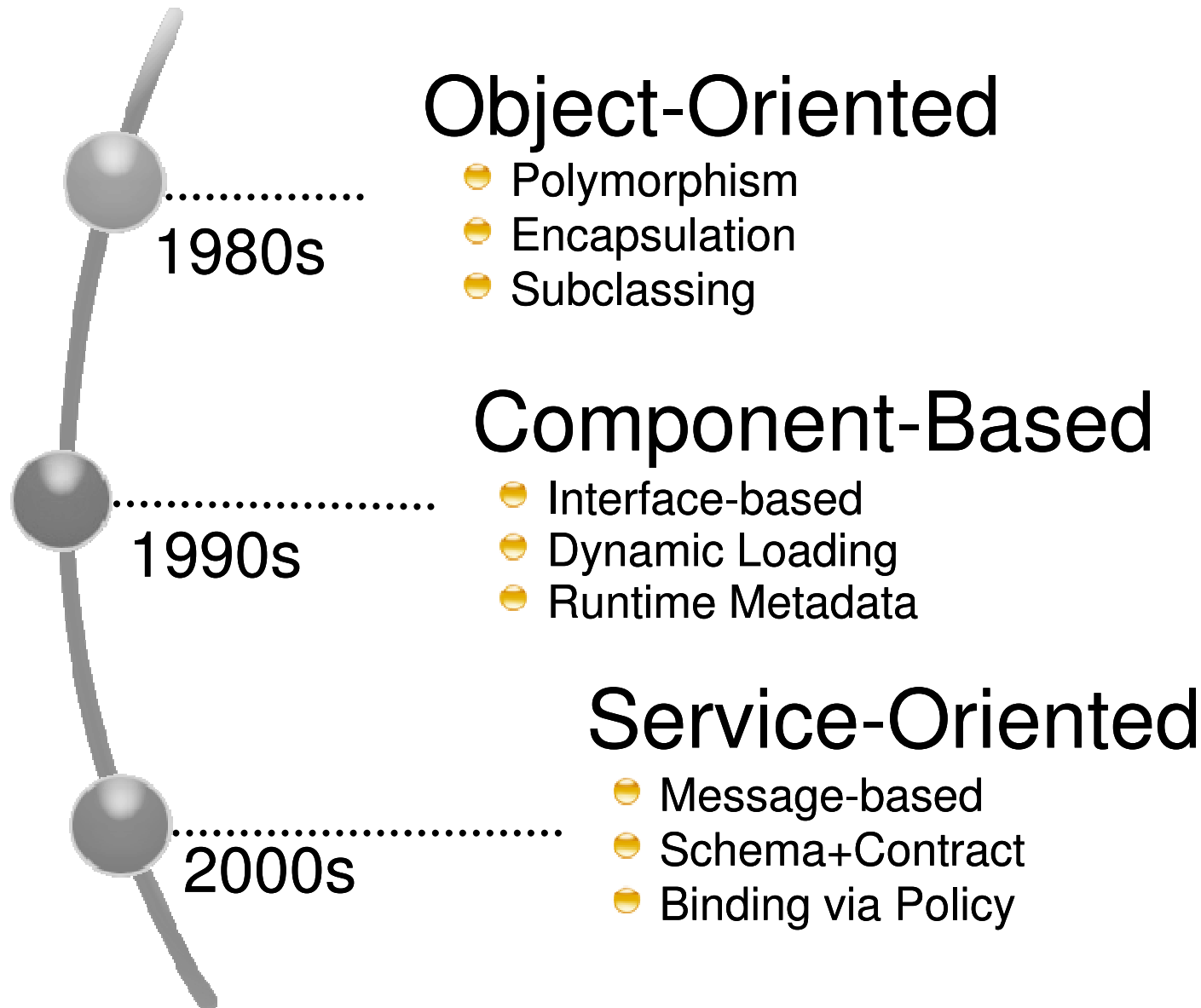# Windows Communication Foundation

# Outline
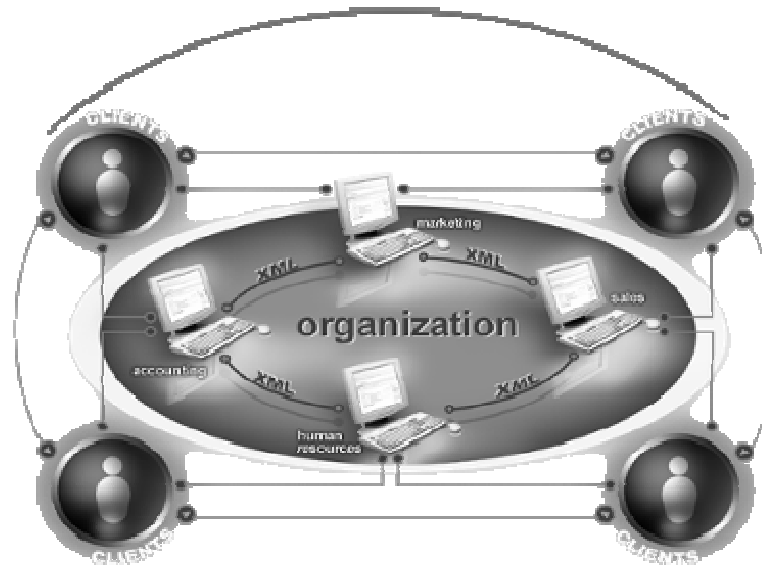
- WCF Contracts
  - Service
  - Data
  - Message
- Bindings
- Security
- Reliability
- Declarative
- Summary

# From Objects to Services

## Object-Oriented
- Polymorphism
- Encapsulation
- Subclassing

**1980s**

## Component-Based
- Interface-based
- Dynamic Loading
- Runtime Metadata

**1990s**

## Service-Oriented
- Message-based
- Schema+Contract
- Binding via Policy

**2000s**

# The Challenge

*Radically Simplifying Distributed Application Development*



**Development of connected systems
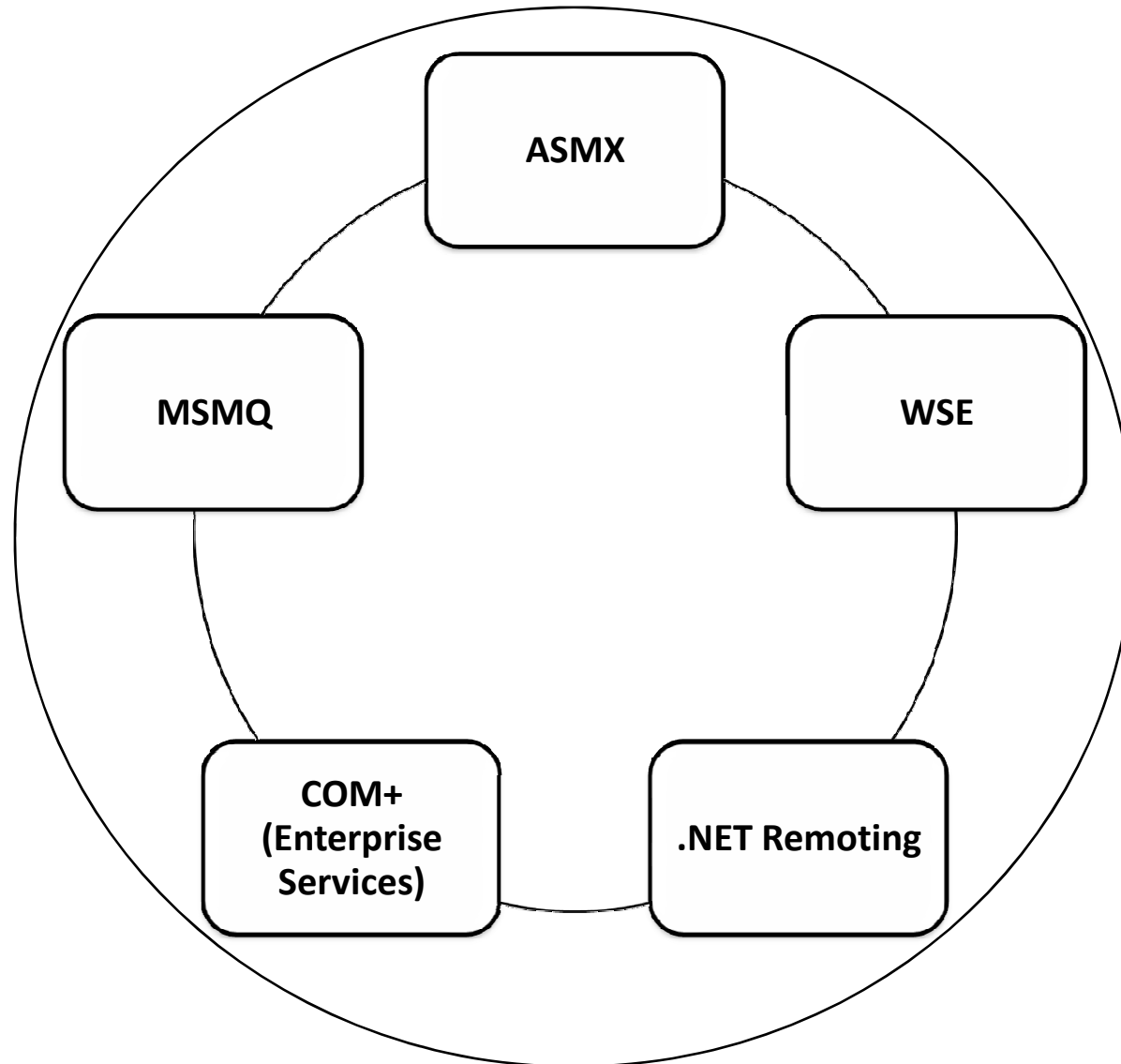remains costly and frustrating**

- **Different programming models for different tasks**
- **Need for security and reliable messaging**
- **Interoperability with applications on other platforms**
- **Productive service-oriented programming model needed**

# Windows Communication Foundation

Unified framework for
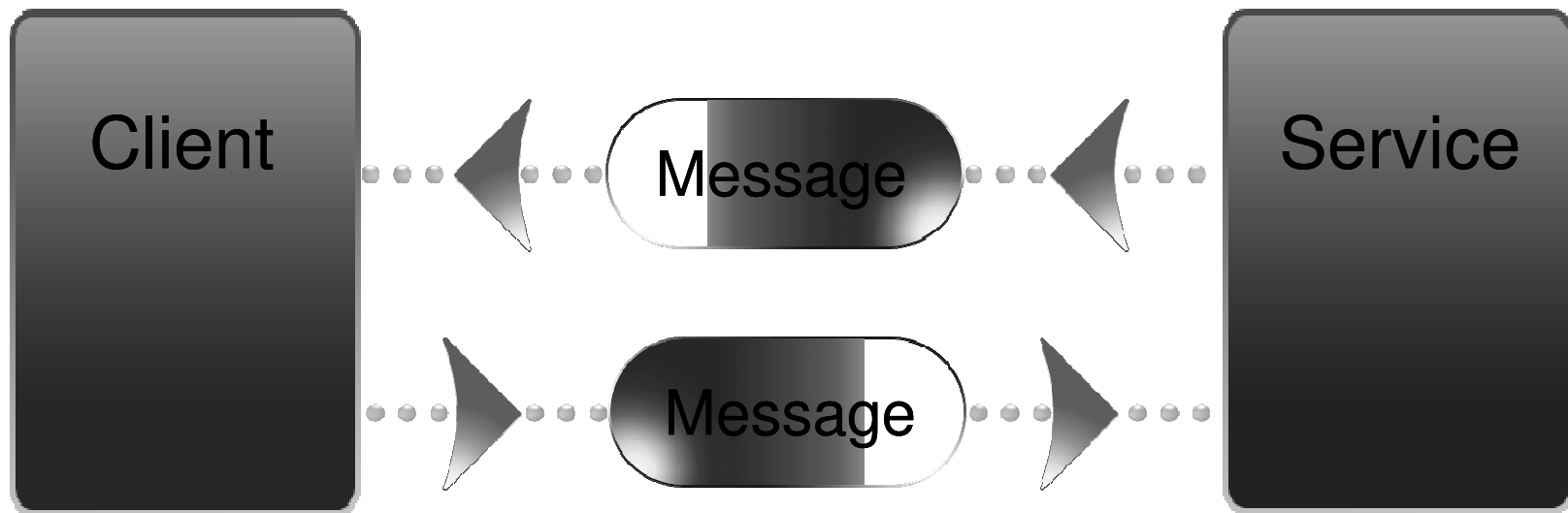
rapidly building

service-oriented applications

WPF

WCF

WF

InfoCard

Microsoft
.net

# What Does WCF Replace?

# UNDERSTANDING WCF PRINCIPLES

# Services and Clients

# Endpoints

Client

Endpoint

Message

Service

Endpoint

Endpoint

# Address, Binding, Contract

Client

Endpoint

C B A

Message

Endpoints

A C

A B C
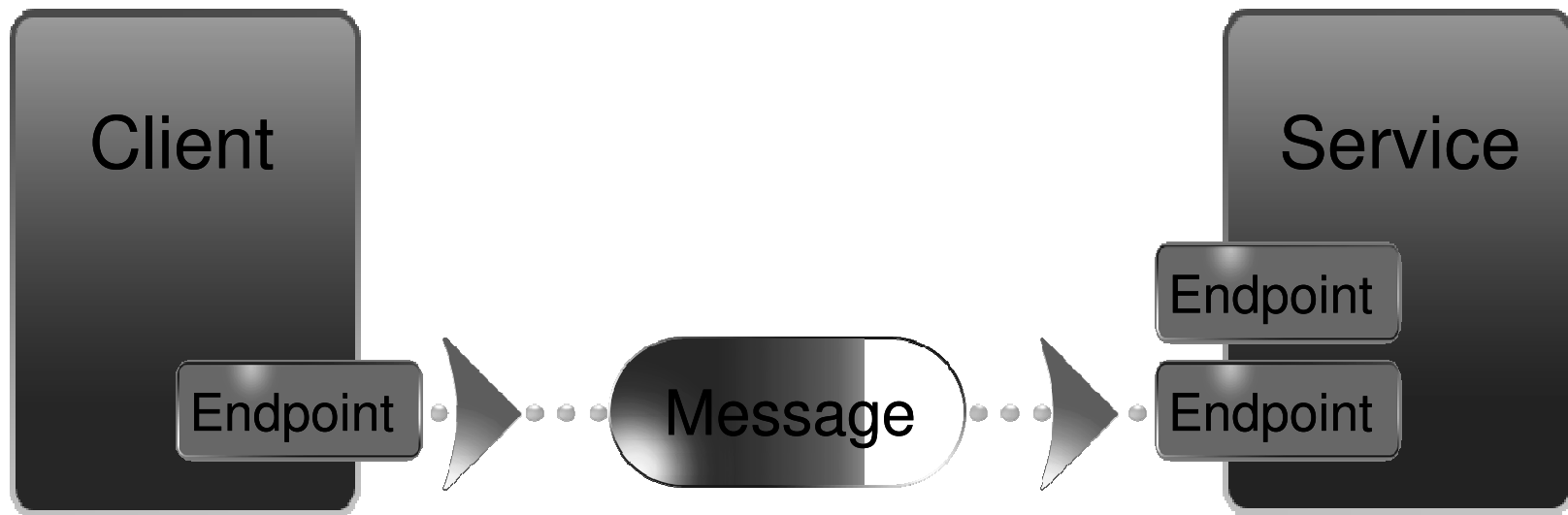
Service

| Address | Binding | Contract |
|---------|---------|----------|
| *(Where)* | *(How)* | *(What)* |
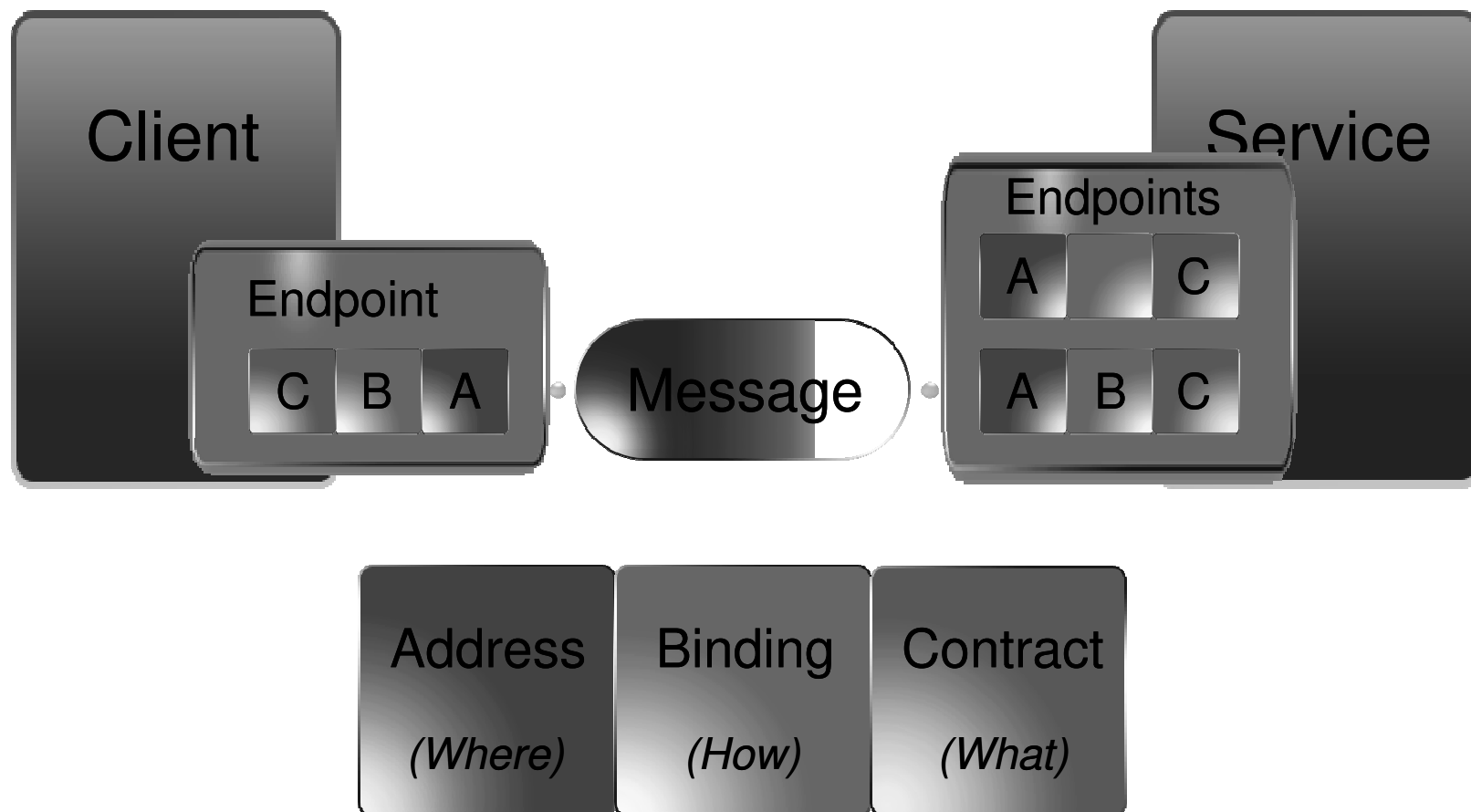
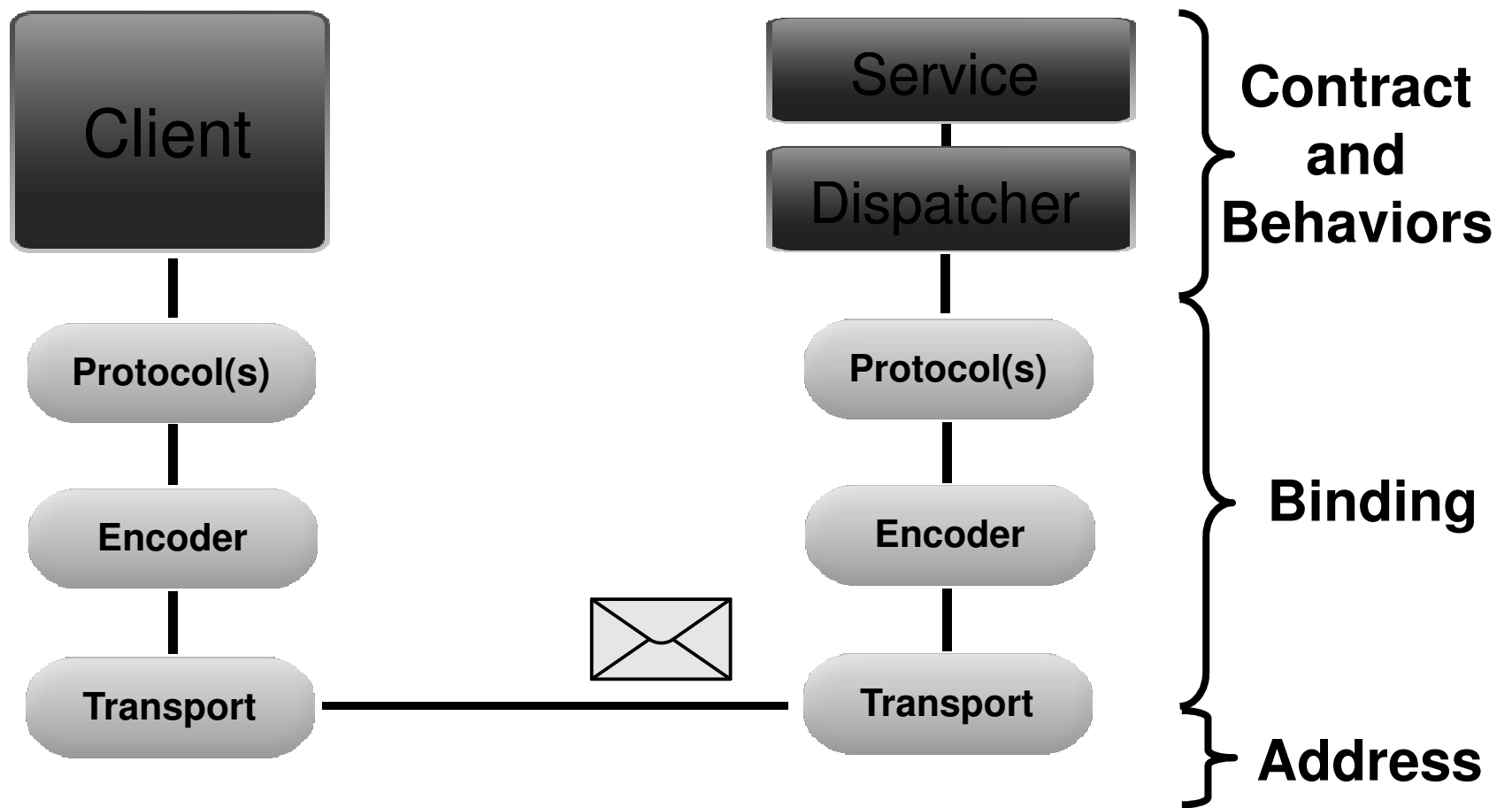# WCF Architecture: Messaging Runtime

The what

# CONTRACTS

# Three Types of Contracts

## Service Contract

Defines Operations, Behaviors and Communication Shape

What does your service do

## Data Contract

Defines Schema and Versioning Strategies

What object data is used

## Message Contract

Allows defining application-specific headers and unwrapped body content

Allows control over the SOAP structure of messages

# Ways to Talk

**One Way**

Client → Service

**Request-Reply**

Client → Service

**Duplex (Dual)**

Client ↔ Service
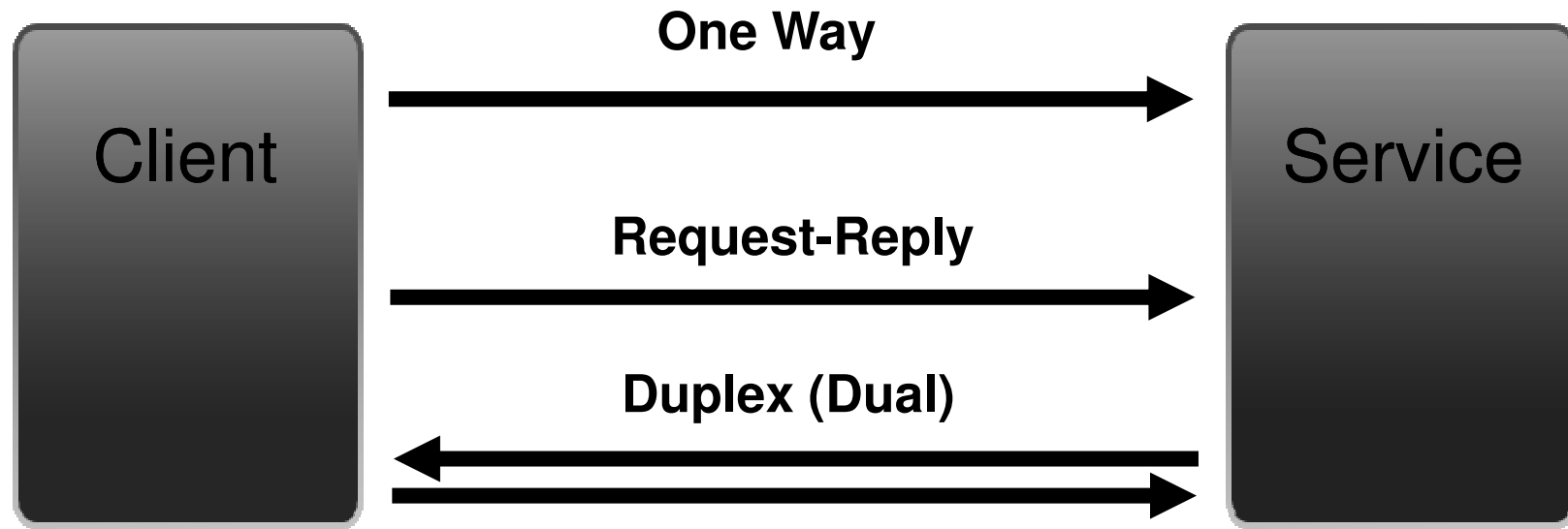
- One Way:
  - Datagram-style delivery
- Request-Reply
  - Immediate Reply on same logical thread
- Duplex
  - Reply "later" and on backchannel (callback-style)

What does your service do?

# SERVICE CONTRACTS

# Service Contract

```csharp
using System.ServiceModel;

[ServiceContract]
public interface ICalculate
{
        [OperationContract]
        double Add( double a, double b);
        [OperationContract]
        double Subtract( double a, double b);
}
```

# Service Contract: OneWay

```
[ServiceContract]
public interface IOneWayCalculator
{
    [OperationContract(IsOneWay=true)]
    void StoreProblem (ComplexProblem p);
}
```

# Service Contract: Duplex Asymmetric

```
[ServiceContract(Session=true,
         CallbackContract=typeof(ICalculatorResults)]
public interface ICalculatorProblems
{
    [OperationContract(IsOneWay=true)]
    void SolveProblem (ComplexProblem p);
}


public interface ICalculatorResults
{
    [OperationContract(IsOneWay=true)]
    void Results(ComplexProblem p);
}
```

# Service Contract: Duplex Symmetric

```
[ServiceContract(Session=true,
          CallbackContract=typeof(IChat)]
public interface IChat
{
    [OperationContract(IsOneWay=true)]
    void Talk(string text);
}
```

What object data needs to flow back and forth?

# DATA CONTRACTS

# Data Contract

```
[DataContract]
public class ComplexNumber
{

    [DataMember]
    public double Real = 0.0D;
    [DataMember]
    public double Imaginary = 0.0D;

    public ComplexNumber(double r, double i)
    {
        this.Real = r;
        this.Imaginary = i;
    }
}
```

Defines the mapping between the type and a SOAP envelope

# MESSAGE CONTRACTS

# Message Contract

```
[MessageContract]
public class ComplexProblem
{
    [MessageHeader]
    public string operation;
    [MessageBody]
    public ComplexNumber n1;
    [MessageBody]
    public ComplexNumber n2;
    [MessageBody]
    public ComplexNumber solution;

    // Constructors…
}
```

How?

# BINDINGS

# Bindings & Binding Elements

**Binding**

HTTP → Text → Security → Reliability → TX →

**Transport**

TCP → HTTP →

MSMQ → IPC →

Custom →

**Encoders**

Text →

Binary →

Custom →

**Protocol**

Security → Reliability →

TX → .NET →

Custom →
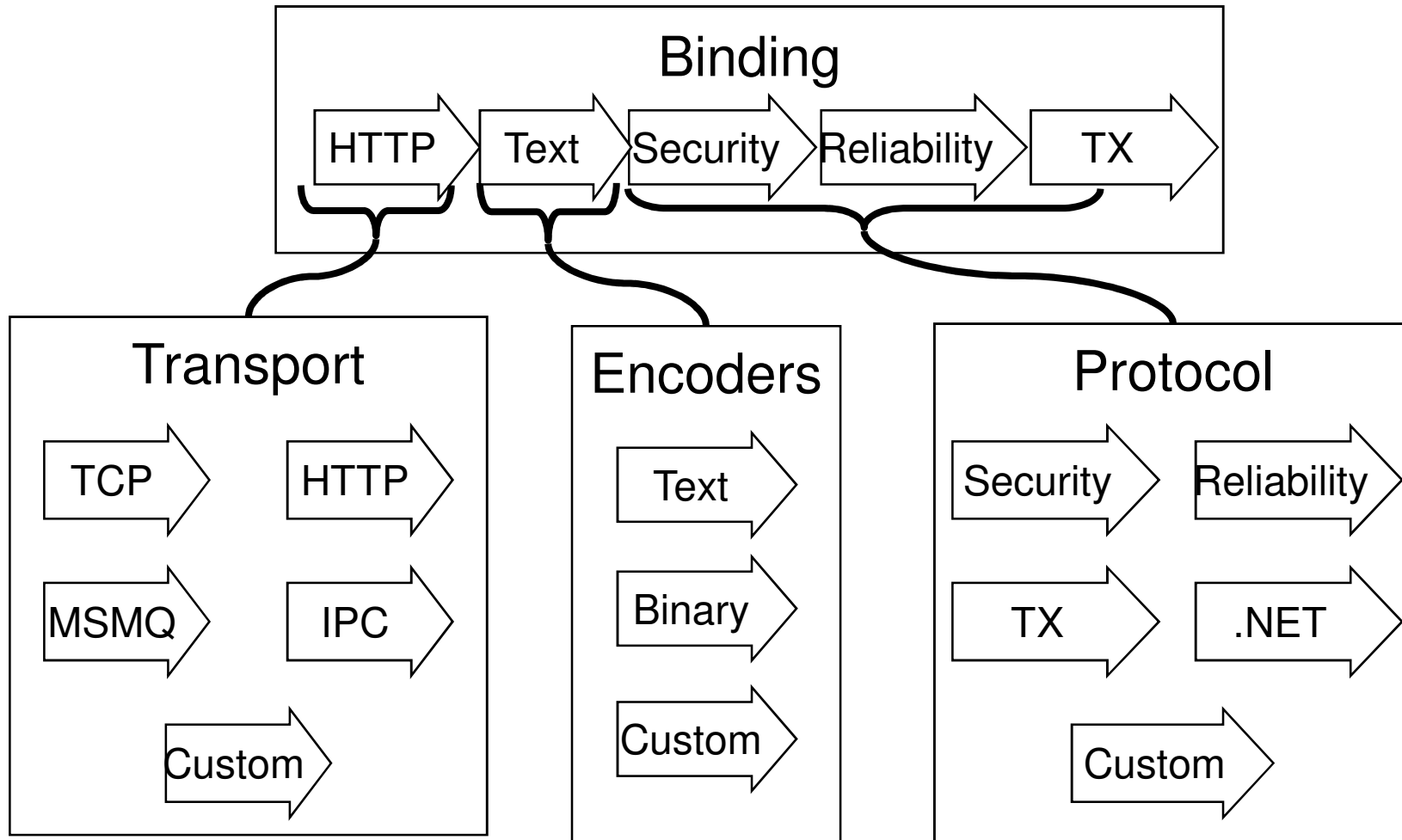
# Standard Bindings

| Binding | Interop | Security | Session | TX | Duplex |
|---|---|---|---|---|---|
| **BasicHttpBinding** | **BP 1.1** | **N, T** | **N** | **N** | n/a |
| **WSHttpBinding** | **WS** | **M, T, X** | **N, T, RS** | **N, Yes** | n/a |
| **WSDualHttpBinding** | **WS** | **M** | **RS** | **N, Yes** | **Yes** |
| **WSFederationBinding** | **Federation** | **M** | **N, RS** | **N, Yes** | **No** |
| **NetTcpBinding** | **.NET** | **T, M** | **T ,RS** | **N, Yes** | **Yes** |
| **NetNamedPipeBinding** | **.NET** | **T** | **T, N** | **N, Yes** | **Yes** |
| **NetPeerTcpBinding** | **Peer** | **T** | **N** | **N** | **Yes** |
| **NetMsmqBinding** | **.NET** | **T, M, X** | **N** | **N, Yes** | **No** |
| **MsmqIntegrationBinding** | **MSMQ** | **T** | **N** | **N, Yes** | n/a |

**N = None | T = Transport | M = Message | B = Both | RS = Reliable Sessions**

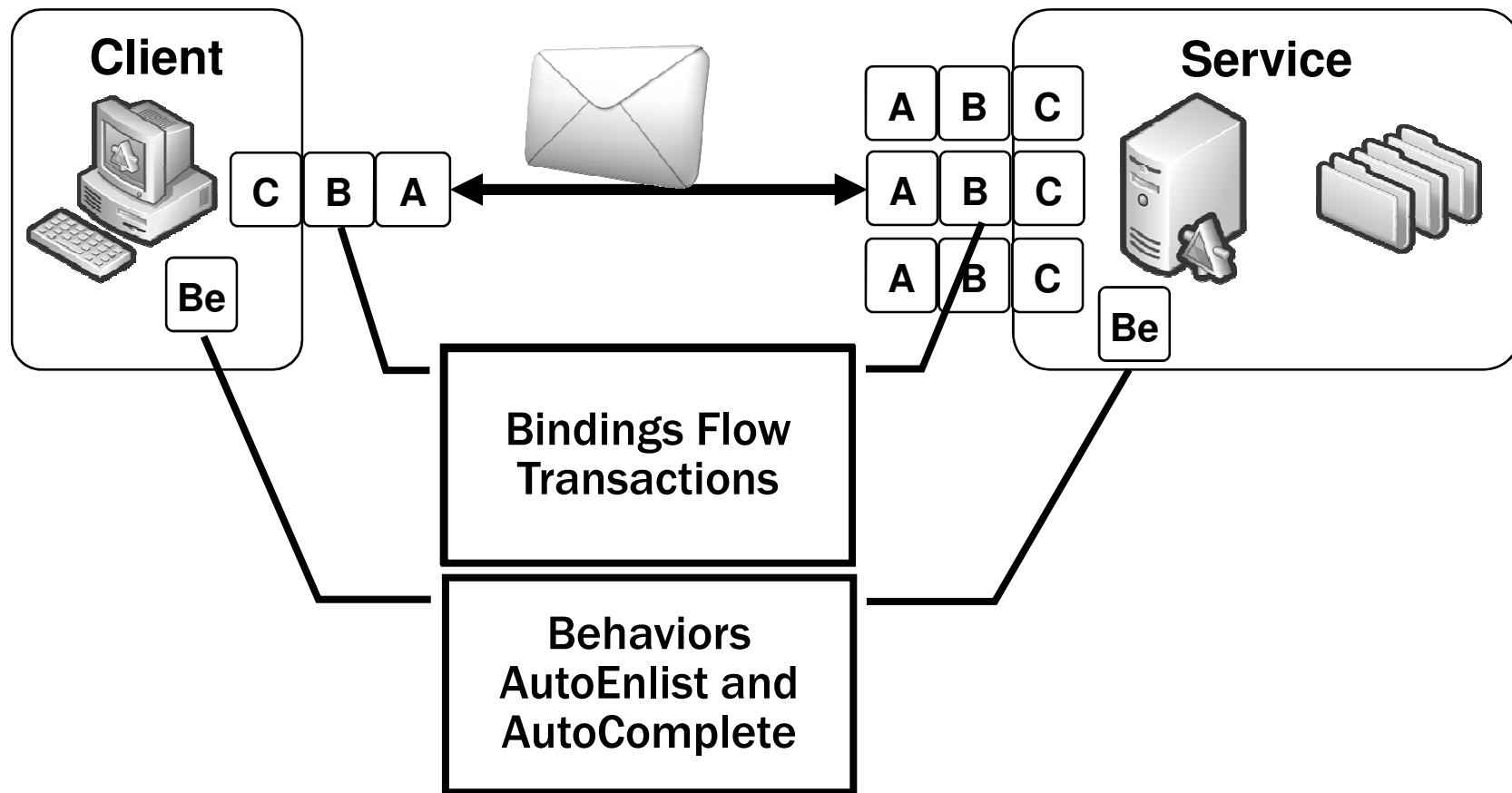# Bindings & Behaviors: Security
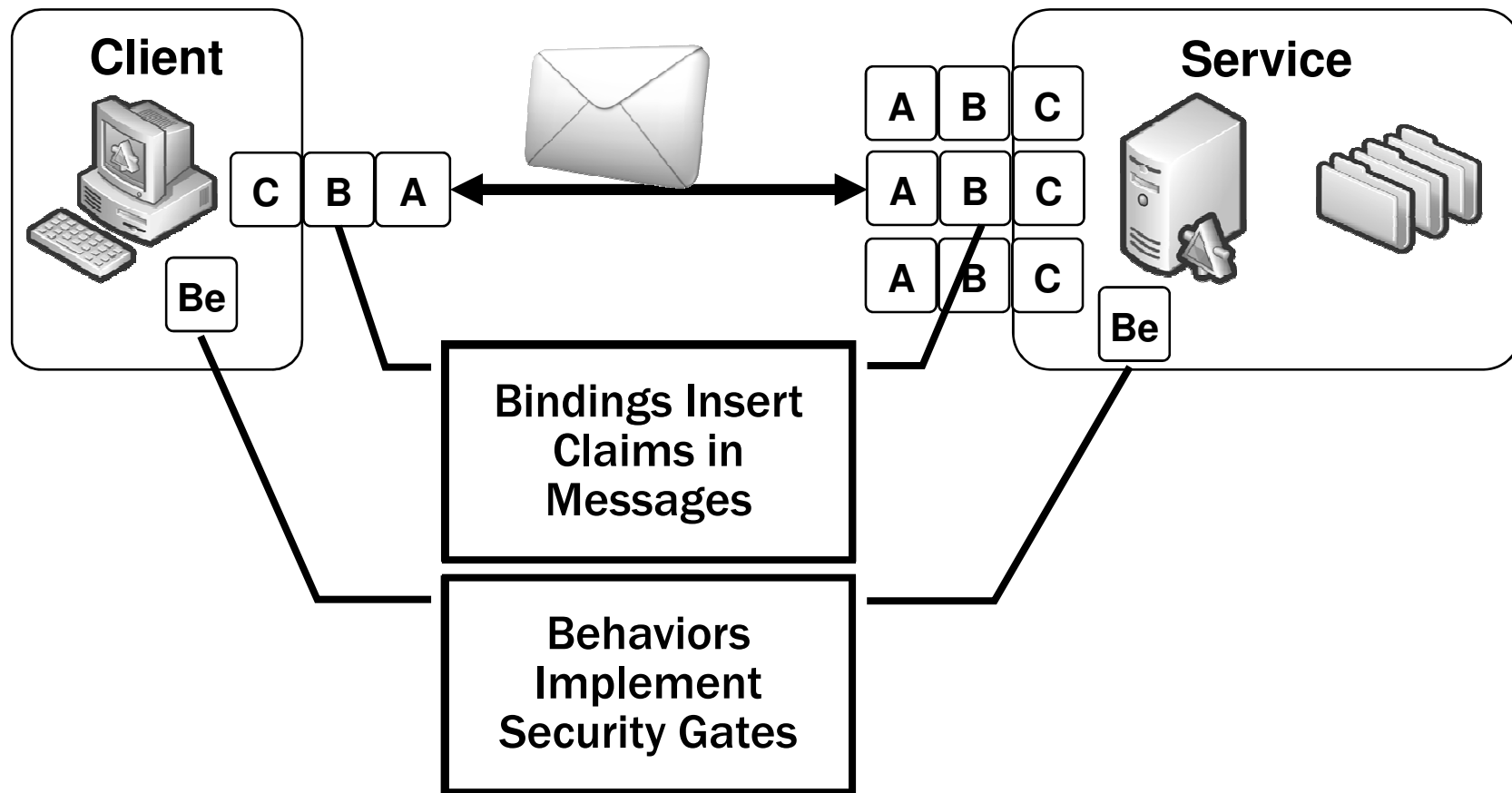
# Feature Overview
**Security**

- Claims based end-to-end security
  - Secure end-to-end message exchanges
  - Secure access to resources
  - Record resource access requests
- X509, Username/Password, Kerberos, SAML, custom credentials
- Message security
  - Confidentiality and integrity
  - Transport or message level
- Access to resources
  - Authentication and authorization

# Bindings & Behaviors: Transactions



Client

Be

C B A

A B C
A B C
A B C

Be

Service

**Bindings Flow Transactions**

**Behaviors AutoEnlist and AutoComplete**

# Bindings & Behaviors: Reliable Sessions



**Client**

C B A

**Service**

A B C
A B C
A B C

Bindings provide Session and Guarantees
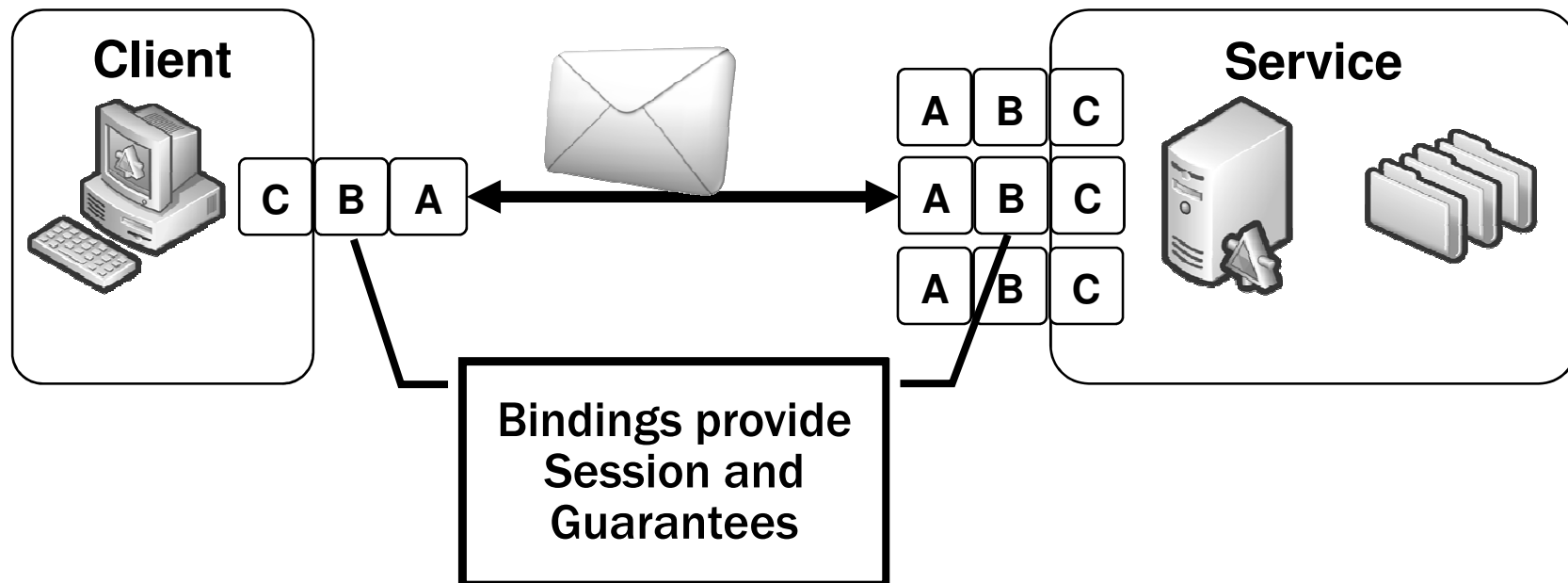
# Feature Overview
**_Reliability and Transactions_**

- End-to-end Reliable messaging
  - In-order guarantees
  - Exactly once guarantees
- Transport-Independent Sessions
  - Integration with ASP.NET Sessions in IIS-Hosted compatibility mode
- Transactions
  - Guaranteed atomic success or failure across services

# Code vs. Config

# Defining Endpoints

```xml
<?xml version="1.0" encoding="utf-8" ?>
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <system.serviceModel>
    <services>
      <service serviceType="CalculatorService">
        <endpoint address="Calculator"
                  bindingSectionName="basicProfileBinding"
                  contractType="ICalculator" />
      </service>
    </services>
  </system.serviceModel>
</configuration>
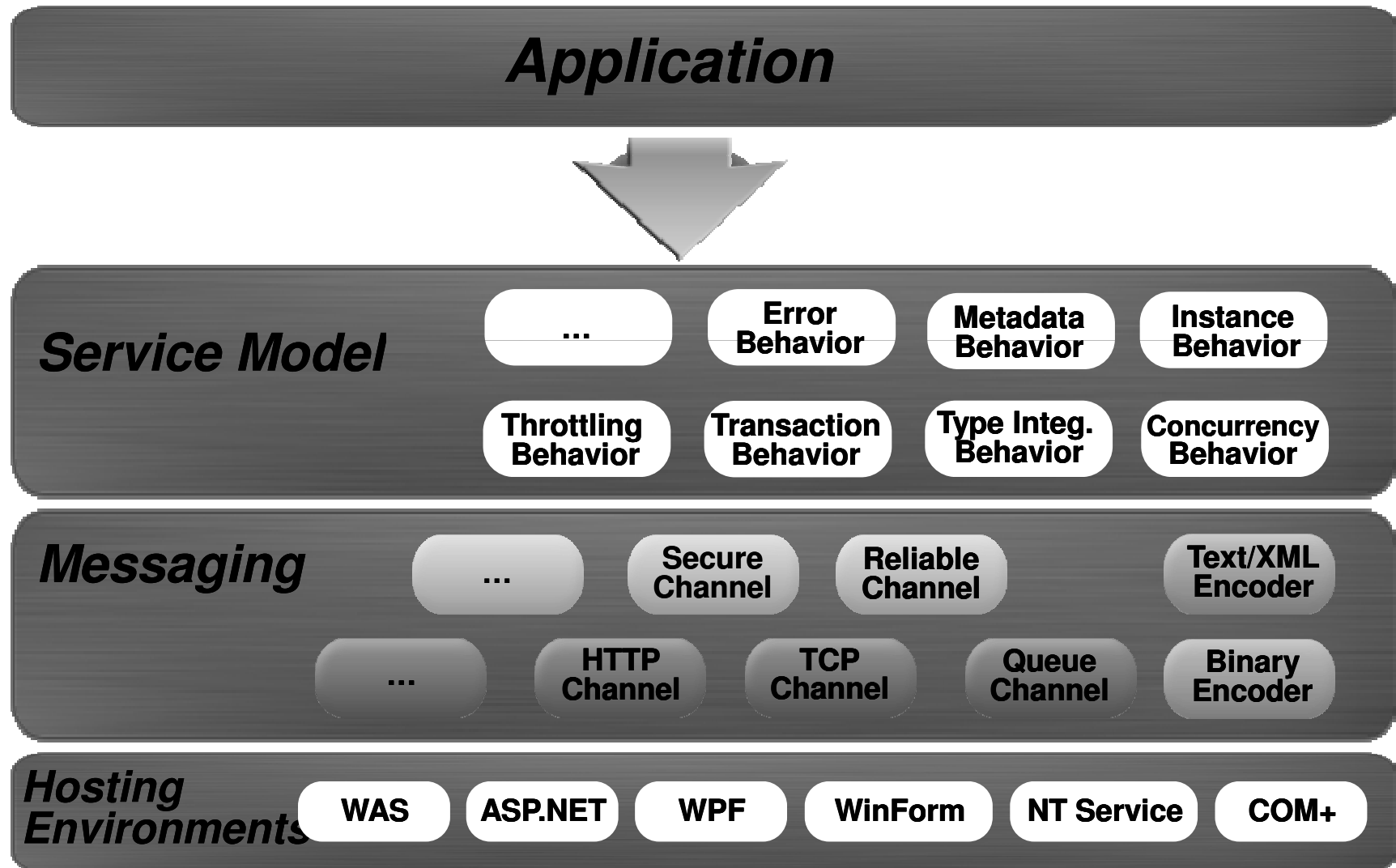```

# Configuring Bindings

```
<endpoint address="Calculator"
          bindingSectionName="basicProfileBinding"
          bindingConfiguration="Binding1"
          contractType="ICalculator" />
```

```
<bindings>
  <basicProfileBinding>
    <binding configurationName="Binding1"
             hostnameComparisonMode="StrongWildcard"
             transferTimeout="00:10:00"
             maxMessageSize="65536"
             messageEncoding="Text"
             textEncoding="utf-8"
    </binding>
  </basicProfileBinding>
</bindings>
```

# Custom Bindings

```xml
<bindings>
    <customBinding>
        <binding configurationName="Binding1">
            <reliableSession bufferedMessagesQuota="32"
                inactivityTimeout="00:10:00"
                maxRetryCount="8"
                ordered="true" />
            <httpsTransport manualAddressing="false"
                maxMessageSize="65536"
                hostnameComparisonMode="StrongWildcard"/>
            <textMessageEncoding maxReadPoolSize="64"
                maxWritePoolSize="16"
                messageVersion="Default"
                encoding="utf-8" />
        </binding>
    </customBinding>
</bindings>
```

# WCF Summary

**Application**

**Service Model**
- ...
- Error Behavior
- Metadata Behavior
- Instance Behavior
- Throttling Behavior
- Transaction Behavior
- Type Integ. Behavior
- Concurrency Behavior

**Messaging**
- ...
- Secure Channel
- Reliable Channel
- Text/XML Encoder
- ...
- HTTP Channel
- TCP Channel
- Queue Channel
- Binary Encoder

**Hosting Environments**
- WAS
- ASP.NET
- WPF
- WinForm
- NT Service
- COM+

# WCF Summary

- WCF is the future of distributed computing
- It combines the best of all existing Microsoft distributed computing stacks
- It uses WS-* standards for interoperability and .NET value-add for performance and integration with existing solutions
- WCF is available for Windows Vista, Windows XP SP2, Windows Server 2003, Windows Server 2008