# PV178: Programming for .NET Framework
## Windows Communication Foundation

Vojtěch Forejt, forejt@fi.muni.cz
Martin Osovský, osovsky@ics.muni.cz

**Faculty of Informatics** and **Institute of Computer Science**
Masaryk University

April 9, 2009

## Distributed Applications

- Applications composed of several processes.
- Processes usually communicate via a network.

- Java RMI, .NET Remoting, Web services. . .
- Each technology has slightly different purpose.
- The technologies are not compatible.
- Migrating from one technology to another requires a lot of effort.

# Service Oriented Architecture

- Method of development in which systems provide functionality as **loosely coupled services**
    - Service – mechanism providing access to one or more capabilities using a prescribed interface and consistent constraints and policies
- Each service implements one action (get a weather forecast, book a hotel room, get latest blog posts,. . . )
- Services communicate using predefined protocols.

# Service Oriented Architecture

- Short movie...

## .NET Remoting

- Allows creation of object shared by several applications
- Platform specific
- Similar to Java RMI

## Web Services

- Services communicating over standard internet protocols
- Exact behaviour described by standards (SOAP, REST, WSDL, UDDI,. . . )
- Messages are mostly passed as XML files
- Communication between different languages and/or platforms

## Service Oriented Architecture – Diagram
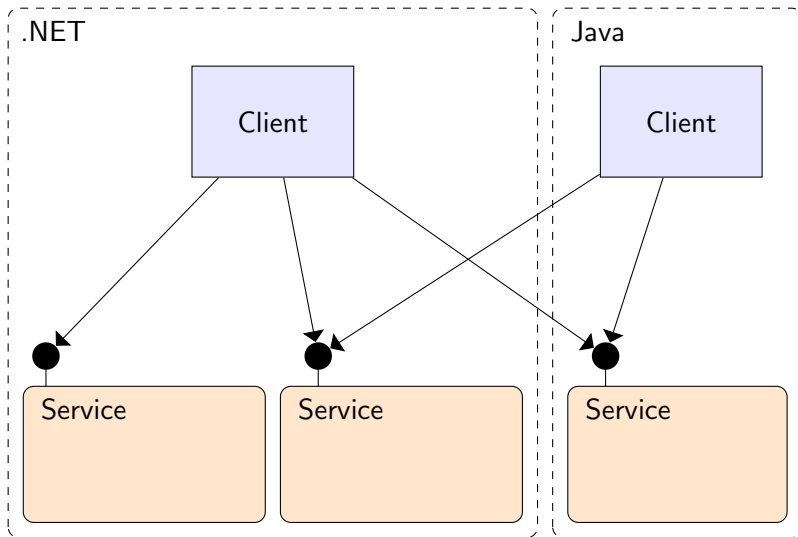
# Windows Communication Foundation

- Framework for building applications that inter-communicate.
- From .NET 3.0, under development in Mono.
- Follows principles of SOA.
- The underlying architecture may not be hard-coded, can be changed using config files.
- Namespace `System.ServiceModel`.

## WCF Service

- Service is a CLR type that exposes a functionality via a set of methods accessible by remote clients
- **Endpoints** – Specify the way in which service communicates with clients
    - **Address** – URI
    - **Binding** – protocols supported
    - **Contract** – functionality supported (methods, types)

## Address

- Says **where** the service is available.
- Address at which the service awaits incomming messages
- `<scheme>://<domain>[:port]/[path]`
    - Examples:
      `http://example.com/ServiceA`
      `net.msmq://localhost`
- Every endpoint must have unique address

## Binding

- Says **how** the service can be accessed
- **Transport protocol** – TCP, HTTP, named pipe, MSMQ,. . .
- **Message encoding** – XML, binary
- **Other protocols** – for security, reliability.

## Binding cont.

- WCF provides classes for most commonly used binding types

- Web services – `BasicHttpBinding`, `WSHttpBinding`, ...
- Binary – `NetTcpBinding`, `NetNamedPipeBinding`,...

## Service contract

- Says **what** the service provides
- Set of operations that the service exposes
- Provided via a class marked with
  `ServiceContractAttribute` and its methods with
  `OperationContractAttribute`.

# Proxy

- Used by clients to communicate with services
- Hide process of serialization from the client.
- Based on the service contract.

## Channels

- Facilitate the communication between proxy and service
- Channel stack – transport channel, message encoding channel, channels for security,. . .

## Service hosting

- Service is made available through a host process
    - Internet Information Services
    - Windows Activation Services
    - Windows service
    - Windows Form
    - Console

## ServiceHost class

- Used to host services not hosted by IIS
- Type of a service is passed in constructor
- Endpoint are created by `AddServiceEndpoint` method, or created from config file.
- `Open` and `Close` methods start and stop listening on the endpoint.

# WCF – Structure
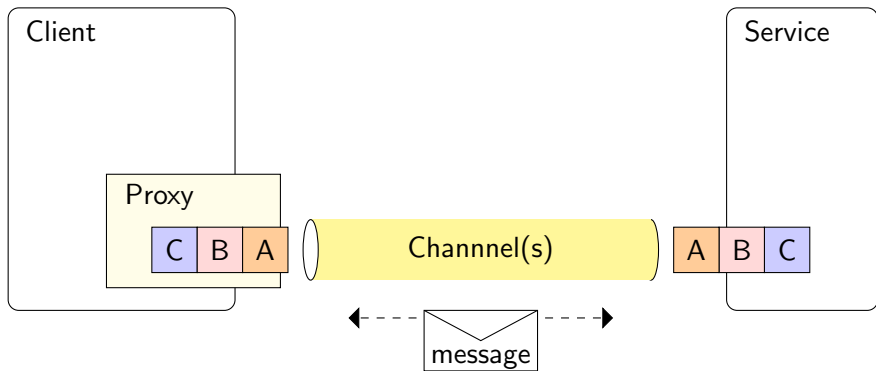
# Example

- SimpleWebService

## Data Contracts

- The methods described by service contract may need to use "complex" types.
- Data contract describes how a complex type is serialized.

- `DataContractAttribute` is applied to the complex type to be serialized.
- `DataMemberAttribute` is applied to the serialized members.

## WCF in Visual Studio

- Rich design time support
  - Editing of config files
  - Proxy generation
  - Service Testing

- Example