Vorlesung
Software aus Komponenten

3. Komponentenmodelle

Prof. Dr. Hans-Gert Gräbe
Wintersemester 2014/15

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

3.7. Das .NET-Konzept
Geschichtliche Einordnung

- 1996: erste Arbeiten an .NET
- 2000: .NET-Framework 1.0 Beta
- Oktober 2000 – C# und die CLI werden von MS, HP und Intel zur Standardisierung bei der ECMA eingereicht
  - ECMA – European Computer Manufacturers Association
  - Dezember 2001 – Weitergabe des ersten Standards an die ISO
  - April 2003 – Verabschiedung der ISO-Standards ISO/IEC 23270 (C#) und ISO/IEC 23271 (CLI)
- Januar 2002: .NET Framework 1.0 und Visual Studio .NET
- April 2003 – Auslieferung von .NET Framework 1.1 zusammen mit Windows Server 2003, der eine integrierte .NET-Laufzeitumgebung zur Verfügung stellt.
  - Damit Übergang zur neuen Plattform auf der Ebene von Konzepten für Unternehmensserver
  - Integration in die Produktfamilie geht jedoch nicht so rasch voran wie erwartet

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- Ende 2005: .NET Framework 2.0 zusammen mit Visual Studio 2005, MS SQL Server 2005, MS BizTalk Server 2006
  - Damit weitere Integration in die Produktfamilie, aber 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 Erweiterungen der Architektur
- 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 Microsoft Reference Source License

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- 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
- Mai 2014: .NET 4.5.1. - .NET 4.6 ist in der Pipeline

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)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

.NET als Komponentenframework

.NET 3.0 schärft das Profil als Komponentenframework mit mehreren Konzepten, vergleichbar den CORBA Facilities

- *WPF – Windows Presentation Foundation:* Beschreibungssprache XAML zur Darstellung von Objekten auf dem Bildschirm.
- *WCF – Windows Communication Foundation:* Dienstorientierte Kommunikationsplattform für für verteilte Anwendungen, in der viele Netzwerk-Funktionen zusammengeführt und standardisiert zur Verfügung gestellt werden.
- *WWF – Windows Workflow Foundation:* Infrastruktur für die einfache Entwicklung von Workflow-Anwendungen, sowohl in geschäftlicher als auch technischer Hinsicht, mit visuellen Modellierungsmöglichkeiten.
- *Windows CardSpace:* Identitätsmanagement-Infrastruktur für verteilte Anwendungen als Basis eines einheitlichen Authentifizierungs- und Autorisierungs-Frameworks

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 FreeBSD bereit. Diese Aktivitäten wurden später wieder aufgegeben.
- Verschiedene Aktivitäten der Linux-Community, die Konzepte umzusetzen und eine freie .NET-Version zu schaffen.
  - 2009 startet das dotGNU-Projekt, das eine Laufzeitumgebung Portable.NET erstellen will. Kommt über Release-Version 0.1 nicht hinaus und wird Ende 2012 eingestellt.
    - As of December 2012, the DotGNU project has been decommissioned, until and unless a substantial new volunteer effort arises.
- Bleiben hinter der Leistungsfähigkeit der Windows-Versionen zurück.
- Einziges leistungsfähiges „freies“ Projekt ist das Mono-Projekt http://www.mono-project.com/

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Zur Geschichte des Mono-Projekts

- Miguel de Icaza und Nat Friedman gründen 1999 die Firma *Helix Code*, die 2001 in *Ximian* umbenannt wird.
  - Geschäftsmodell: Solutions and Services, basierend auf Mix von freier und kommerzieller Software
  - Beteiligt an Gründung des Gnome Projekts
  - 2002 Start des Mono-Projekts
- 2003 von *Novell* übernommen, das damit sein Linux-Portfolio weiter stärkt.
- 2011 wird Novell im Rahmen des großen Patentdeals von der *Attachmate Group* übernommen, die kein Interesse an der Weiterführung des Mono-Projekts haben.
  - After several months of discussions, the US Department of Justice (DOJ) and the German Federal Competition Office (FCO) have allowed a consortium of Microsoft, Oracle, Apple and EMC to acquire 882 patents from Novell only subject to conditions clearly intended to prevent their use against Free Software players. (FSFE Newsletter, April 2011)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Zur Geschichte des Mono-Projekts (Fortsetzung)

- 2011 gründen Icaza und Friedman die Firma *Xamarin* http://xamarin.com und bündeln dort die weitere Entwicklung am Mono-Projekt
  - Fokus der Firma liegt auf mobilen Anwendungen.
- Der Mono-Kern, die Laufzeitumgebung, ist unter der LGPL v.2 frei verfügbar, aber Xamarin bietet auch kommerzielle Lizenzen für die Mono-Plattform an
  - If you are planning to use Mono as a bundled part of your commercial product, on embedded hardware, or in any other situation where using the LGPL-licensed Mono is impossible or problematic, Xamarin can sell you a commercially-friendly license that will suit your needs.
  - Many commercial users of Mono acquire a commercial license when they want the flexibility and peace of mind to use Mono without worrying about the terms of the LGPL.
- Neue Etappe der Zusammenarbeit: Ende 2013 gründen Microsoft, Xamarin und andere die *.NET Foundation* als neuer Rechteinhaber und Lizenzgeber des .NET Frameworks.
  - http://www.dotnetfoundation.org/

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

.NET Open Sourcing

- 2008 veröffentlichte Microsoft den Quelltext des Frameworks unter der restriktiven Microsoft Reference License.
- Ende 2013 gründeten Microsoft, Xamarin und andere die *.NET Foundation* als neuer Rechteinhaber und Lizenzgeber des .NET Frameworks. http://www.dotnetfoundation.org/
  - 2007 hatte Microsoft noch behauptet, dass das Mono-Projekt Rechte von Microsoft verletzt
- Ende 2014 wird eine Teilmenge des Reference Source Quellcodes auf GitHub gehosted und unter der MIT-Lizenz veröffentlicht.
  - Dies geschah auch, um Lücken zwischen Mono und .NET durch Verwendung desselben Codes geschlossen werden können.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

.NET Open Sourcing

- Gleichzeitig hat Microsoft damit begonnen, die überarbeiteten Komponenten des Framework unter der Bezeichnung *.NET Core* auf GitHub ebenfalls unter MIT-Lizenz zu veröffentlichen.
- Basis für das kommende, modular aufgebaute *.NET Framework 5*.
  - .NET Core ist von Microsoft an die .NET Foundation überstellt worden.
- Durch die Verwendung der MIT-Lizenz gibt es faktisch keine Einschränkungen mehr, wie der Quellcode von .NET Core verwendet werden darf.
  - Mit der Gründung der .NET Foundation und der Übertragung der Rechte und Quellcodes an die Foundation arbeitet Microsoft mit Xamarin aktiv zusammen, um .NET auf unterschiedlichen Plattformen bereitzustellen. Durch die Offenlegung der Quellcodes unter der MIT-Lizenz bzw. Apache 2.0 Lizenz ist der Quellcode des .NET Frameworks nahezu beliebig – auch in Closed-Source-Projekten – verwendbar. Lizenz- und patentrechtliche Auseinandersetzungen sind somit kaum noch möglich und auch nicht mehr zu befürchten. (Wikipedia)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Die Entwicklungsumgebung von Microsoft

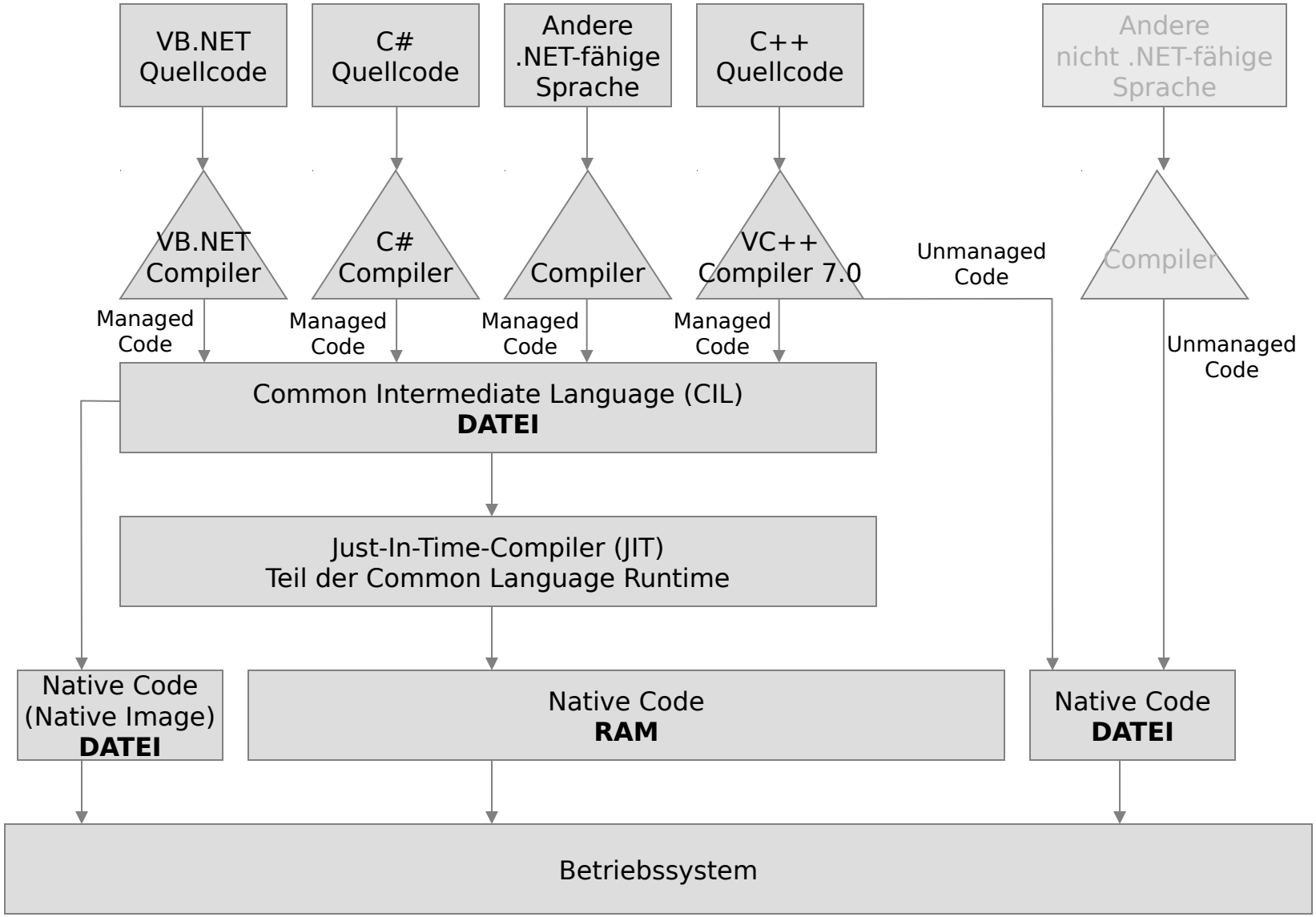
Microsoft bietet .NET in verschiedenen Formen an.

- Als reine Laufzeitumgebung samt benötigter Klassenbibliotheken (Framework)
- Als kostenloses SDK Development Kit für Entwickler
- Als kostenpflichtige integrierte Entwicklungsumgebung (IDE) in Form des Microsoft Visual Studio .NET.
- Speziell für Einsteiger und Studenten gibt es die kostenlosen Microsoft Visual Studio Express Editions mit Einschränkungen gegenüber den kostenpflichtigen Standard- oder Professional-Varianten.
- Eine ebenfalls kostenfreie IDE für .NET (und Mono) unter Windows findet man im Open-Source-Projekt SharpDevelop.
  - #develop (short for SharpDevelop) is a free IDE for C#, VB.NET and Boo projects on Microsoft's .NET platform. SharpDevelop is distributed under the MIT license. http://www.icsharpcode.net
- Studenten können über das *DreamSpark-Programm* kostenfrei die Professional-Variante des Visual Studios zu beziehen.

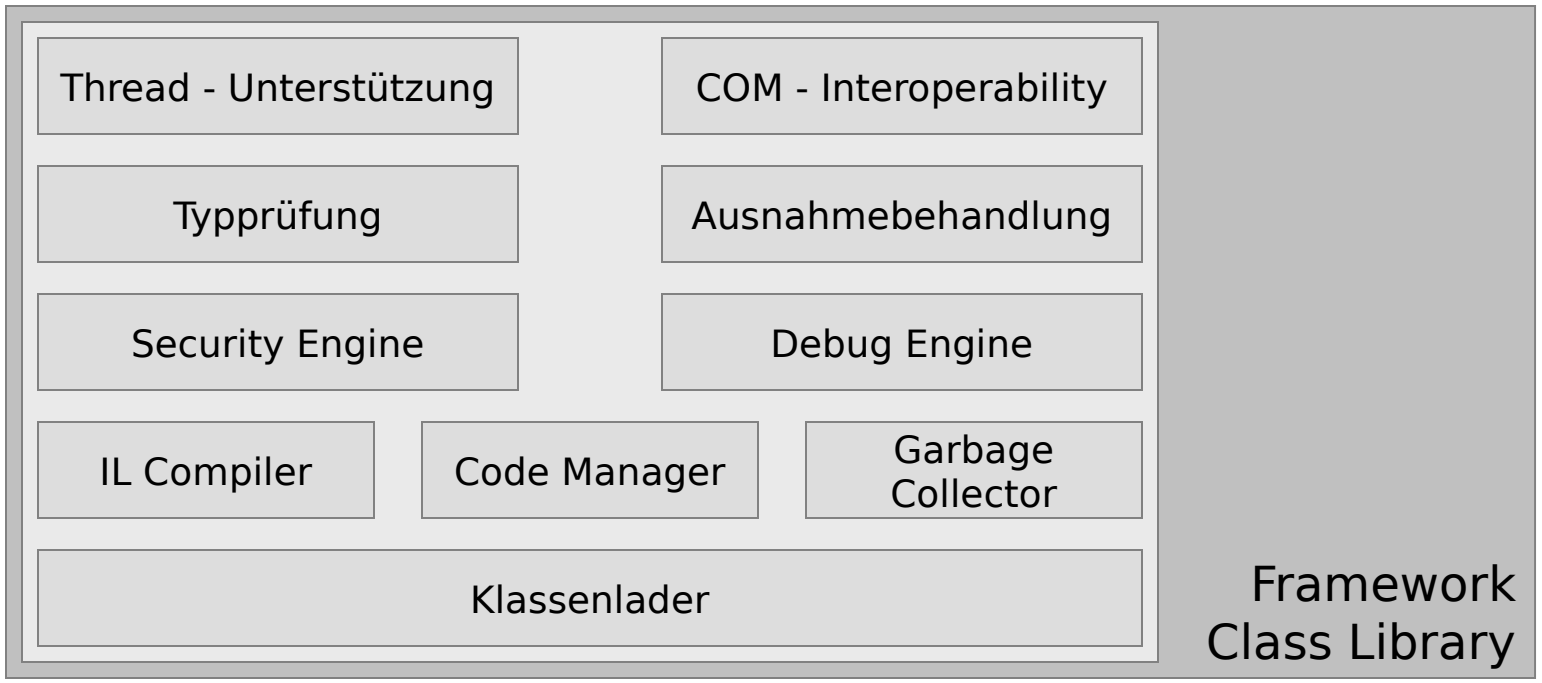
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

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 **Framework 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

- CLR übersetzt Zwischensprachencode (CIL) in Maschinencode
- Speicherverwaltung
- Verwaltung von Prozessen und Threads
- Durchsetzung von Sicherheitsmechanismen
- Laden von Komponenten
- **Alle** .NET-Sprachen setzen auf die CLR als Runtime auf

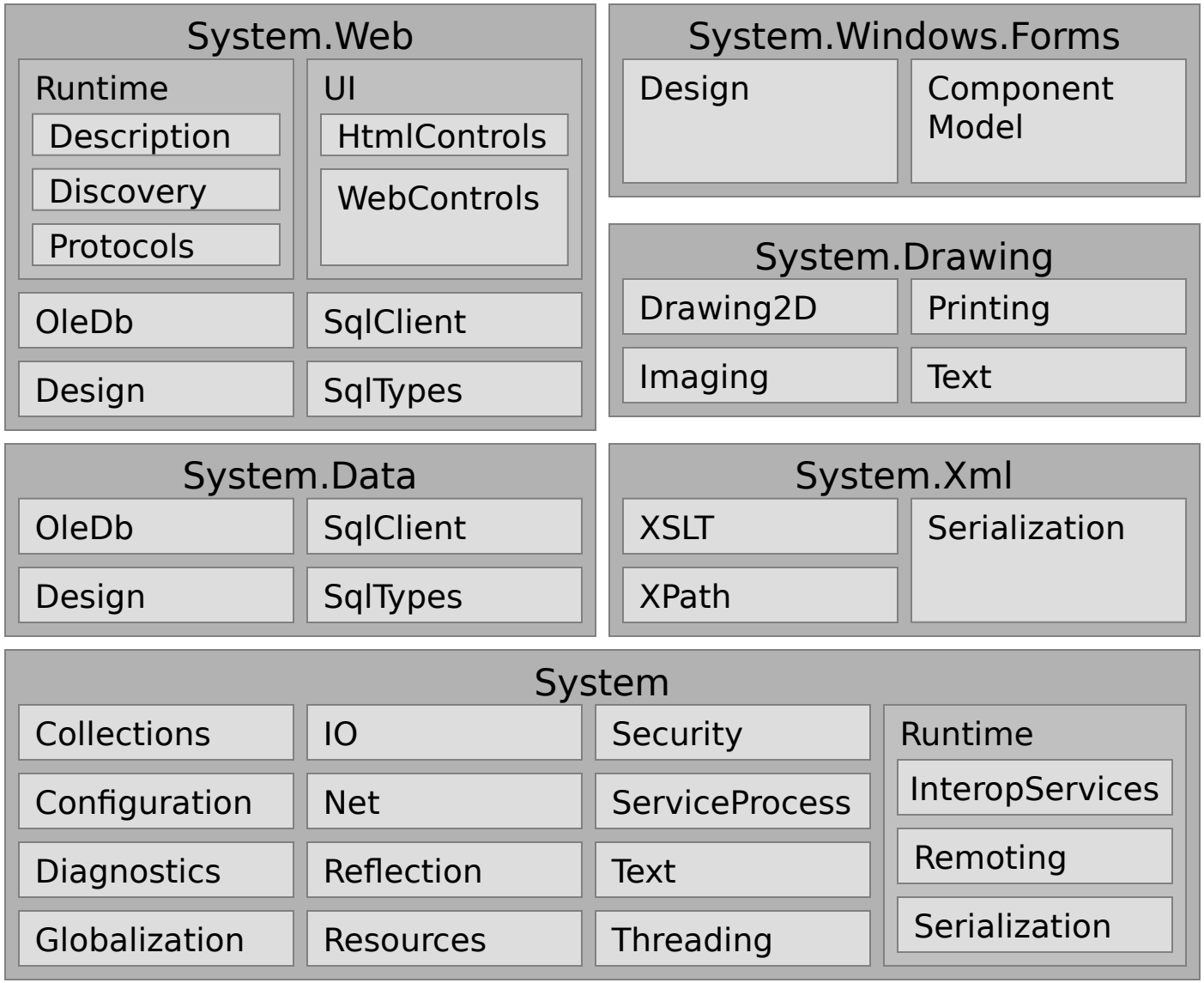


UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- Konsistentes Programmiermodell
  - alle Anwendungsdienste als objektorientiertes Programmiermodell
- Vereinfachtes Programmiermodell
  - keine Registrierung und eigenständige Registry-Verwaltung
- Stabile Installationen
  - isolierte Anwendungskomponenten
  - keine „DLL-Hölle“ mehr
  - Versionierung von Komponenten
- Vereinfachte Installationen
  - Anwendungsdateien können einfach in Zielverzeichnis kopiert werden
  - keine Registry-Einträge nötig
- Viele verfügbare Plattformen
  - Compiler generiert IL-Code
  - Ausführbar auf Maschinen, die über ECMA-kompatible Versionen der CLR verfügen
- Integration verschiedener Programmiersprachen
  - Typen, die in unterschiedlichen Sprachen geschrieben wurden
  - Common Type System

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- Einfacheres Wiederverwenden von Code
  - durch oben beschriebene Techniken
- Automatische Speicherverwaltung
  - Garbage Collection
- Typsicherheit
  - Zugriff auf Objekte immer auf kompatible Weise
  - Code springt nur an bekannte Stellen (Eintrittspunkt von Methoden)
  - keine Pufferüberläufe
- Komfortables Debuggen
  - Debuggen von Anwendungen unterschiedlicher Sprachen
- Konsistente Fehlerverarbeitung
  - **Alle** Fehler werden über Ausnahmen gemeldet
- Sicherheit
  - basierend auf Herkunft des Codes (code based security) oder der Daten (role based security)
- Interoperabilität
  - Zugriff an der CIL vorbei auf Komponenten möglich, die den älteren COM-Standard implementieren



- Schnittstelle zum Betriebssytem
- komplett Objektorientiert
- Allen .NET Sprachen stehen dieselben Dienste zur Verfügung
- Zugriff auf Dateisystem, Fensteranzeige, Druckfunktionen, Remoting, Grafik, Datenbankzugriff

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- CIL ist eine Art "objektorientierter Maschinencode"
- arbeitet Stack-orientiert, keine Register
- unabhängig von CPU
- Verifizierung des Codes durch die CLR bei der Übersetzung in nativen Code
  - nur Speicheradressen lesen, in die vorher Daten geschrieben wurden
  - Methoden mit der korrekten Anzahl von Argumenten aufrufen
  - jedes Argument hat richtigen Typ
- wird zur Laufzeit vom Just-In-Time-Compiler kompiliert
- Caching von bereits übersetzten Typen
- Optimierung anhand ausführender Architektur

```
.method private hidebysig static void Main() cil managed {
        .entrypoint
        .maxstack 3
        .locals ([0] int32 v, [1] object o)
        IL_0000:  ldc.i4.5
        IL_0001:  stloc.0
        IL_0002:  ldloc.0
        IL_0003:  box        [mscorlib]System.Int32
        ...
```

Beispiel für IL-Code

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

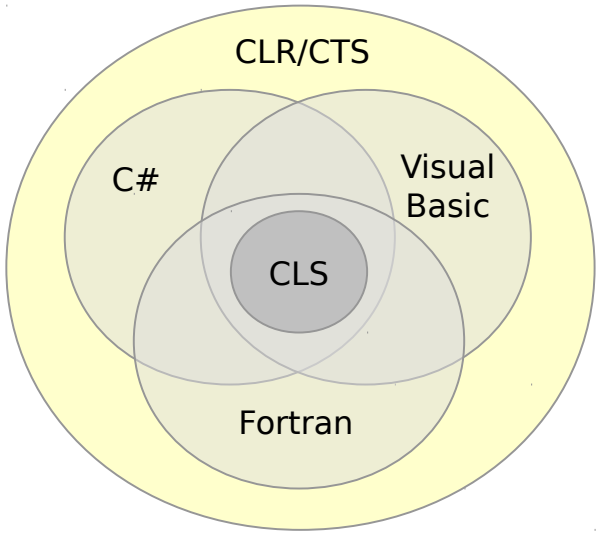
Common Language Specification

kleinster gemeinsamer Nenner der .NET-Sprachen

- standardisierte Typen
- selbstbeschreibende Typinformationen (Metadaten)
- gemeinsame Ausführungsumgebung

```
using System;
[assembly:CLSCompliant(true)]
// Compiler soll CLS-Kompatibilität prüfen

// Fehler, weil Klasse öffentlich ist
public class App {
    // Fehler, weil UInt32 nicht CLS-Kompatibel
    public UInt32 Abc() { return 0; }
    // Fehler, weil keine Unterscheidung zwischen
    // Groß- und Kleinschreibung in CLS
    public void abc() {}
    //Kein Fehler, da Methode privat ist
    private UInt32 ABC() { return 0;}
```



Vollständige Liste der CLS-Regeln im Abschnitt „Cross-Language Interoperability“ in der Dokumentation des .NET Framework SDK