Windows Communication Foundation (WCF)

Overview

Content

- Overview
- Basics
- Experiences

Overview

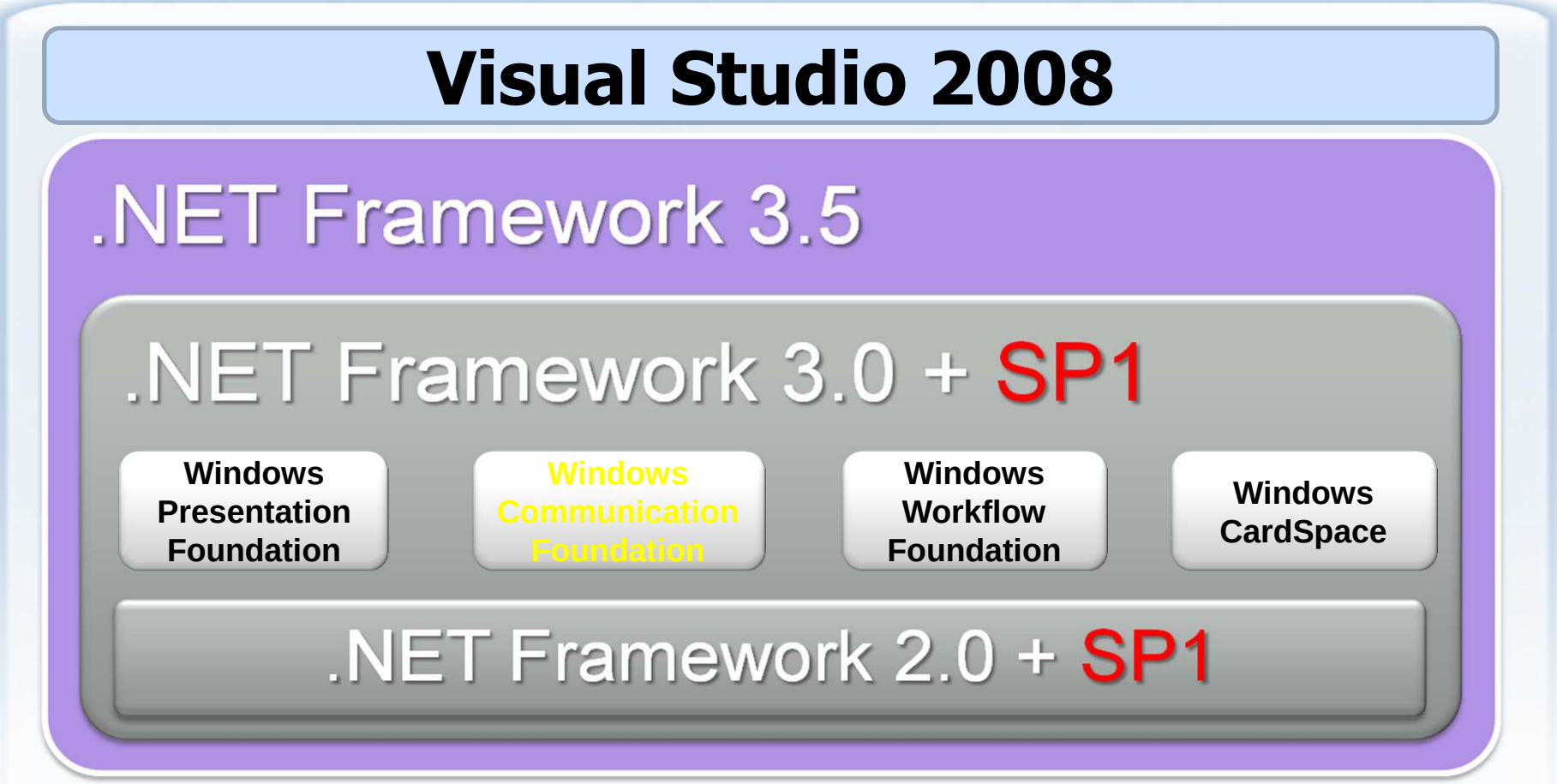
Definition WCF

- „… Windows Communication Foundation (previous Codename „Indigo“) is a set of .technologies for creating and running of distributed systems. …“
- „… It is a new class of communication infrastructure, which was created for web service architeture. …“
- „Service orientierte architecture“ for distributed applications offering a service, which consumed by clients.
- Services are „loosely coupled“ (not connection orientated)
- WCF implements several Web Service Standards (Adressing, Security, etc.)

- Based on Web Services Standards
  - SOAP 1.2 http://www.w3.org/TR/soap
  - WSDL 1.1 http://www.w3.org/TR/wsdl
  - XML Schema http://www.w3.org/TR/xmlschema-0/

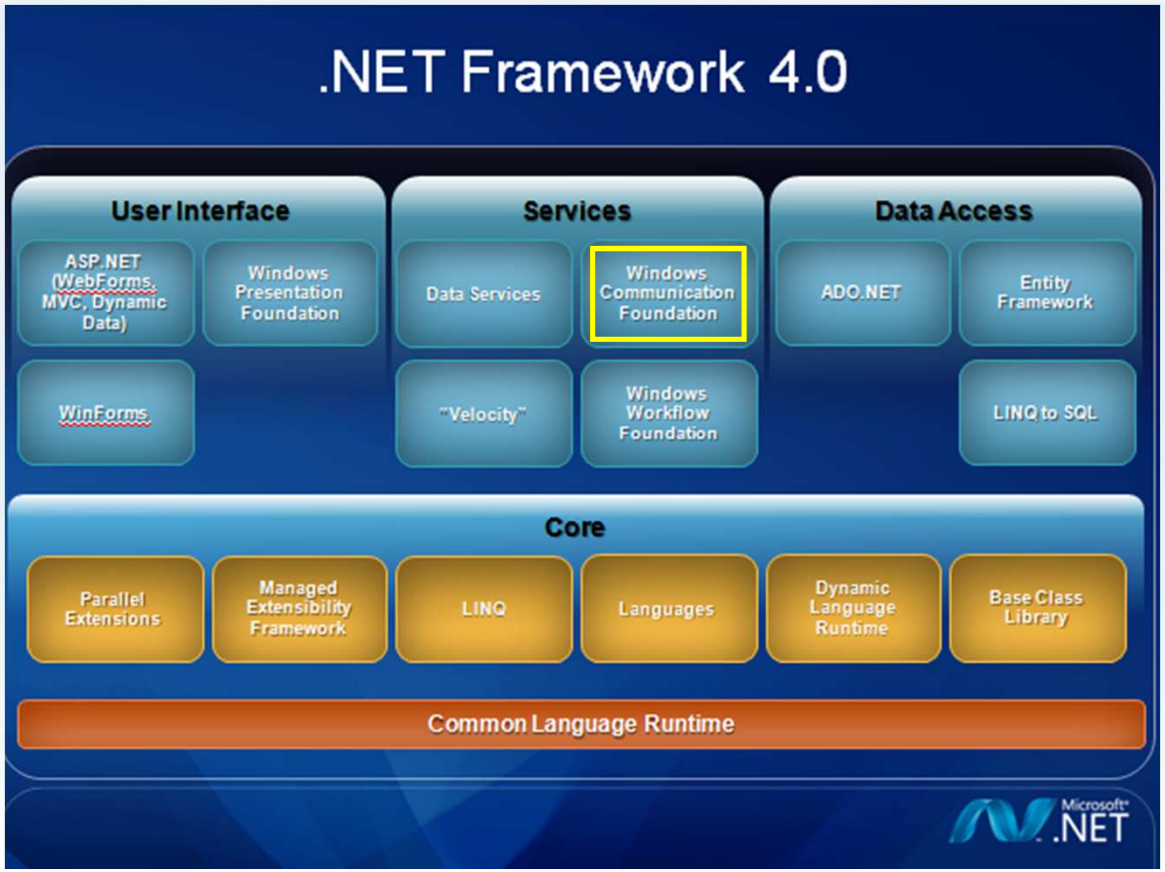
Overview

Architecture of .NET Framework 3.5



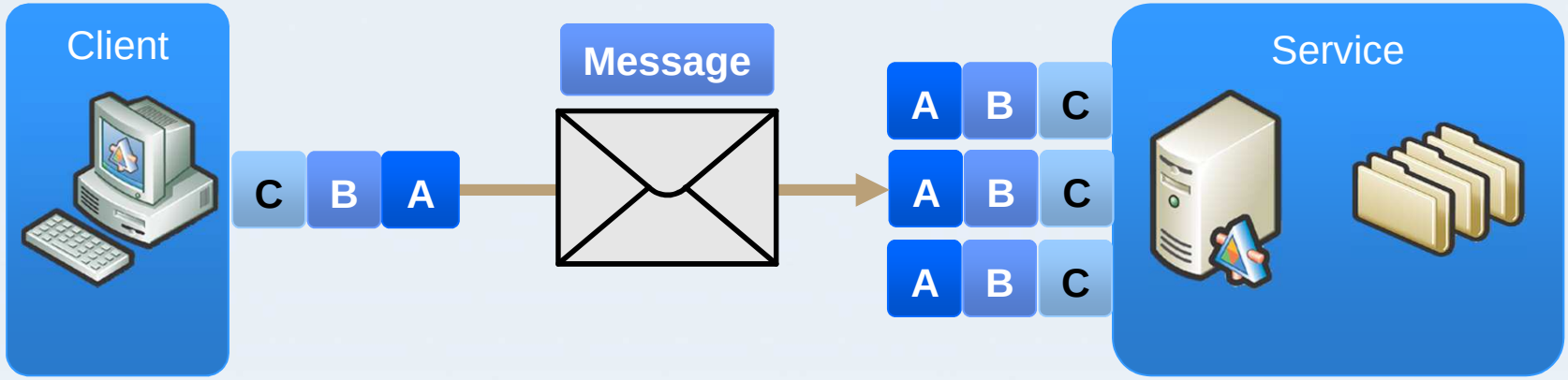
Overview

Architecture .NET Framework 4.0



Basics

Bird‘s Eye view concept

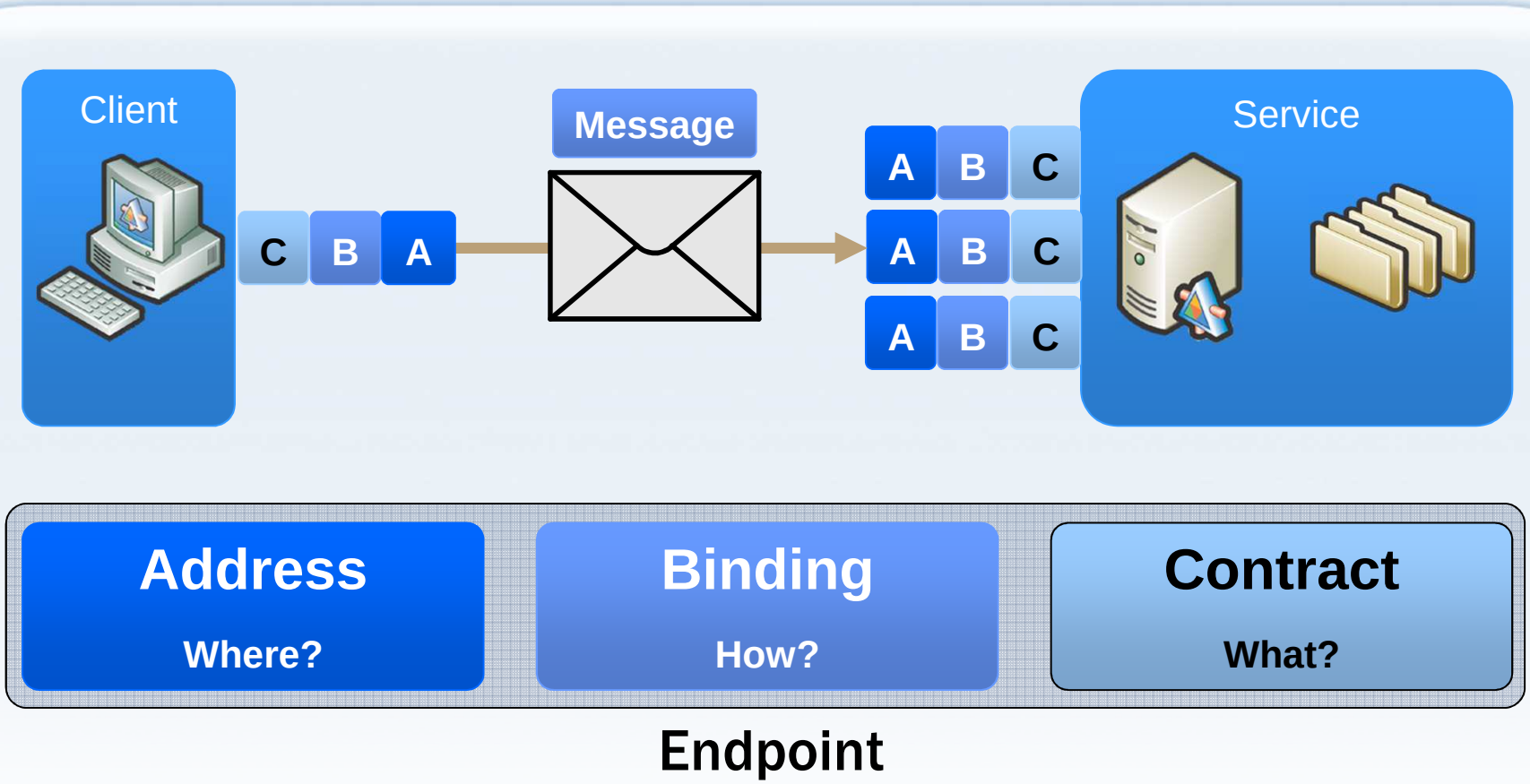


Definition WCF Service

Service class – Endpoint – Host environment

Basics

Basic principle Endpoint (ABC)



Behavioral Contracts

Namespaces

- System.ServiceModel
  - Service contract
  - Operation contract
- System.Runtime.Serialization
  - Data contract

Behavioral Contracts

Message Exchange Patterns

MEPs describes the used protocol between message exchange. During instance it will be defined if the comsumer receives return values or not.

There are three different types:

- Request/Response
  - Most common used => classic Web Service
- OneWay
  - Marked with OperationContract-Attribute IsOneWay=true
  - With FaultContract not useable
- Duplex
  - Marked with ServiceContract-Attribute CallbackContract=typeOf(ISomeInterface)
  - All methods must be marked with OneWay
  - Asynchrone operation, qualified for long processes
  - Autonomus sending from Service to Consumer possible (without call from consumer)
  - Attention with Firewall and NAT (solved in .NET 4.0 with routing service)
  - Threading => Callback runs in an other context than Consumer Call

Behavioral Contracts

Fundamental terms

- Service contract
  Definition which operations are provided from service
- Operation contract
  Definition how operation is structured
- Data contract
  Defines the structure of sending data
- Message contract
  Defines the message header and its message structure
- (Fault contract)
  Defines the return value in case of error

Behavioral Contracts

Code Example

```
[ServiceContract()]
public interface IBDEData
{
      …
      [OperationContract()]
      int BDERequestSoftwareRepository(Guid id);
      [OperationContract()]
      int BDEAddSoftwareRepository(CSoftwareRepository softwareRepository);
      [OperationContract()]
      int BDERemoveSoftwareRepository(CSoftwareRepository softwareRepository);
      …
}
```

! WCF ist Opt-In !

Structural Contracts

Data Contract

Data Contracts are contract conventions about format and structure of content data in messages which are exchanged between service and consumer (comparable with XSDs in XML world).

Has to be marked with DataContract-Attribute

- Can be allocated only to enums, structs and classes
- Not inheritable
  - In case of inherited Data Contract the attribute has to be allocated again

Structural Contracts

Code example

```
[ServiceContract()]
public interface IBDEData
{
      …
      [OperationContract()]
      int BDERequestSoftwareRepository(Guid id);
      [OperationContract()]
      int BDEAddSoftwareRepository(CSoftwareRepository softwareRepository);
      [OperationContract()]
      int BDERemoveSoftwareRepository(CSoftwareRepository softwareRepository);
      …
}
```

```
[DataContract]
public class CSoftwareRepository
{
    …
    [DataMember]
    public string Description { get; set; }
    [DataMember]
    public string Version { get; set; }
    …
}
```

! WCF ist Opt-In !

Structural Contracts

Message Contract

Message Contracts controls the complete SOAP Message and not only structure and body like the Data Contract.

- Controls how SOAP Message Body is structured and will be serialized.
- Support for Custom Headers
- Real World: Standard WSDL makes sense for long term applications – do not define an own standard!

Structural Contracts

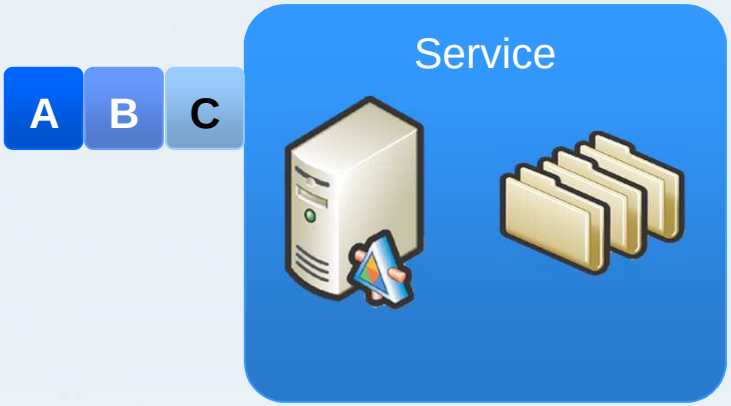
Versioning Data Contract

- Added Members to Data Contract
  - Missing values will be supplemented with Default (Null, 0)
- Removed Members from Data Contract
  - No problem, if DataContract Attribute is set to IsRequired=false (Default-Setting)
- Roundtripping (missing Members by return values)
  - Can be solved with IExtensibleDataObject Interface

Exposing Service

Contract

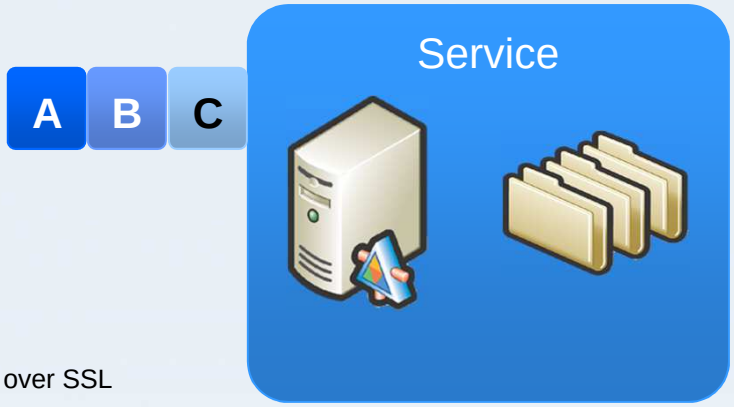
Identifies operations which are provided from service.
Typically the reference to the interface name.



Exposing Service

Binding

Defines the access mode to service (protocol). Wcf differ between 13 types.



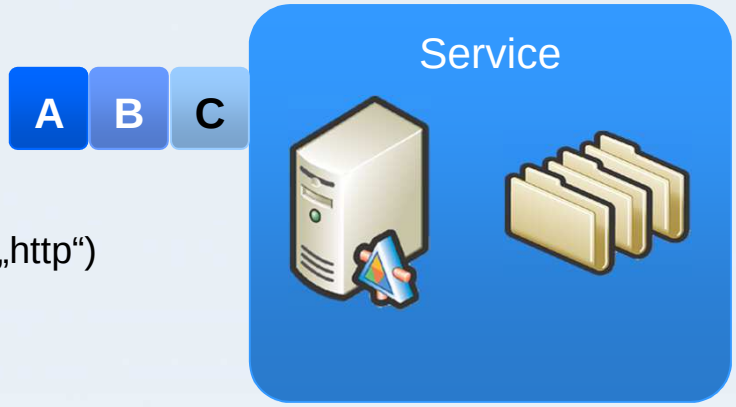
This are the most common types:

- basicHttpBinding
  - Hypertext Transfer Protocol or Hypertext Transfer Protocol over SSL
- netTcpBinding
  - With Transmission Control Protocol transmitted binary encoded SOAP message

Exposing Service

Address

Adress for an endpoint (in URL-format)



- Scheme: Top Level Part of address (for example „http“)
- Machine: identifies the machine (for example „www.contonso.com“)
- Port: optional port number
- Path: Path to service

Exposing Service

Configuration

The specification has to be done imperative with code or declarative with configuration.

- Real World: Do it always by declarative way!!!

```
<configuration>
  <systemserviceModel>
    <services>
      <service name="MyNamespace.OrderService">
        <endpoint address=http://localhost:8000/OrderService/
                  binding="wsHttpBinding"
                  contract="MyNamespace.IOrderService" />
      </service>
    </services>
    …
```

Exposing Service

Publication of Meta Data

In case of request Meta Data can be transmitted from service to client. The transmission is solved in WSDL-format (Web Service Description Language).

- Publication by endpoint
  - Metadata Exchange Point (IMetadataExchange) or with attribute HttpGetEnabled = true
  - HTTP-Get request with ?wsdl
- Examples
  - http://soap.amazon.com/schemas2/AmazonWebServices.wsdl
  - http://www.weather.gov/forecasts/xml/DWMLgen/wsdl/ndfdXML.wsdl

Deploying Service

Host Environment

Der WCF-Service can be hosted on following locations:

- Web Server
  - IIS Server
  - WAS (Windows Process Activation Service), with Service Host Factory
  - Apache (Moonlight 2.0, part of Mono Project)
- Managed Application
  - Console Application
  - Windows Service
  - Windows Forms
  - Windows Presentation Forms

Deploying Services

Instance Service by Host

```
Type serviceType = typeof(CSomeService);
ServiceHost host = new ServiceHost(serviceType);
host.Open();
```

Consuming Services

Proxy Class

Access from Client-side to service can be solved with Proxy-Class.

There are several ways for creating a Proxy-Class

- Tool „svcutil“
- Visual Studio with function „Add Service Reference“
- (Dynamic mit ChannelFactory)

Advantage „svcutil“ in comparism with Built-In-Function of Visual Studio

- Full control about Proxy Class generation type
- Useful by automatic build process integration

Consuming Services

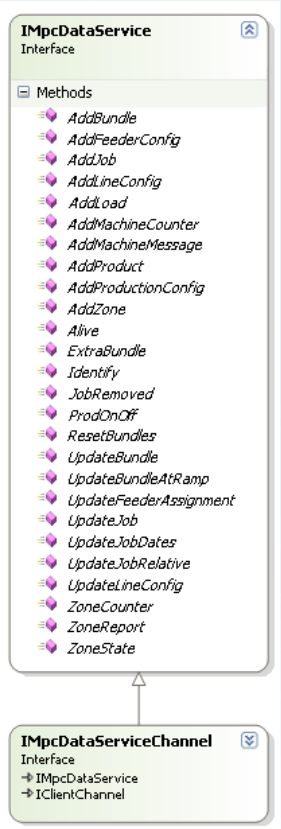
Instance Service by Client-Side

Code-Fragment of a Proxy-Klasse (automatisch generated Code):

```
[System.Diagnostics.DebuggerStepThroughAttribute()]
[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "3.0.0.0")]
public partial class MpcDataServiceClient :
System.ServiceModel.ClientBase<Client.WcfMpcService.IMpcDataService>,
Client.WcfMpcService.IMpcDataService
{
  …
}
```

Instance ChannelFactory on Client-Side:

```
MpcDataService _mpcDataServiceClient = new MpcDataServiceClient();
_mpcDataServiceClient.Alive(_aliveDC);
```



Consuming Services

Configuration Client Endpoint

Deklaratives Code-Beispiel:

```
<system.serviceModel>
  <behaviors>
    <serviceBehaviors>
      <behavior name="Trottling">
        <serviceThrottling maxConcurrentCalls="64" maxConcurrentSessions="64„
                           maxConcurrentInstances="250" />
      </behavior>
    </serviceBehaviors>
  </behaviors>
  <bindings>
    <netTcpBinding>
      <binding name="BDEStreamData" transferMode="Streamed„
               maxReceivedMessageSize="67108864" />
    </netTcpBinding>
    </bindings>
    <client>
      <endpoint address="net.tcp://localhost:4002/BDEStreamData„
                binding="netTcpBinding„
                bindingConfiguration="BDEStreamData"
                contract="Client.WcfNgrService.IBDEStreamData"
                name="BDEStreamData" />
    </client>
    …
</system.serviceModel>
```

Instrumentation

Tools for Analysis

For analysis and debugging purposes the framework provides a lot of tools:

- End-To-End Service Tracing
  - Actiovation of source System.ServiceModel in system.diagnostics and definition Tracing Level
  - Exposition in Service Trace Viewer
- Monitor für Service-Health
- Log Messages
- Implementation of Message Inspectors

Security

Overview

WCF-Architektur contains all nescessary functions for safe-guarding security purposes.

- Transport-Level Security (Endpoint-To-Endpoint)
  - Authendification Sender/Service
  - Message integrity
  - Protection of Message
  - Reply detection
- Message-Level Security (End-To-End)
  - Independ Security of Transport Layer
  - Different parts of message can be secured
  - With Credential Type Attribut and Secure Token solved
- User Level Security
  - Authenticate / Authorize / Impersonate of Clients

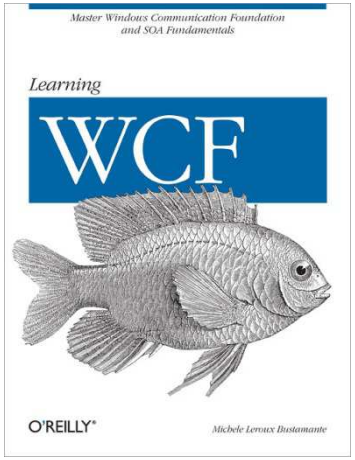
Experiences

Summary

- All-embracing Framework
- All requirements are solved
- Runs good and stable
- Flexible
- High-performance (depent from design)
- Good standard
- Fast integration

References

- Microsoft .NET Framework 3.5 – Windows Communication Foundation Self-Paced Training Kit
  - Microsoft Press
  - ISBN-13: 978-0-7356-2565-5



- Learning WCF (Master Windows Communication Foundation and SOA Fundamentals
  - O'Reilly
  - ISBN-13: 978-0-596-80550-0