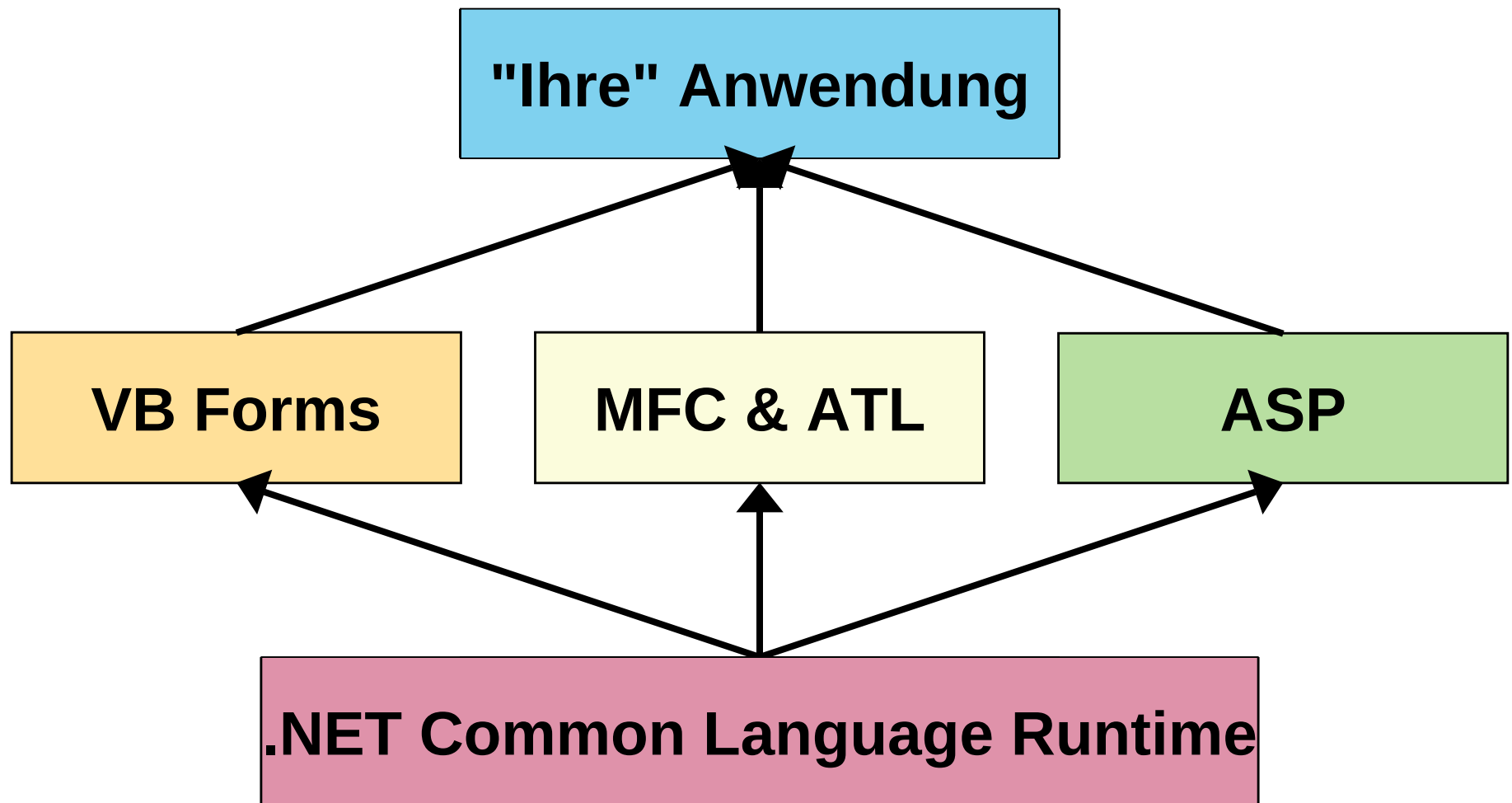
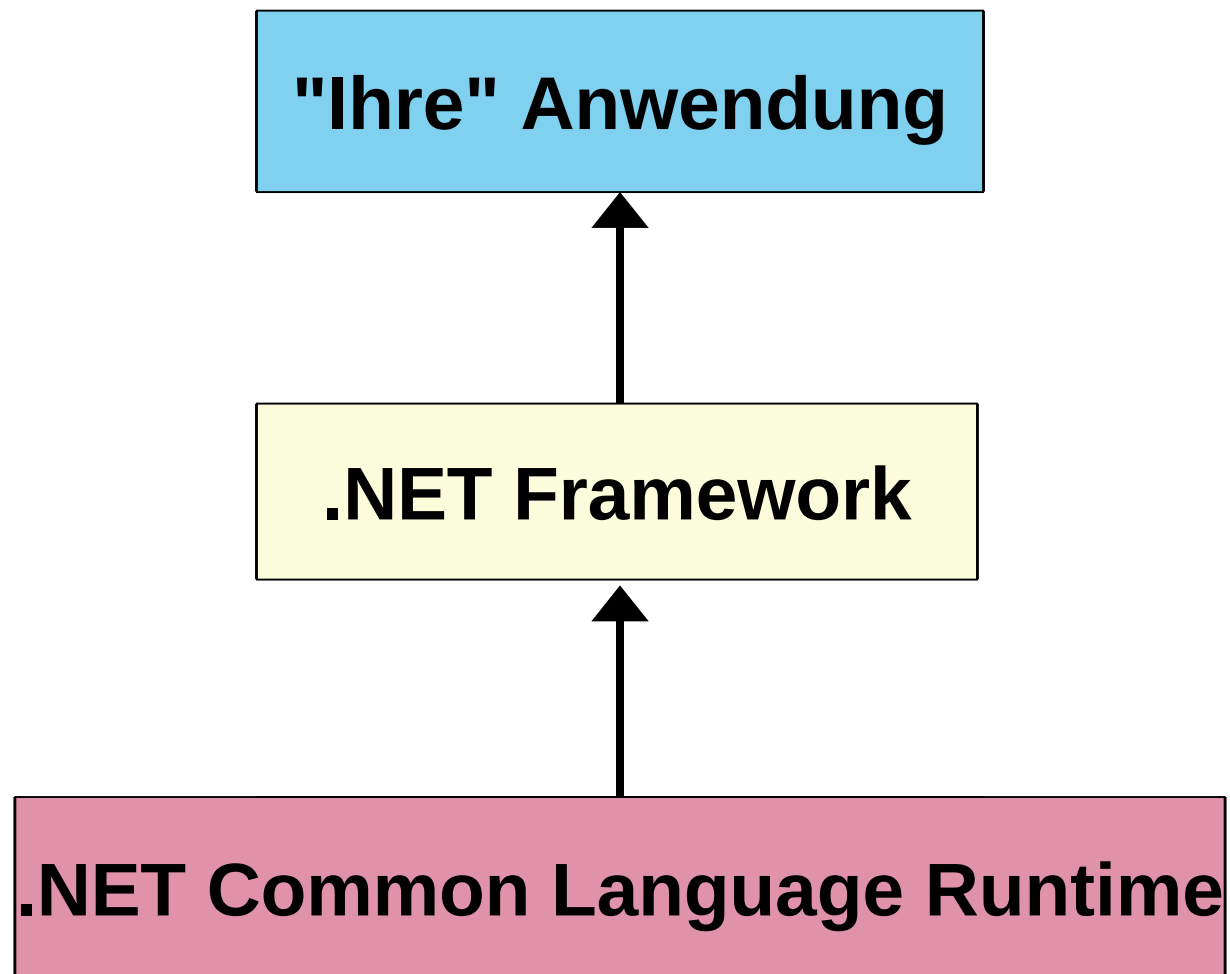


Der Einsatz von .NET im industriellen Umfeld

- Was ist .NET
- Einsatzszenarien
- Vorteile/ Nachteile
- Fazit

- Neue Programmierphilosophie für MS Windows
 - Einheitliches Programmiermodell
 - Programmiersprachenunabhängig
- Betriebssystem-Aufsatz
 - Neues, erweitertes Sicherheitssystem





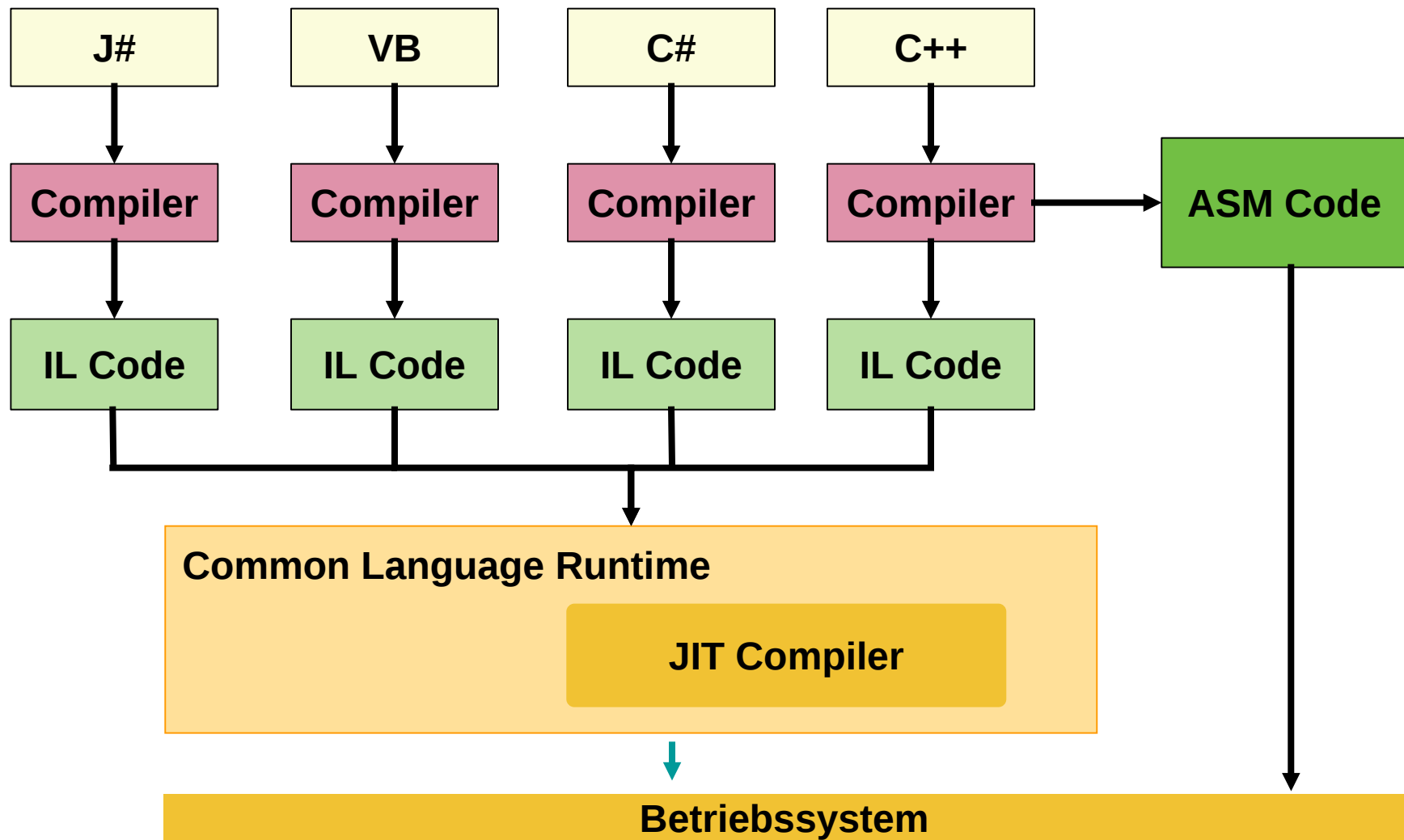
**Web
Services**

User Interface

Data

XML

Base Class Library



Sprachen werden gleichwertig, da alle Compiler IL-Code erzeugen

Die Common Language Runtime **CLR** ist die Weiterentwicklung der bisherigen Konzepte zu einem einheitlichen Integrationsmodell.

Sie bietet

- ein einheitliches und sprachübergreifendes Typsystem
- ein einheitliches und erweiterbares Format für Metadaten
- eine einheitliche und sprachübergreifende Klassenbibliothek
- eine einheitliche und sprachübergreifende Laufzeitumgebung

Sämtlicher Code wird unter Aufsicht der Common Language Runtime ausgeführt

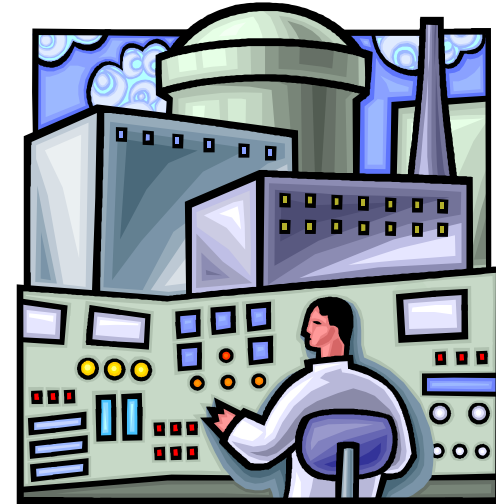
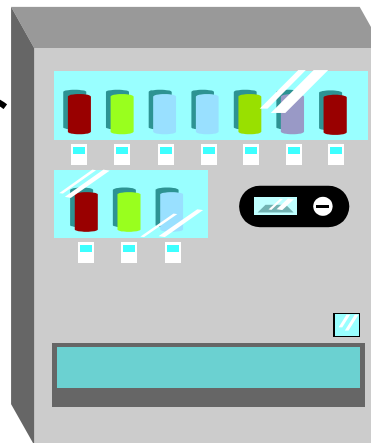
- Runtime führt Sicherheitsüberprüfungen aus
- Runtime übernimmt Speicherverwaltung und Fehlerbehandlung (GC, Exceptions)
- Runtime führt Versionsprüfungen aus

→ Dieser Code wird "**Managed Code**" genannt.

Direkt im
Embedded
Device



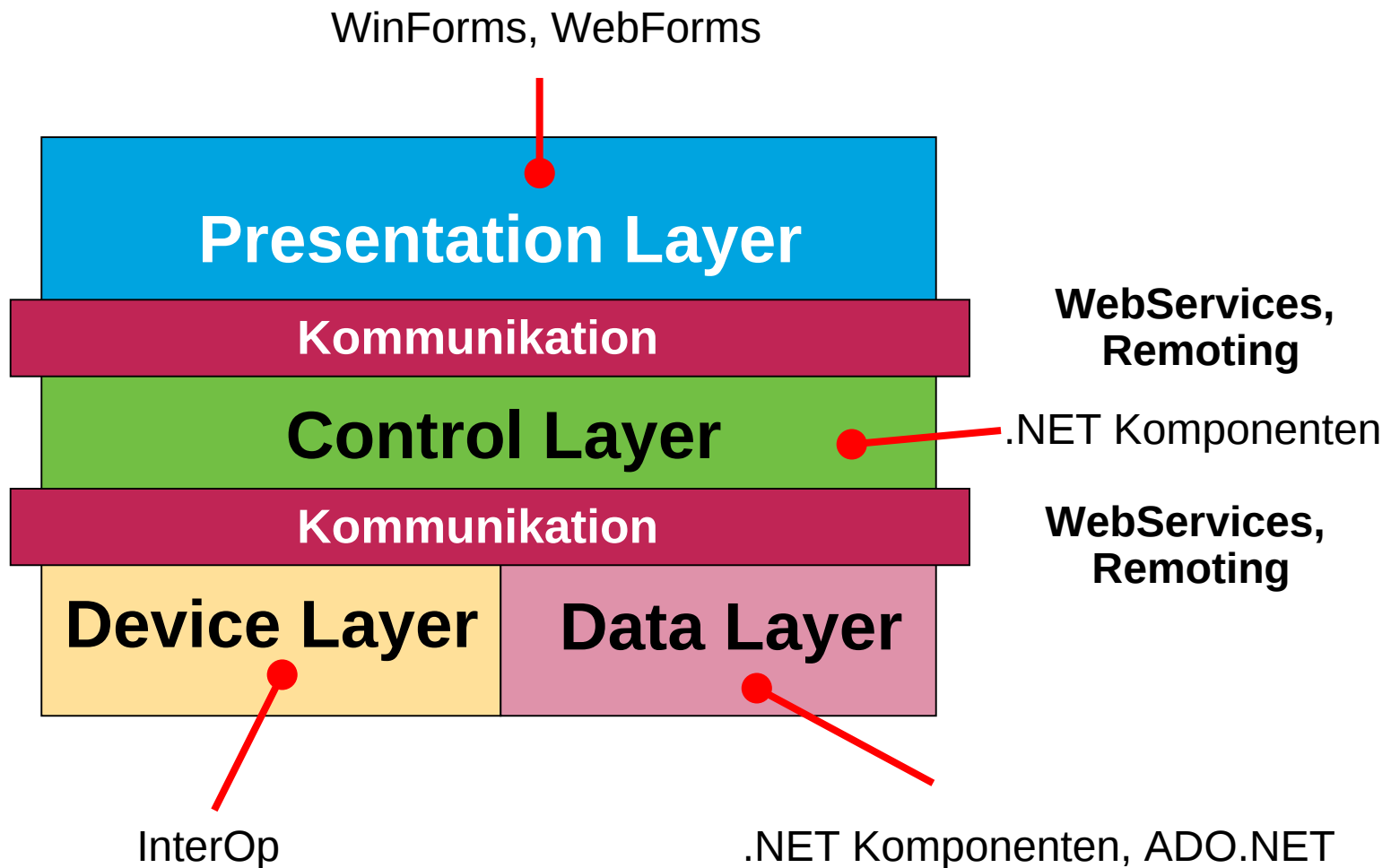
Diagnosegeräte



Bedienung/ Visualisierung

Welche Teile aus dem .NET Framework sind für industrielle Anwendungen interessant?

- WebForms, WinForms inklusive GDI+
- .NET Komponentenmodell
- Remoting
- WebServices
- XML
- ggf. ADO.NET
- InterOp für die Integration und Migration bestehender Komponenten



- WinForm
 - Ersetzt die MFC und Visual Basic Programmierung
 - Volle RAD-Unterstützung (Rapid Application Development)
(damit auch für Prototyping geeignet)
- WebForm
 - Volle Internet-Fähigkeit für den Presentation Layer
 - Setzt völlig neue Maßstäbe für den Bau von Applikationen für
Fernwartung
Ferndiagnose
Fernsteuerung

Implementierung des Presentation Layer

- .NET Komponenten
 - Ersetzen den Inprocess Server oder das bisherige DLL-Modell
 - Liefern Assemblies als DLLs
 - Shared Assemblies unterstützen die Versionierung
 - Sehr einfach im Deployment
 - XCOPY-Deployment
 - Keine Registry-Einträge mehr nötig
 - Shared Assemblies werden in den Global Assembly-Cache eingetragen

Einfaches Konzept für die Modularisierung

- Remoting
 - Ermöglicht eine sehr einfache Verteilung von Komponenten im Netz
 - Ersetzt DCOM
 - Sehr einfach in der Programmierung
 - Sehr einfach in Handhabung und Installation
- **Kommunikation zum**
 - **Control Layer**
 - **Abbildung von Prozessen im Internet/Intranet**
 - **Device Layer**
 - **Fernwartung/-diagnose von Systemkomponenten und externen Geräten**

- Webservices
 - Bilden Funktionen (Methoden) über SOAP im Internet ab
 - Ideal für
 - Fernwartung
 - Ferndiagnose
 - Fernsteuerung
 - **Kommunikation zum**
 - **Control Layer**
 - **Abbildung von Prozessen im Internet**
 - **Device Layer**
 - **Fernwartung/-diagnose von Systemkomponenten und externen Geräten**

- InterOp
 - ermöglicht
 - die Integration bestehender Applikationen
 - weiche Migration bestehender Sourcen ins .NET
 - Techniken
 - Aufruf von Funktionen in DLLs
 - Einbinden von COM-Komponenten
 - Einbinden von ActiveX-Controls
 - Kombination von Managed und Unmanaged C++ als Gateway

- Entwickelt für Embedded Devices
 - ROM-Bedarf ca. 3,1 MB (.NET Framework: ca. 30 MB)
- Was ist nicht enthalten?
 - Reflection
 - Remoting
 - Serialization
 - Code Dom
 - Printing
 - COM Interoperability
 - Server Side Functionality
 - XPath/XSLT

Vorteile von .NET

- Komfortable und einfache Entwicklung
 - Schnellere Entwicklung von Projekten (RAD)
- Sicherer Code
- Garbage Collection
- Funktionierende Versionierung

Nachteile von .NET

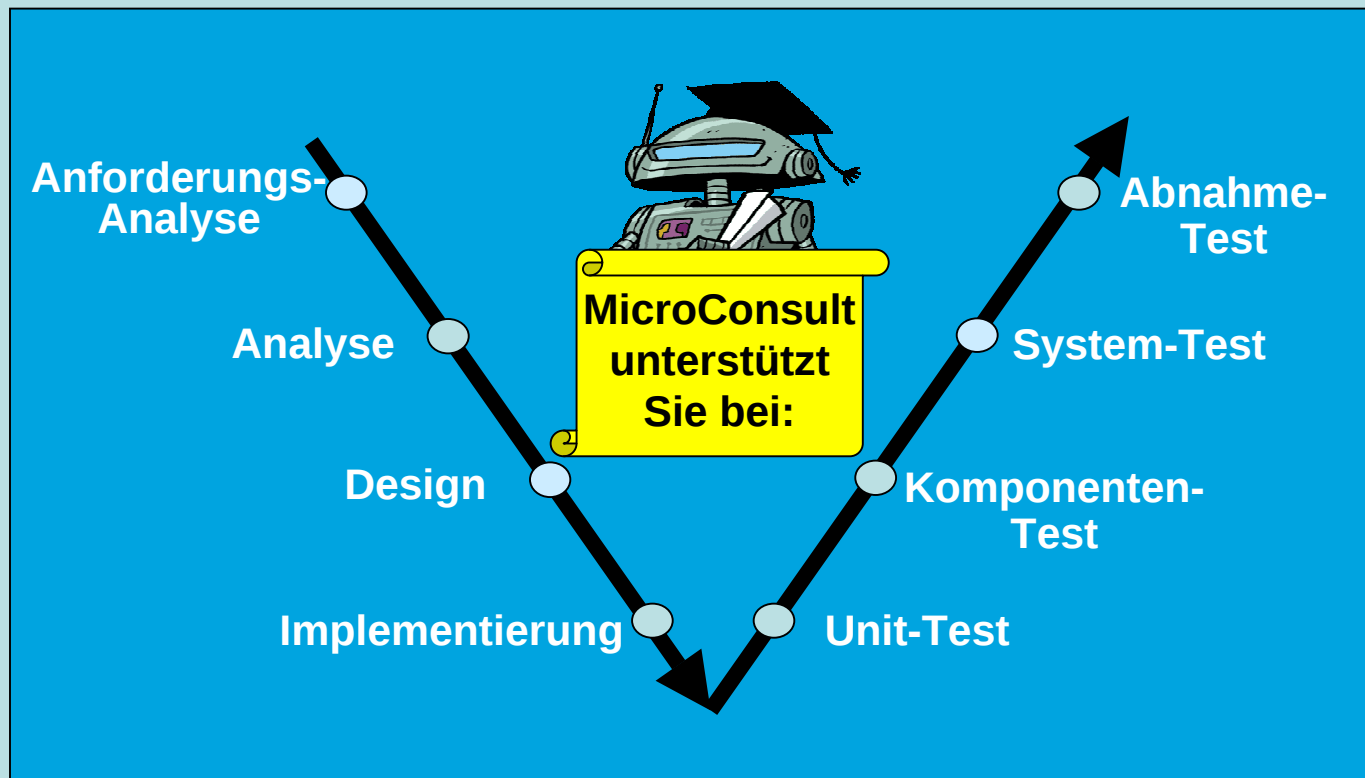
- C++ Konzepte sind oft nicht übernehmbar.
- Umstellung von Entwicklergewohnheiten sind nötig.
- Entwickler haben noch kein Gefühl für die Performance.
- Garbage Collector ist kaum beeinflussbar.

- .NET ist keine "Eierlegende Wollmilchsau"
- An den richtigen Stellen eingesetzt, sorgt es für einen Produktivitätsschub.
- Im Echtzeitbereich ist es eher ungeeignet.
- Treiber können nicht mit .NET-Mitteln erstellt werden.

.NET entwickelt sehr große Stärken:

- **Wo es um Visualisierung geht!**
- **Wo ein Höchstmaß von Internet Connectivity erforderlich ist!**
- **Wo Distributed Systems gefragt sind!**

Training, Coaching, Engineering



HW-/SW-Technologien, Tools, Methoden, Prozess, Team