



Client/Server-Programmierung

WS 2019/2020

Roland Wismüller
Betriebssysteme / verteilte Systeme
roland.wismueller@uni-siegen.de
Tel.: 0271/740-4050, Büro: H-B 8404

Stand: 17. Januar 2020



Client/Server-Programmierung

WS 2019/2020

8 Weitere Client/Server-Technologien



8.1 Technologien zur Netzwerkprogrammierung

- ➔ **Sockets:** plattform- und netzwerkunabhängiges Kommunikations-API (Basis jeglicher Client/Server-Kommunikation)
 - ➔ Socket = Abstraktion für Kommunikations-Endpunkt
 - ➔ Datagramm- und *Stream*-Sockets
 - ➔ bei Verwendung der IP-Protokolle: direkte Kommunikation über UDP bzw. TCP
 - ➔ Java bietet vereinfachtes API zur Socket-Kommunikation
 - ➔ Literatur:
 - ➔ Folien-Skript "Rechnernetze II", Kap. 8.1
 - ➔ W.R. Stevens: Programmieren von UNIX-Netzen, Hanser / Prentice Hall, 1992.
 - ➔ T. Langner: Verteilte Anwendungen mit Java, Markt + Technik, 2002. Kap. 3

8.1 Technologien zur Netzwerkprogrammierung ...



- ➔ **Sun RPC:** einfache RPC-Implementierung
 - ➔ einfache IDL, Bindung i.W. nur für C
 - ➔ sehr eingeschränkt (z.B. nur ein Parameter pro Prozedur)
 - ➔ Literatur:
 - ➔ Folien-Skript „Rechnernetze II“ (WS 08/09), Kap. 9.2
 - ➔ W.R. Stevens: Programmieren von UNIX-Netzen, Hanser / Prentice Hall, 1992.
- ➔ **DCE RPC:** Weiterentwicklung, Basis von Microsoft DCOM
 - ➔ IDL mit Bindung an verschiedene Programmiersprachen
 - ➔ Schnittstellen an Objekte statt Maschinen gebunden
 - ➔ weltweit eindeutige ID (UUID) für jede Schnittstelle
 - ➔ Literatur: A. Tanenbaum, M. van Steen: Verteilte Systeme, Pearson Studium, 2003, Kap. 2.2



8.2 Technologien für dynamische WWW-Seiten

➔ CGI-Skripten (*Common Gateway Interface*)

- ➔ ausführbare Programme (in beliebigen Sprachen) werden beim Web-Server für bestimmte URLs registriert
- ➔ bei Aufruf der URL startet Web-Server das CGI-Programm als eigenen Prozeß
 - ➔ Standard-Eingabe: Rumpf der HTTP-Anfrage
 - ➔ d.h. Anfrage-Parameter bei POST-Methode
 - ➔ Standard-Ausgabe: Rumpf der HTTP-Antwort
 - ➔ Zusätzliche Information durch Umgebungsvariablen, z.B. QUERY_STRING: Parameterstring in der URL nach dem ?
- ➔ Vorläufer von Servlets
- ➔ Literatur: Orfali/Harkey, Kap. 11

8.2 Technologien für dynamische WWW-Seiten ...



➔ PHP (*PHP: Hypertext Preprocessor*)

- ➔ Skript-Sprache, in HTML-Dokumente eingebettet
 - ➔ innerhalb von *Tags* `<?php ... ?>`
 - ➔ ab PHP 4 mit objektorientierten Konzepten
- ➔ ähnlich zu JSP, aber interpretierte Abarbeitung
- ➔ Literatur: <http://www.php.net/>

➔ ASP (*Active Server Pages*)

- ➔ Integration verschiedener Skript-Sprachen (u.a. VB Script, JScript) in HTML-Dokumente
 - ➔ innerhalb von *Tags* `<% ... %>` (Vorbild für JSP)
- ➔ ASP zunächst interpretiert, bei ASP.NET Vorkompilation des Codes analog zu JSP
- ➔ Literatur: <http://www.asp.net/>

8.3 Microsoft .Net und DCOM

- ➔ Konkurrenz / Analogie zu J2EE von Oracle
- ➔ Das *.Net-Framework* definiert:
 - einheitliche Laufzeitumgebung für alle Microsoft-Sprachen (z.B. Visual Basic, Visual C++, C#)
 - einheitliche Zwischensprache (CIL) mit JIT-Übersetzung
 - einheitliches Typsystem, Garbage Collection, Threads, Sandboxing, Ausnahmen ...
 - gemeinsame Klassenbibliothek (etwa analog zu J2EE)
 - u.a. Realisierung von ASP.Net und ADO.Net (Datenbank-Zugriff, vergleichbar zu JDBC)
- ➔ Zusätzlich: Komponentenmodell, Weiterentwicklung von COM+
 - COM+ = DCOM + MS *Transaction* + *Message Queue Server*
- ➔ Kommunikation: u.a. DCOM, Web Services, *.Net-Remoting*

Anmerkungen zu Folie 437:

.Net-Remoting realisiert entfernte Methodenaufrufe mit Hilfe von Proxies und löst einige der Probleme von DCOM. Es ist inzwischen durch die Windows Communication Foundation ersetzt, die verschiedene Kommunikationstechnologien (u.a. DCOM und Web-Services) mit einer einheitlichen Programmierschnittstelle zusammenfasst.



COM / DCOM (*Distributed Component Object Model*)

- ➔ Sprachunabhängiges Komponentenmodell
- ➔ COM-Schnittstelle:
 - binäres API einer Komponente (Array von Funktionszeigern)
 - identifiziert durch GUID (*Globally Unique Identifier*)
 - identisch mit UUID von DCE-RPC
 - beschrieben in DCOM IDL
 - unveränderbar, ggf. muß neue Schnittstelle erzeugt werden
- ➔ COM-Objekt:
 - Instanz einer Komponente
 - implementiert eine oder mehrere COM-Schnittstellen



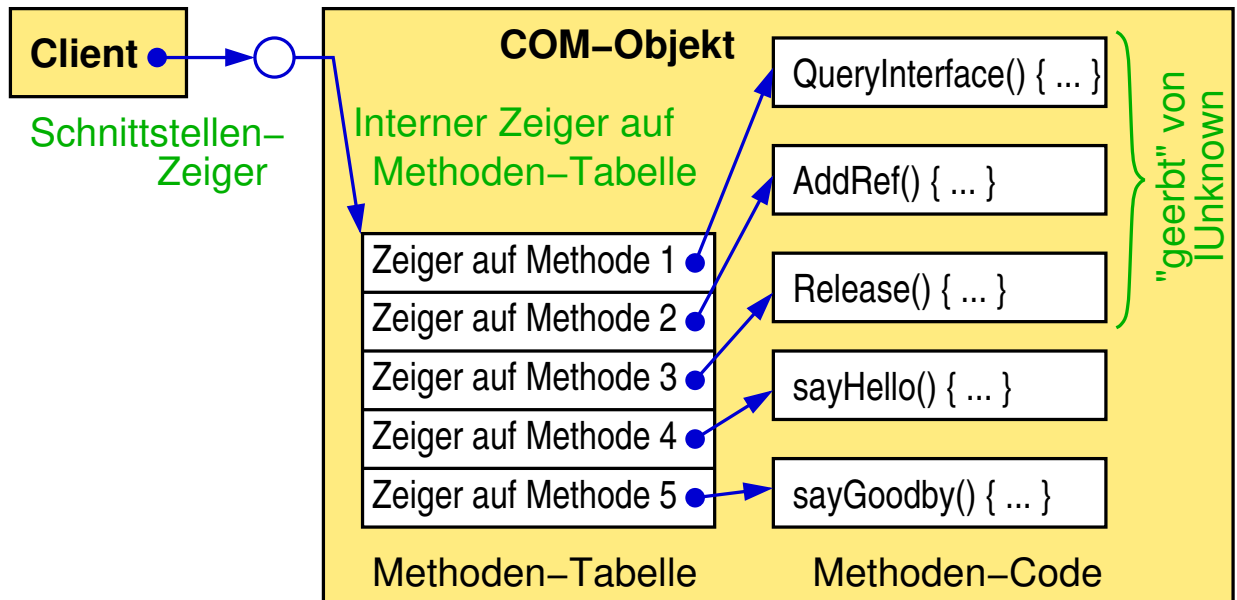
COM / DCOM (*Distributed Component Object Model*) ...

- ➔ Die Schnittstelle `IUnknown`:
 - muß von jeder COM-Schnittstelle geerbt werden
 - definiert die Methoden
 - `QueryInterface()`: liefert zu einer über die GUID gegebene Schnittstelle den Zeiger auf diese Schnittstelle zurück
 - bzw. Fehler, falls Schnittstelle nicht unterstützt wird
 - `AddRef()` / `Release()`: Referenz-Zähler
- ➔ Für jede COM-Komponente muß eine *Factory* realisiert werden
 - u.a. Methode `CreateInstance()` zur Erzeugung neuer Komponenten-Instanzen



COM / DCOM (*Distributed Component Object Model*) ...

➔ Aufbau einer COM-Schnittstelle:



COM / DCOM (*Distributed Component Object Model*) ...

- ➔ COM-Server: "Verpackungseinheit" von COM-Komponenten
 - DLL, ausführbare Datei (*.exe), Java-Klasse, ...
 - enthält Code einer / mehrerer Komponenten, *Factories*, Registrierungs- und Verwaltungsinformation
- ➔ COM-Server kann lokalisiert sein:
 - im lokalen Prozeß (realisiert als DLL)
 - in eigenem Prozeß auf dem lokalen Rechner (COM-Server ist ausführbares Programm)
 - in eigenem Prozeß auf entferntem Rechner (bei DCOM)
 - auch *Remote*-Instantiierung möglich
- ➔ DCOM realisiert Zugriffs- und Ortstransparenz



Unterschiede zw. COM-Objekten und üblichen Objekten

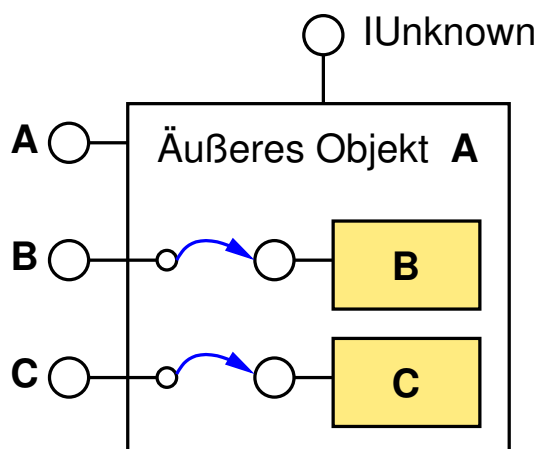
- ➔ In COM existiert keine Objekt-ID
 - ➔ COM-Client erhält Referenz auf eine (zustandslose) Schnittstelle, nicht auf ein Objekt
 - ➔ Client kann sich nicht mit bestimmter Instanz einer Komponente verbinden (Problem, falls Instanz Zustand hat)
 - ➔ Abhilfe durch COM *Monikers*
- ➔ In COM gibt es keine Mehrfach-Vererbung von Schnittstellen
 - ➔ stattdessen: Komponente bietet mehrere Schnittstellen
- ➔ Wiederverwendung von Komponenten durch
 - ➔ *Containment/Delegation*: Weitergabe von Aufrufen
 - ➔ *Aggregation*: `QueryInterface()` gibt Schnittstellen der inneren Objekte nach außen bekannt

8.3 Microsoft .Net und DCOM ...



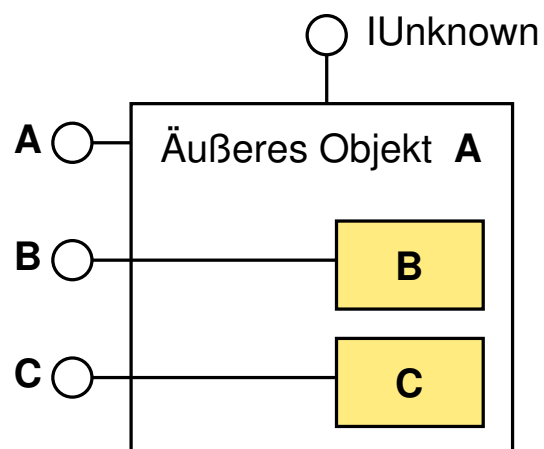
Containment/Delegation und Aggregation

Containment/Delegation:



`QueryInterface()` gibt Schnittstellen des äußeren Objekts zurück. Implementierung von A gibt Aufrufe an B bzw. C weiter.

Aggregation:



`QueryInterface()` gibt Schnittstelle A des äußeren und Schnittstellen B und C der inneren Objekte zurück.



➔ Literatur:

- ➔ Hammerschall, Kap. 10
- ➔ Orfali / Harkey, Kap. 15
- ➔ Hofmann / Jobst / Schabenberger, Kap. 3 und 4
- ➔ <http://www.microsoft.com/com/>