Vorlesung
Software aus Komponenten

3. Komponentenmodelle

Prof. Dr. Hans-Gert Gräbe
Wintersemester 2013/14

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

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

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)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

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

> "... 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 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. Java-Idee ohne Beschränkung auf Java als Programmiersprache
- Ziele
  - Sicherheit
  - Plattformunabhängigkeit
  - Interoperabilität
  - Homogenität

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

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 untereinander nicht kompatibel
  - String-Datentypen waren sogar nicht binär kompatibel - .NET ist konsequent Unicode basiert
  - kein einheitliches Modell der Speicherverwaltung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- 1996: erste Arbeiten an .NET
- 2000: .NET-Framework 1.0 Beta
- 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
- Januar 2002: .NET Framework 1.0 und Visual Studio .NET
- April 2003 – .NET Framework 1.1 und Verabschiedung der ISO-Standards ISO/IEC 23270 (C#), ISO/IEC 23271 (CLI)
- 2004: Marktanteil steht noch immer in keinem Verhältnis zur Aufmerksamkeit, die .NET in den Medien findet

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

ECMA-Standardisierung erlaubt Implementierung des Standards auch auf anderen Plattformen.

Versionen jenseits von Windows:

- Microsoft selbst stellte 2002 mit der Shared Source CLI Versionen für Mac OS und Linux bereit. Diese Aktivitäten wurden später wieder aufgegeben.
- Mit dem Mono-Projekt entstehen ab 2004 erste freie Implementierungen von .NET-Konzepten, aktuell zwei Linien:
  - Mono-Projekt von Ximian, dotGNU-Projekt arbeitet an einer Laufzeitumgebung Portable.NET

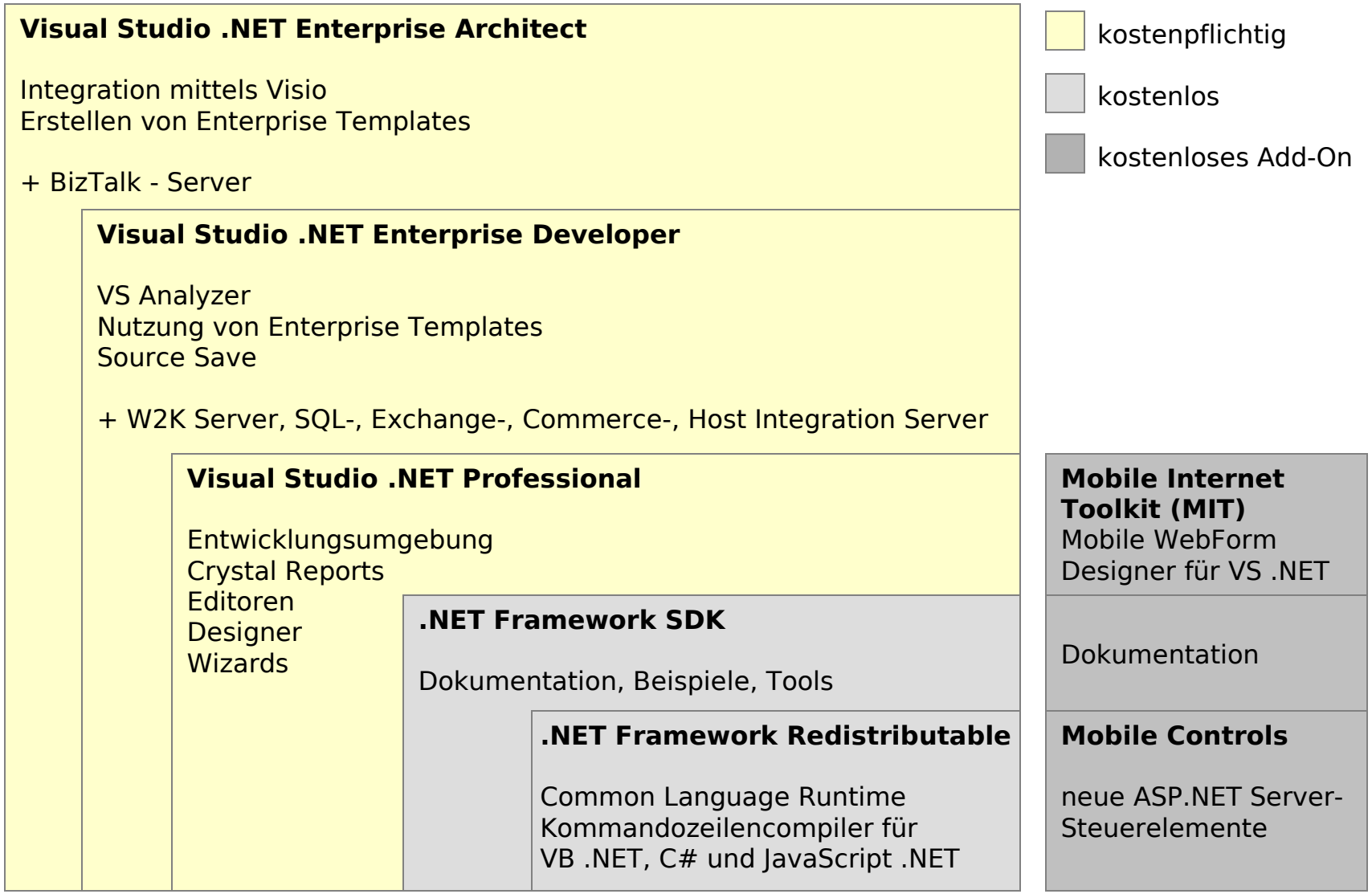
  Bleiben hinter der Leistungsfähigkeit der Windows-Versionen zurück.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- Ende 2005: Visual Studio 2005, .NET Framework 2.0
  - Zusammen mit MS SQL Server 2005, MS BizTalk Server 2006
  - Viele Anwendungen verwenden noch starken Durchgriff auf assemblerspezifischen Windows-Code jenseits der IL, damit nur eingeschränkte Plattformunabhängigkeit und Interoperabilität
- Ende 2006: .NET 3.0, später integraler Bestandteil von Windows Vista und Windows Server 2008, mit tiefgreifenden auch konzeptionellen Änderungen an der Architektur
  - WPF – Windows Presentation Foundation: Beschreibungssprache XAML zur Darstellung von Objekten auf dem Bildschirm.
  - WCF – Windows Communication Foundation: Dienstorientierte Kommunikationsplattform für lose gekoppelte verteilte Anwendungen.
  - WWF – Windows Workflow Foundation: Infrastruktur für die einfachere Entwicklung von Workflow-Anwendungen.
  - Windows CardSpace: Identitätsmanagement-Infrastruktur für verteilte Anwendungen.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

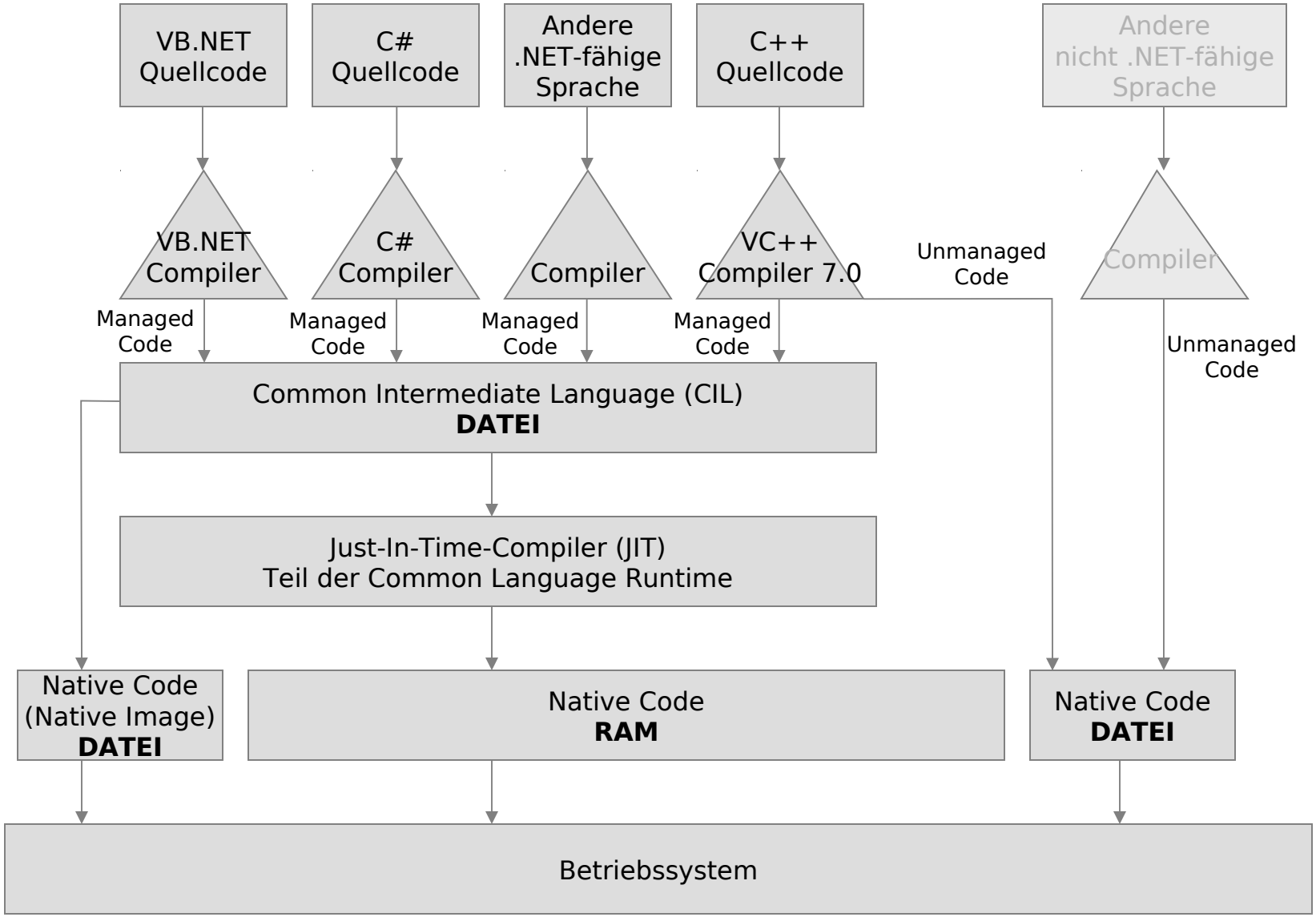
- Ende 2007: Visual Studio 2008 und .NET Framework 3.5
  - Base Class Library (BCL) wird in Framework Class Library (FCL) umbenannt
    - Umfasst fast 12.000 Klassen in 300 Namensräumen
  - Teilweise Freigabe des Quellcodes der Base Class Library unter der restriktiven MS Reference Source License
  - Damit wird die Diskussion um Urheberrechtsverletzungen durch das Mono-Projekt weiter angeheizt
  - Deal zwischen Novell (SuSe-Linux und Mono-Projekt) und Microsoft
- April 2010: .NET 4.0 und Visual Studio 2010
  - Stärkere Ausrichtung auf Mehrkern-Systeme, verteilte Architekturen und Thread-Parallelität
  - Neues Programmiermodell für Multithreading und asynchronen Code
- August 2012: .NET 4.5 (Aktuell .NET 4.5.1)


Visual Studio .NET Enterprise Architect
Integration mittels Visio
Erstellen von Enterprise Templates
+ BizTalk - Server
Visual Studio .NET Enterprise 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

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Das .NET-Konzept

- CLI – **Common Language Infrastructure**
  - Basis zur Ausführung von Programmen, die in unterschiedlichen Programmiersprachen erstellt wurden
  - Zugriff auf eine **virtuelle Maschine** und eine gemeinsame Klassenbibliothek – die **Base Class Library**
- CIL - **Common Intermediate Language**
  - Hochsprachenunabhängige Zwischensprache
- CLR – **Common Language Runtime**
  - Laufzeitumgebung für CIL-Zwischencode
- CTS – **Common Type System**
  - Sicherung der Kompatibilität der Ressourcenzugriffe über einen sprachübergreifenden Standard von OO-Datentypen
  - .NET wurde von Anfang an für den Betrieb mit mehreren Programmiersprachen entwickelt
- Assemblies – **Packungsformat** für Komponenten


VB.NET Quellcode
C# Quellcode
Andere .NET-fähige Sprache
C++ Quellcode
Andere nicht .NET-fähige Sprache
VB.NET Compiler
C# Compiler
Compiler
VC++ Compiler 7.0
Compiler
Unmanaged Code
Managed Code
Managed Code
Managed Code
Managed Code
Unmanaged Code
Common Intermediate Language (CIL)
DATEI
Just-In-Time-Compiler (JIT)
Teil der Common Language Runtime
Native Code (Native Image)
DATEI
Native Code
RAM
Native Code
DATEI
Betriebssystem