SRH Hochschule
Heidelberg

SRH Hochschule Heidelberg

Fachbereich Informatik

Dipl. Inf. (FH) Johannes Hoppe

SRH Hochschule
Heidelberg

MSP – WCF Vorlesung

25.03.2009

90 Minuten, 8 Teilnehmer

Dipl. Inf. (FH) Johannes Hoppe

SRH Hochschule Heidelberg

Vorlesungsinhalt

Ziel des praktischen Teils:

Erwerbung von Grundkenntnissen in der Anwendungsentwicklung mit der Windows Communication Foundation

Vorraussetzungen

- C# 3.0 / .NET 3.0 Grundkenntisse
- Microsoft Visual Studio **2005 / 2008**

SRH Hochschule Heidelberg

Einführung

Windows Communication Foundation (WCF):

- Laufzeitumgebung
- erlaubt es .NET CLR Typen für eine verteilte Kommunikation zu veröffentlichen (d.h. als Service zu agieren)
- andere Services wie Applikations-eigene .NET CLR Typen zu konsumieren

➔ **SOAP**

SRH Hochschule Heidelberg

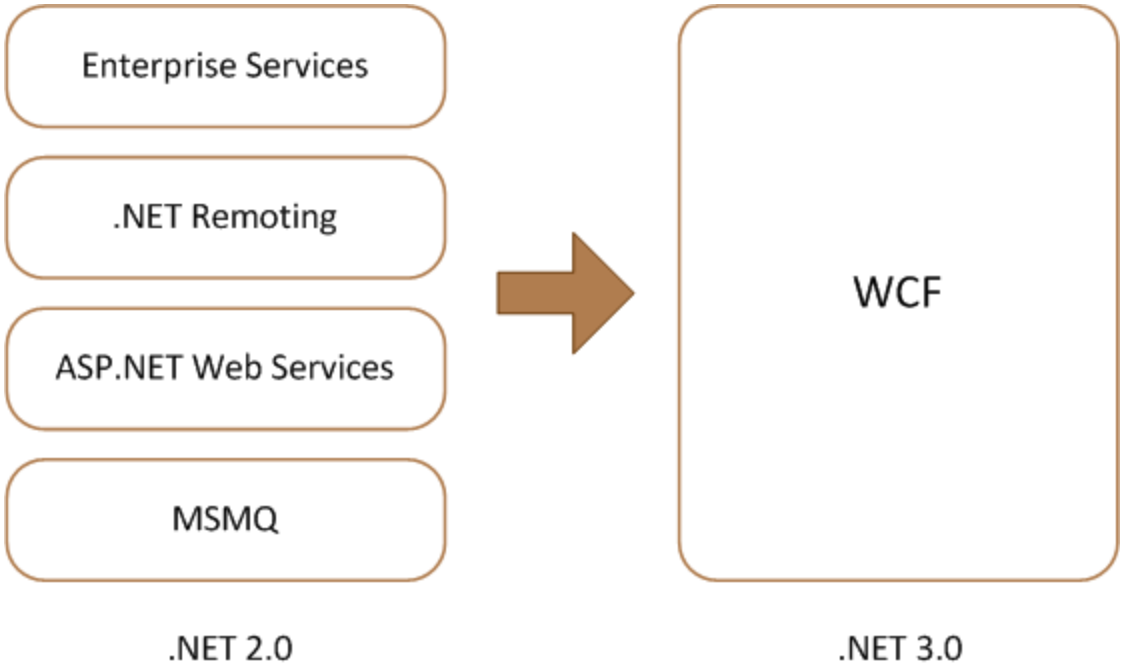
Einführung

Windows Communication Foundation (WCF):

- Vorgänger:
  - Enterprise Services“ (Namespace: System.EnterpriseServices),
  - „.NET Remoting“ (Namespace: SystemRuntime.Remoting)
  - ASP.NET Web Services (Namespace: System.Web.Services)
  - MSMQ (Microsoft Message Queuing – Namespace: System.Messaging)
- unterstützt eine Vielzahl von Protokollen und bleibt so zu älteren Systemen abwärtskompatibel!

SRH Hochschule Heidelberg

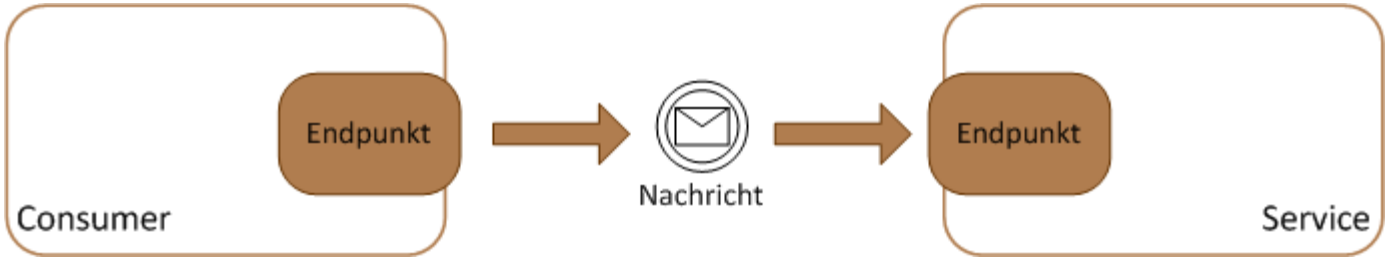
Einführung



SRH Hochschule Heidelberg

Endpunkte

- das Prinzip des Endpunkts
- Endpunkt: „Anfang oder das Ende eines Nachrichtenaustauschs zwischen zwei verteilten Anwendungen“



SRH Hochschule Heidelberg

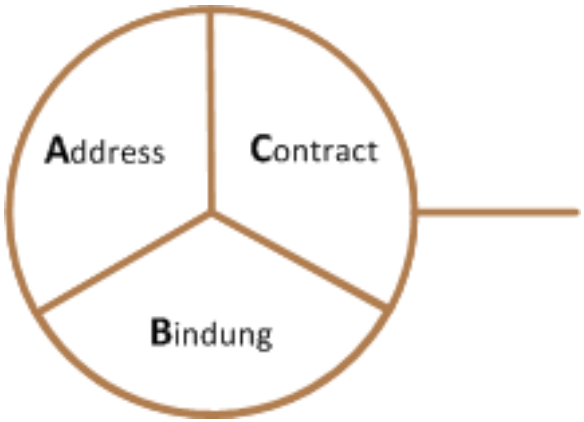
Endpunkte

- Endpunkt beschreibt drei grundlegende Aspekte
  1. den Ort
  2. die Art und
  3. den inhaltlichen Rahmen der Kommunikation

A – Address (Adresse)

B – Binding (Anbindung)

C – Contract (Vertrag)

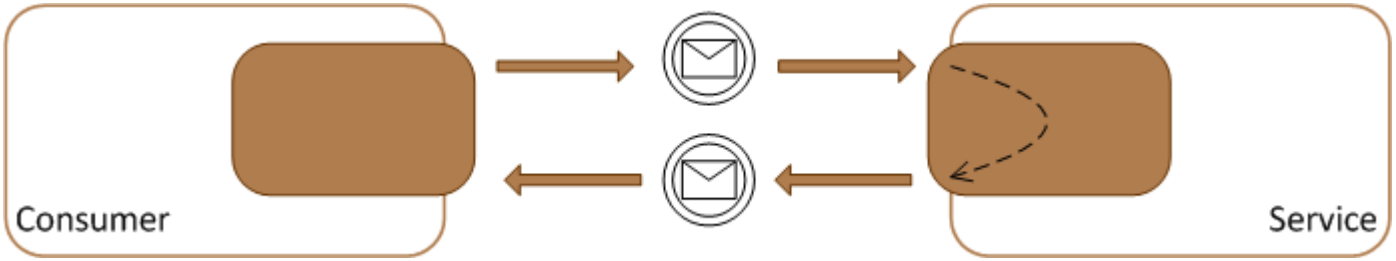


SRH Hochschule Heidelberg

Kommunikationsmuster

- Auch Message Exchange Patterns, „MEP“s
  1. Request-Reply
  2. Einweg
  3. Duplex

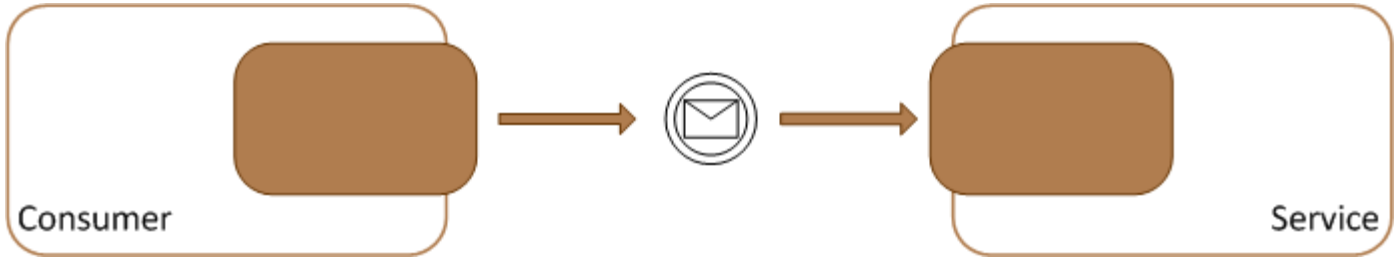
Request-Reply



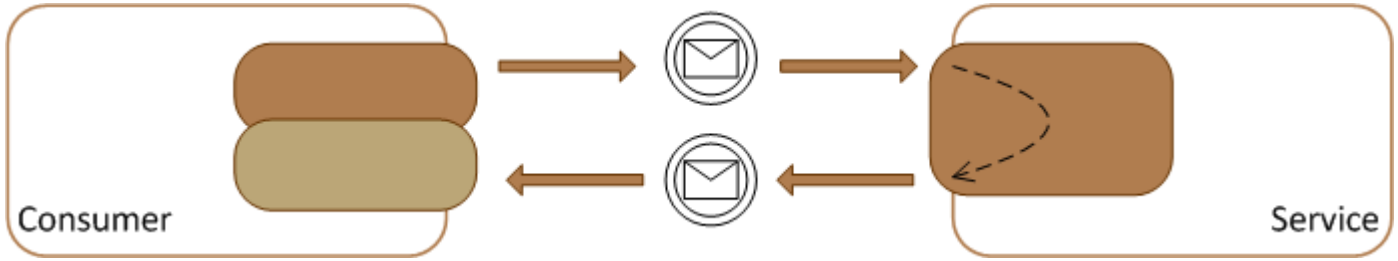
SRH Hochschule Heidelberg

Kommunikationsmuster

Einweg



Duplex



SRH Hochschule Heidelberg

Hosting und Instanzen

Hosting:

- innerhalb einer eigenen Anwendung (Self-Hosting)
- Webserver „IIS“ / Service Host in einer ASP.NET Anwendung
- IIS-Dienst „Windows Activation Service“ (WAS)

SRH Hochschule Heidelberg

Instanzen

Per-Session Service:

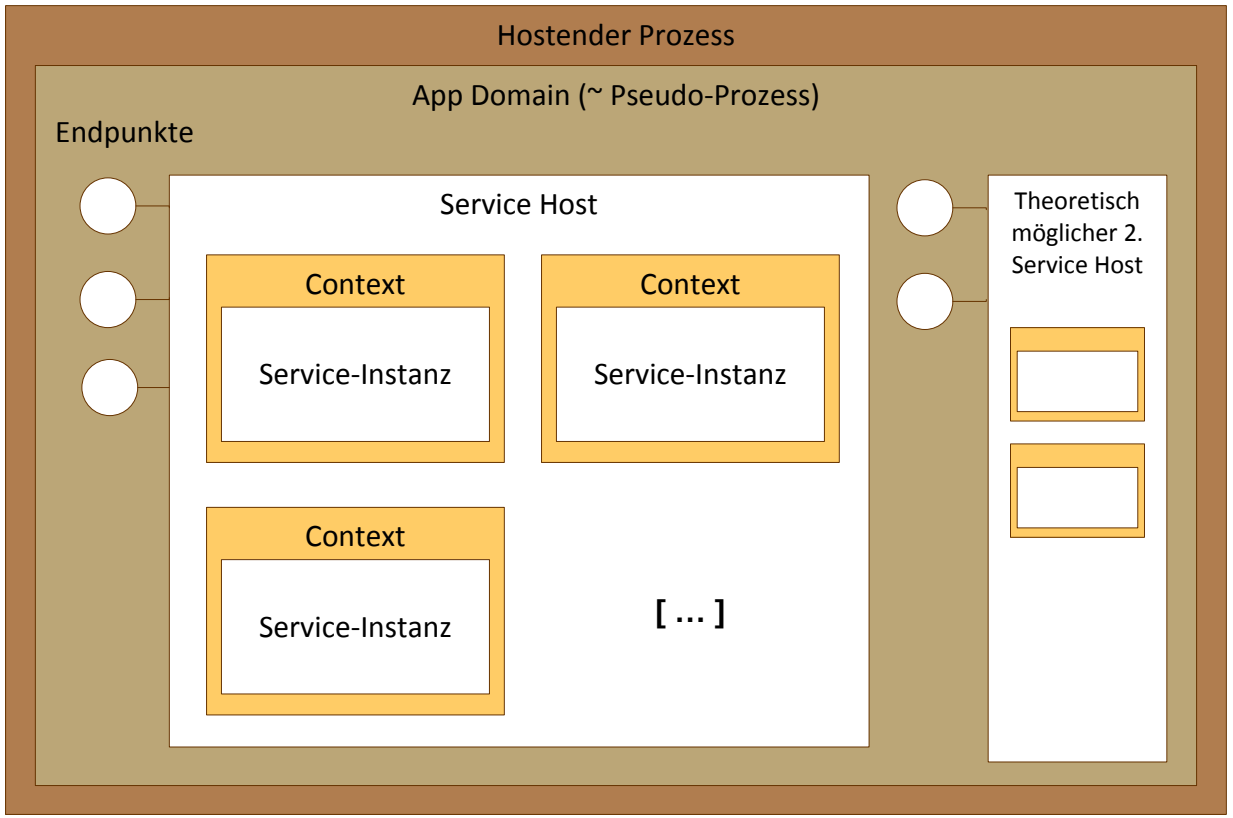
- Standardeinstellungen von WCF
- Client und Server treten in eine Session (Sitzung)
- Jede Sitzung verbleibt bis zum zu ihrem Ende im Arbeitsspeicher
- Pro Sitzung gibt es eine spezielle Service-Instanz die sich in einem eigenen Context (Gültigkeitsbereich) befindet.

```
[ServiceBehavior(InstanceContextMode =
InstanceContextMode.PerSession)]
```

SRH Hochschule Heidelberg

Instanzen

Per-Session Service:



SRH Hochschule Heidelberg

Instanzen

Per-Call Service:

- auf Sitzungen verzichten.
- Statt der Beibehaltung der Daten im Arbeitsspeicher wird für jeden Methodenaufruf eine Service-Instanz erzeugt
- Der Arbeitsspeicher wird aufgeräumt, alle normalen Member-Variablen gehen wieder verloren
- Consumer merkt nichts von der verworfenen Instanz: Service Host lässt alle Verbindungen über einen Proxy bestehe

  ➔ **Peformance-Empfehlung**

```
[ServiceBehavior(InstanceContextMode =
InstanceContextMode.PerCall)]
```

SRH Hochschule Heidelberg

Instanzen

Per-Call Service:

- auf Sitzungen verzichten.
- Statt der Beibehaltung der Daten im Arbeitsspeicher wird für jeden Methodenaufruf eine Service-Instanz erzeugt
- Der Arbeitsspeicher wird aufgeräumt, alle normalen Member-Variablen gehen wieder verloren
- Consumer merkt nichts von der verworfenen Instanz: Service Host lässt alle Verbindungen über einen Proxy bestehe

  ➔ **Peformance-Empfehlung**

```
[ServiceBehavior(InstanceContextMode =
InstanceContextMode.PerSession)]
```

SRH Hochschule Heidelberg

Instanzen

Per-Call Service:

- parallel ablaufende Per-Call Services:

```
[ServiceBehavior(ConcurrencyMode =
ConcurrencyMode.Multiple)]
```

- Vorsicht: Zugriff auf gemeinsam genutzte Variablen
- Locks notwendig

➔ **Deadlocks können entstehen**

SRH Hochschule Heidelberg

Instanzen

Singleton Service:

- exakt eine Service-Instanz, die für alle Anfragen verwendet wird
- Bleibt bis zur Beendung des Service Host aktiv
- **sind prinzipiell nicht skalierbar**
- sollten nur in begründeten Fällen eingesetzt werden!
- z.B. Windows Dienst ➔ Tray Applikation (muss nicht skalieren)

```
[ServiceBehavior(InstanceContextMode =
InstanceContextMode.Single)]
```

SRH Hochschule Heidelberg

Los geht's!

```
[ServiceContract]
public interface IHelloWorld
{
    [OperationContract]
    string ReturnMessage(string text);
}

class Hell
{
    public string ReturnMessage(string text)
    {
        return "Sie haben geschrieben: " + text;
    }
```

Wir bauen uns einen Webservice mit Consumer!

SRH Hochschule
Heidelberg

ENDE

Vielen Dank für eure Aufmerksamkeit!