigenen Vorstellungen und Bedürfnissen und gefährdet damit die Java-Kompatibilität
  - Microsoft-Implementierungen J++ und J#
- Weitere Probleme:
  - Auch die für Windowsprogrammierung meist verwendeten Sprachen Visual Basic, C++ und J++ waren nicht kompatibel
  - String-Datentypen sogar nicht binär kompatibel
  - kein einheitliches Modell der Speicherverwaltung

Geschichtliche Einordnung



- August 2000 – C# und die CLI werden von MS, HP und Intel zur Standardisierung bei der ECMA eingereicht
  - ECMA – European Computer Manufacturers Association
- Dezember 2001 – Fertigstellung des ersten Standards und Weitergabe an die ISO
- April 2003 – Verabschiedung des ISO-Standards
  - ISO/IEC 23270 (C#)
  - ISO/IEC 23271 (CLI)
- Oktober 2003 – Standardisierung der Bindung von C++ an die CLI beginnt
  - ECMA-372 (Dezember 2005)
- 2004: Marktanteil steht noch immer in keinem Verhältnis zur Aufmerksamkeit, die .NET in den Medien findet
- Ende 2005: Visual Studio 2005, .NET Framework 2.0
- Ende 2006: .NET Framework 3.0
- Ende 2007. Visual Studio 2008 und .NET Framework 3.5

- Entscheidung für ein **laufzeitbasiertes System**, das nicht direkt aus der Hochsprache in den Maschinencode des Zielsystems kompiliert.
- Aufgreifen der Erfahrungen von Java sowie den Skriptsprachen, dass sich mit zunehmender Komplexität von Software **der Schwerpunkt von Performanz der Programme zur Wartbarkeit verschiebt**.
- Programmcode kann **zur Laufzeit** mittels Reflection über ein Objektmodell generiert und direkt im Speicher in lauffähigen Code übersetzt werden.
- Zusammenarbeit von Code in der Plattform (managed code) und außerhalb der Plattform (unmanaged code) ist möglich.

Aktuelle Implementierungen

- Microsoft (Windows) http://msdn.microsoft.com/net
- MONO (Linux) http://www.go-mono.com

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Microsoft .NET Strategie

.NET Framework

Visual Studio .NET
Codeeditor
Fenstereditor
Debugger
Server-Explorer
Entwurfshilfen
C# / VB .NET / J#
ASP.NET

.NET Enterprice Server

Application Center
Exchange Server
BizTalk Server
...

-werden speziell an .NET angepasst
- Dienste zentral für .NET Anwen-dungen

.NET My Services

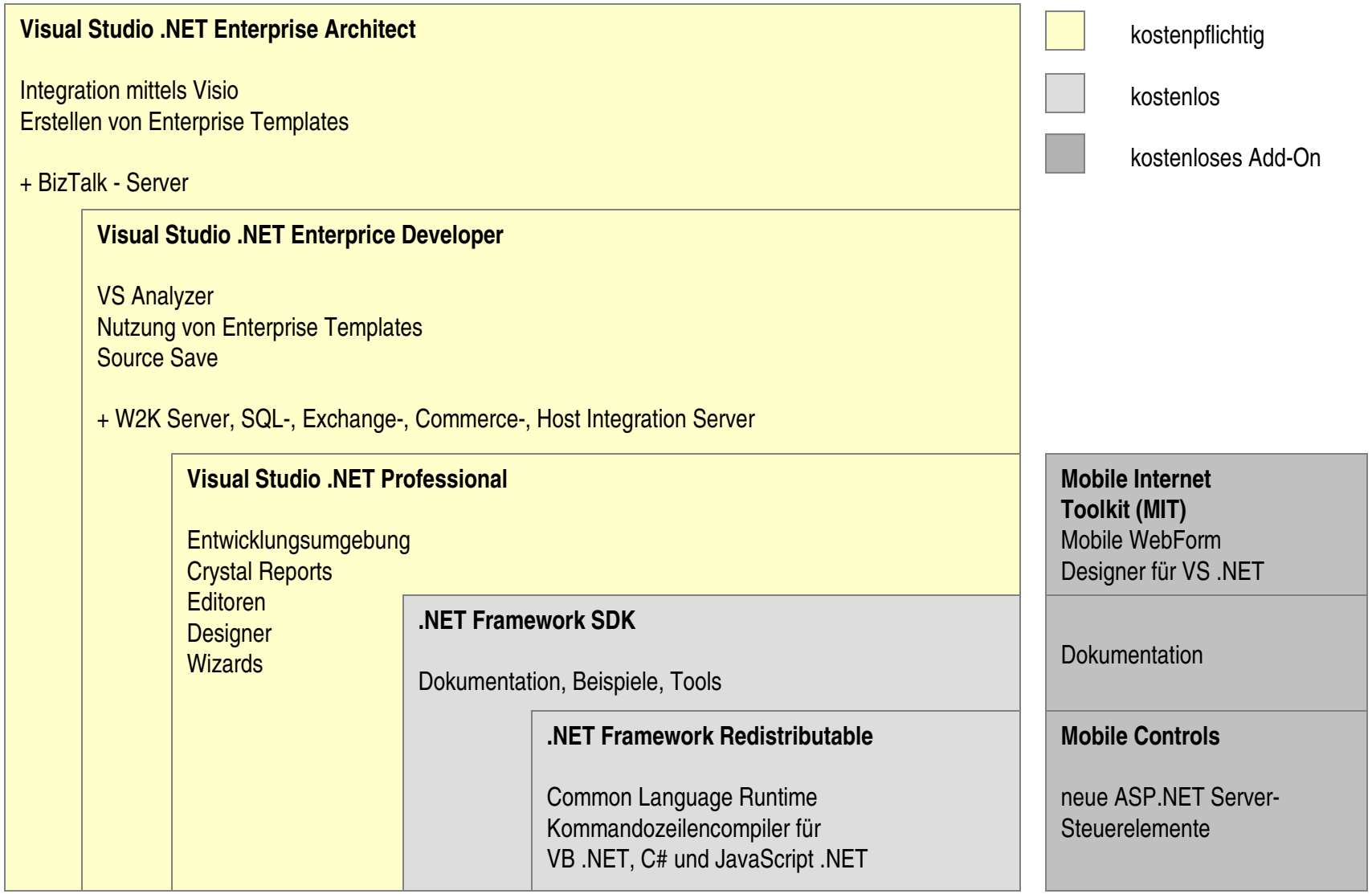
konkrete WebServices

.NET Contacts
.NET Wallet
.NET Lists
Microsoft Passport
...

.NET Devices

Hardware wie PDA, Handys, Tablet, PC, Auto-PC ...

- Mobile Internet Toolkit (MIT)
- .NET Compact Framework (CF)


Visual Studio .NET Enterprise Architect
Integration mittels Visio
Erstellen von Enterprise Templates
+ BizTalk - Server
Visual Studio .NET Enterprice Developer
VS Analyzer
Nutzung von Enterprise Templates
Source Save
+ W2K Server, SQL-, Exchange-, Commerce-, Host Integration Server
Visual Studio .NET Professional
Entwicklungsumgebung
Crystal Reports
Editoren
Designer
Wizards
.NET Framework SDK
Dokumentation, Beispiele, Tools
.NET Framework Redistributable
Common Language Runtime
Kommandozeilencompiler für
VB .NET, C# und JavaScript .NET
kostenpflichtig
kostenlos
kostenloses Add-On
Mobile Internet Toolkit (MIT)
Mobile WebForm
Designer für VS .NET
Dokumentation
Mobile Controls
neue ASP.NET Server-Steuerelemente