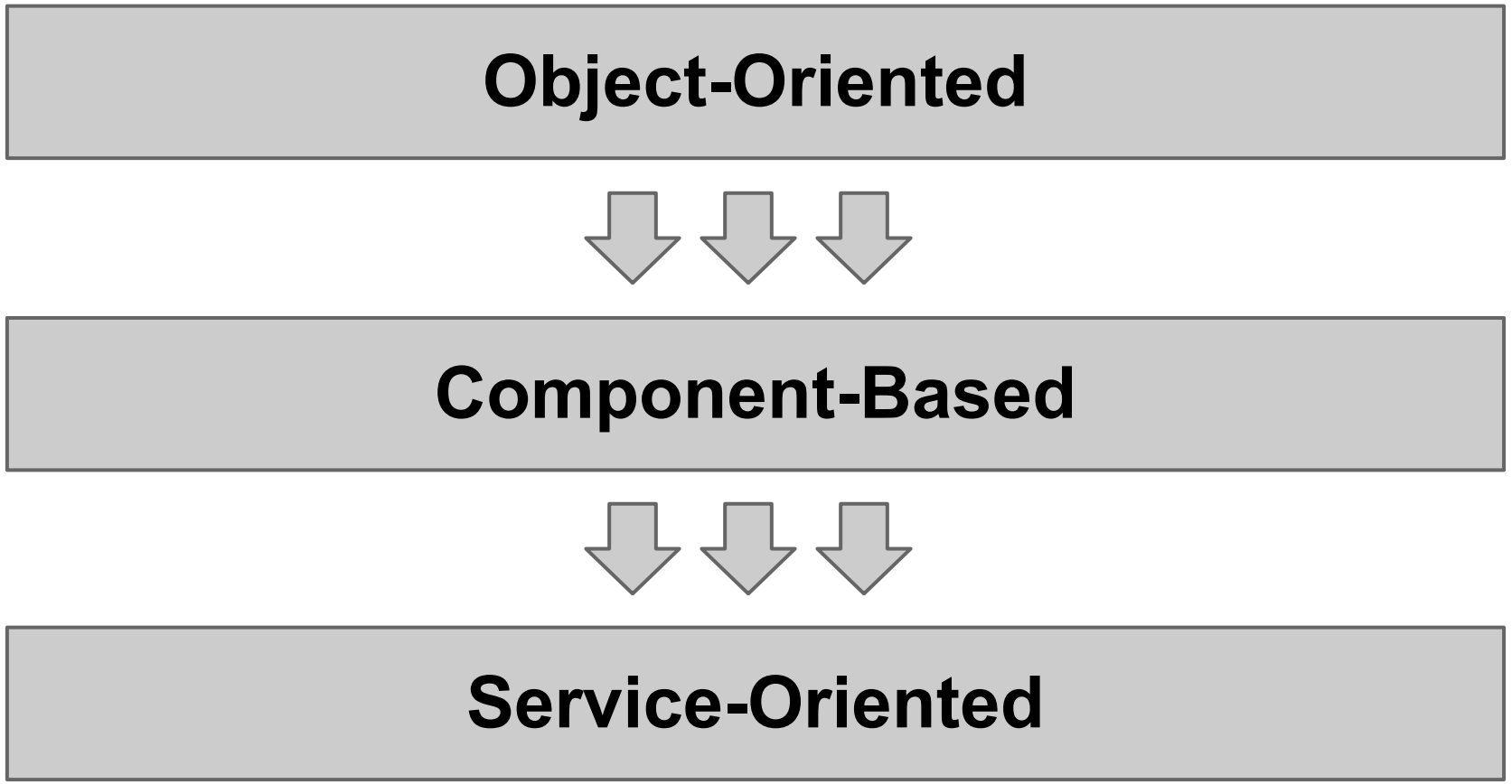
Windows Communication Foundation

Михайлов П.А.

Как мы до этого до...



Хьюстон, у нас проблемы!

- Разработка распределенных программных систем остается сложной и ресурсоемкой
  - Различные методы для решения разных задач
  - Надежность и безопасность взаимодействия
  - Совместимость с приложениями на других платформах

Microsoft и его зоопарк

- ASMX
- WSE
- .NET Remoting
- COM+ (Enterprise Services)
- MSMQ

WCF

- SDK для разработки и развертывания сервисов в системе Windows
- Сервисы описываются метаданными (WSDL или др. отраслевой стандарт)
- Релиз первой версии: 2006 г.
- Текущая версия 4.5

Основные принципы WCF

Схема взаимодействия

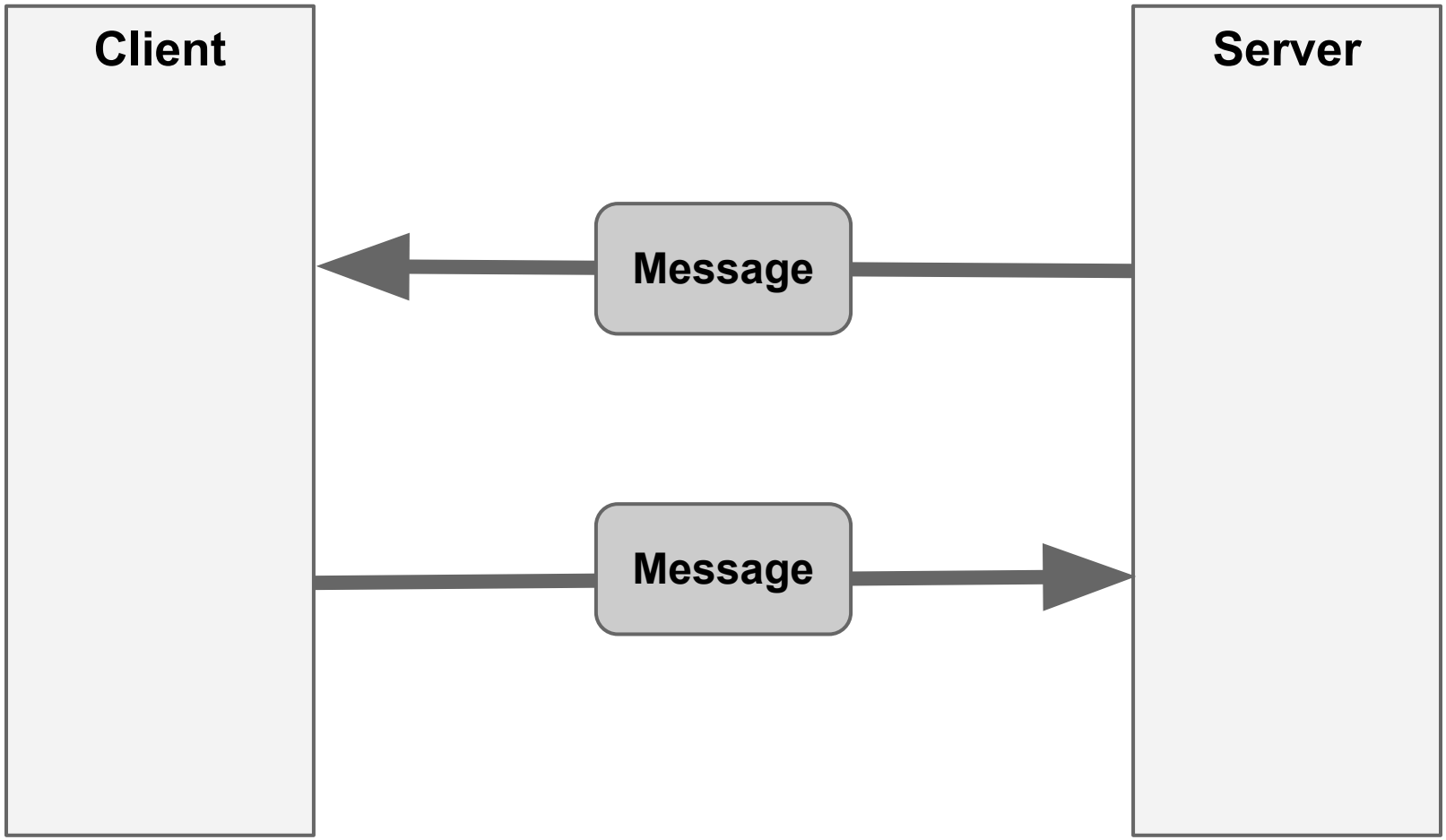


Схема взаимодействия

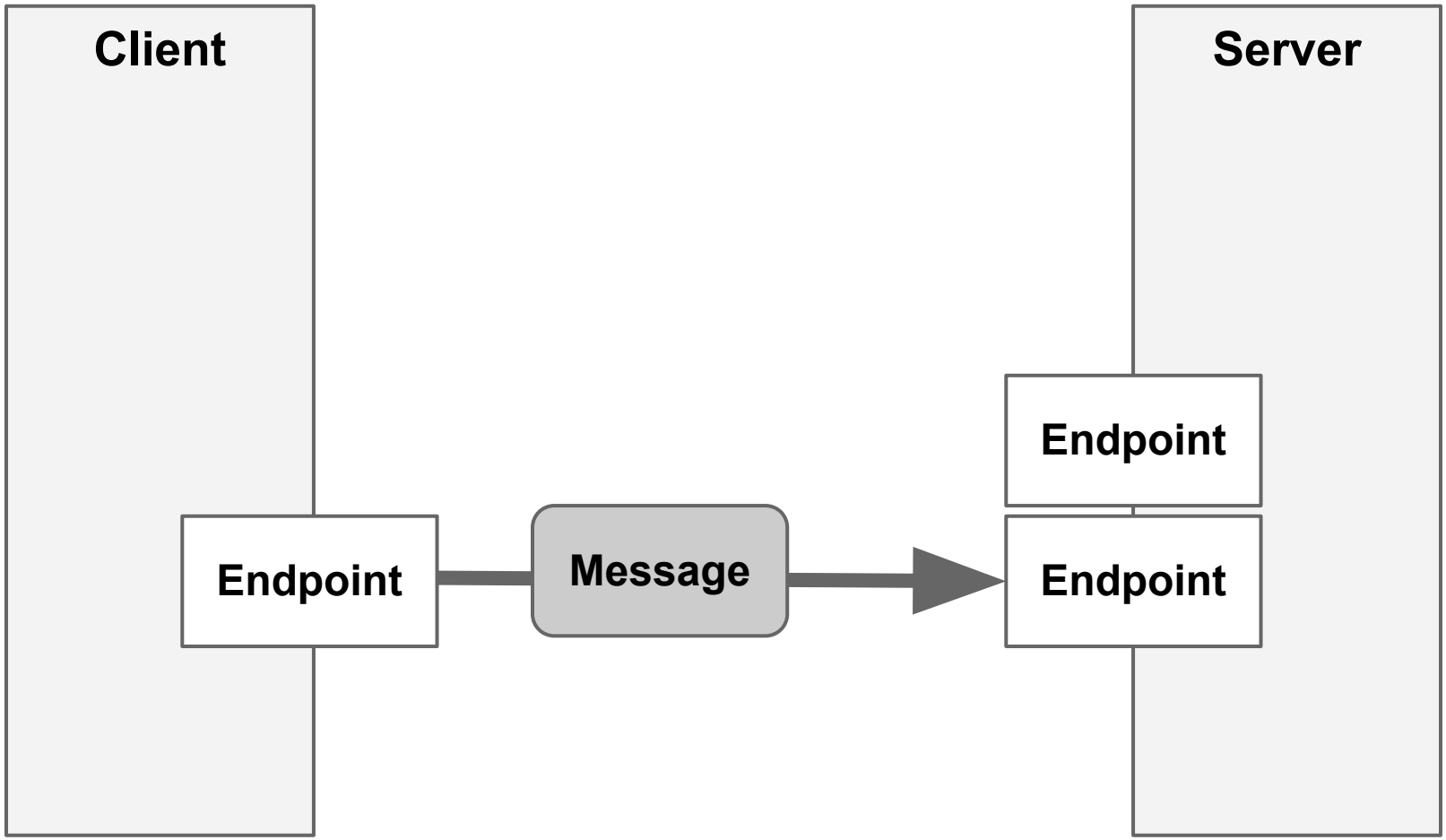
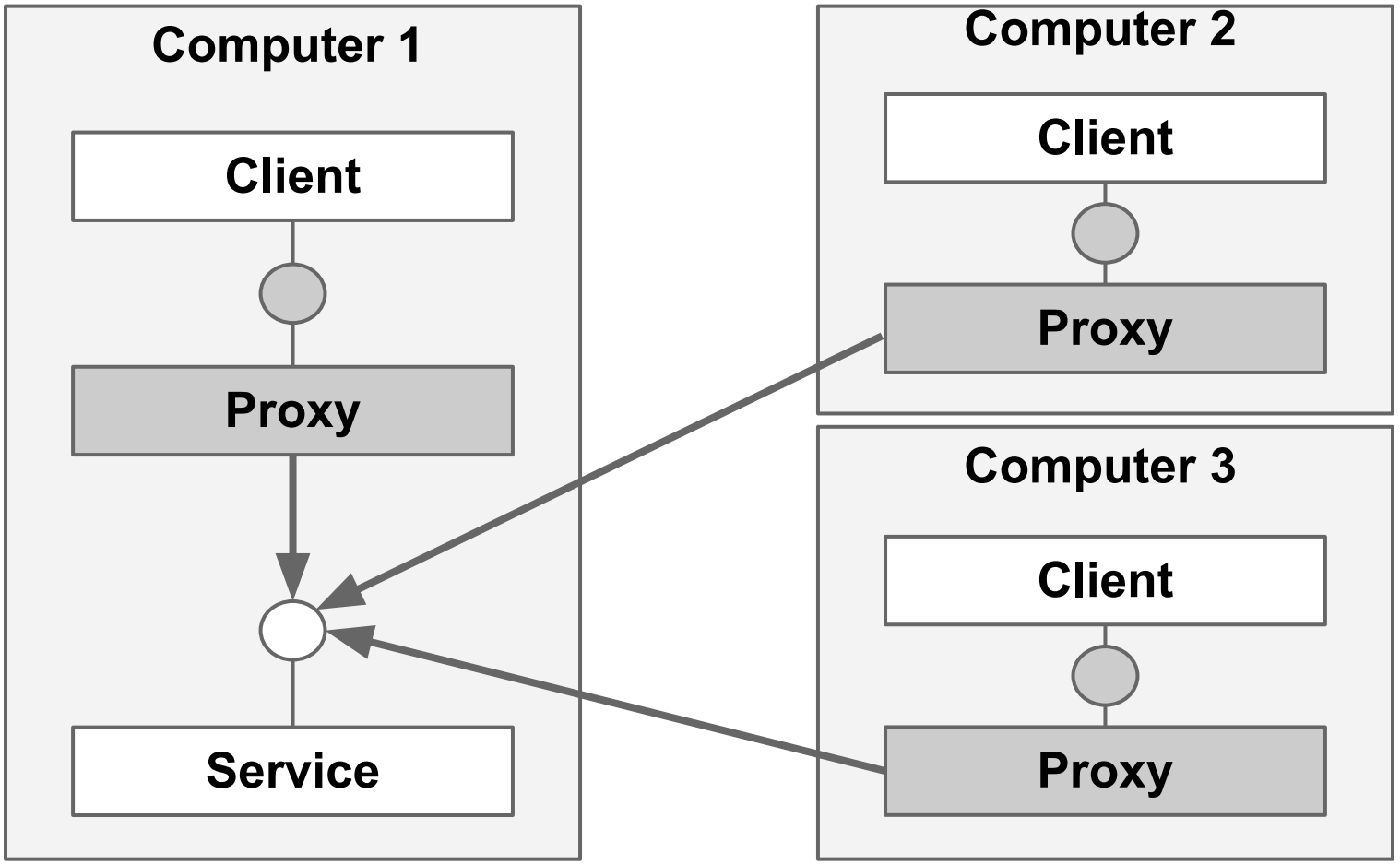


Схема взаимодействия



Конечные точки

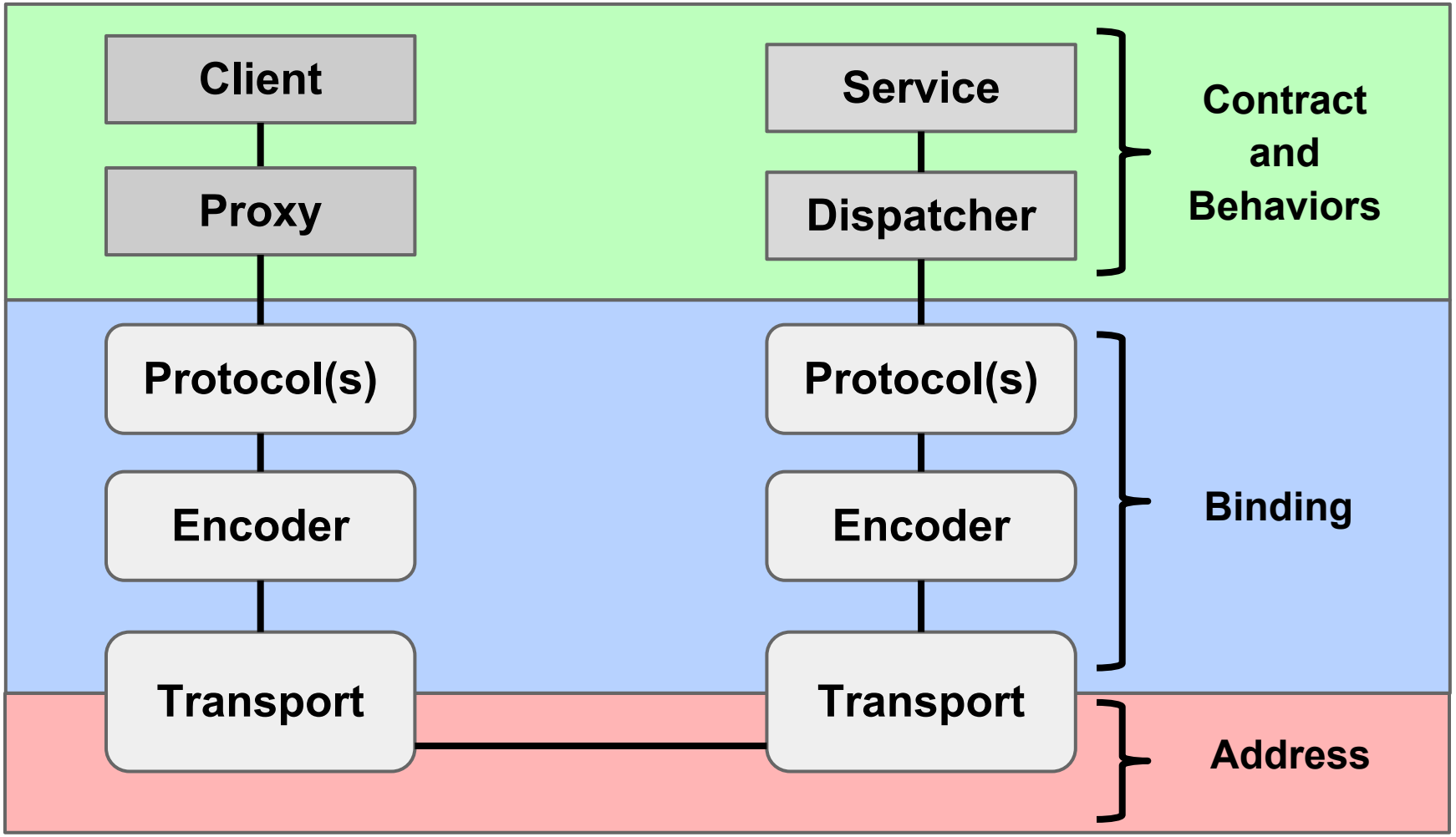
WAT?

- Конечная точка (**Endpoint**) - описание WCF-сервиса

Конечная точка

- Состоит из трех компонентов
  - Адрес (**Где?**)
  - Привязка (**Как?**)
  - Контракт (**Что?**)

Архитектура WCF



Контракты

WAT?

- Контракт (**Contract**) - стандартный платформенно-независимые способ описания того, что служба делает

Типы контрактов

- Контракты сервисов (**Services contracts**)
  - Операции
  - Поведение
  - Способы взаимодействия
- Контракты данных (**Data contracts**)
  - Используемые типы
  - Схемы данных
- Контракты ошибок (**Faults contracts**)
  - Инициируемые ошибки
  - Способы обработки ошибок

Контракт сервиса

```
1  [ServiceContract]
2  interface ISrvContract
3  {
4      [OperationContract]
5      string foo(string str);
6
7      // Не входит в контракт
8      string bar(string str);
9  }
```

```
1  class Service : ISrvContract
2  {
3      public string foo(string str)
4      {
5          return str + bar(str);
6      }
7
8      public string bar(string str)
9      {
10         return str + str;
11     }
12 }
```

Типы контрактов

- Контракты сервисов (**Services contracts**)
  - Функции, методы, операции
  - Поведение
  - Способы взаимодействия
- Контракты данных (**Data contracts**)
  - Используемые типы
  - Схемы данных
- Контракты ошибок (**Faults contracts**)
  - Инициируемые ошибки
  - Способы обработки ошибок

Контракт данных

```
1  [DataContract]
2  public class DataContract
3  {
4      [DataMember]
5      public string foo;
6
7      [DataMember]
8      public string bar;
9
10     public DataContract(string foo, string bar)
11     {
12         this.foo = foo;
13         this.bar = bar;
14     }
15 }
```

Типы контрактов

- Контракты сервисов (**Services contracts**)
  - Функции, методы, операции
  - Поведение
  - Способы взаимодействия
- Контракты данных (**Data contracts**)
  - Используемые типы
  - Схемы данных
- Контракты ошибок (**Faults contracts**)
  - Инициируемые ошибки
  - Способы обработки ошибок

Контракт ошибок

```
1  [ServiceContract]
2  interface ISrvContract
3  {
4      [OperationContract]
5      [FaultContract(typeof(NullReferenceException))]
6      [FaultContract(typeof(string))]
7      string foo(string str);
8
9      // Не входит в контракт
10     string bar(string str);
11 }
```

Модели вызовов операций

"Запрос-Ответ"

```
1  [ServiceContract]
2  interface ISrvContract
3  {
4      [OperationContract]
5      string foo(string str);
6
7      // Не входит в контракт
8      string bar(string str);
9  }
```

Односторонние операции

```
1  [ServiceContract]
2  interface ISrvContract
3  {
4      [OperationContract(IsOneWay = true)]
5      string foo(string str);
6
7      // Не входит в контракт
8      string bar(string str);
9  }
```

Операции обратного вызова

```
interface ICallBackContract
{
    [OperationContract]
    void OnCallBack();
}
```

```
[ServiceContract(CallBackContract = typeof(ICallBackContract))]
interface ISrvContract
{
    [OperationContract]
    string foo(string str);

    // Не входит в контракт
    string bar(string str);
}
```

Привязки

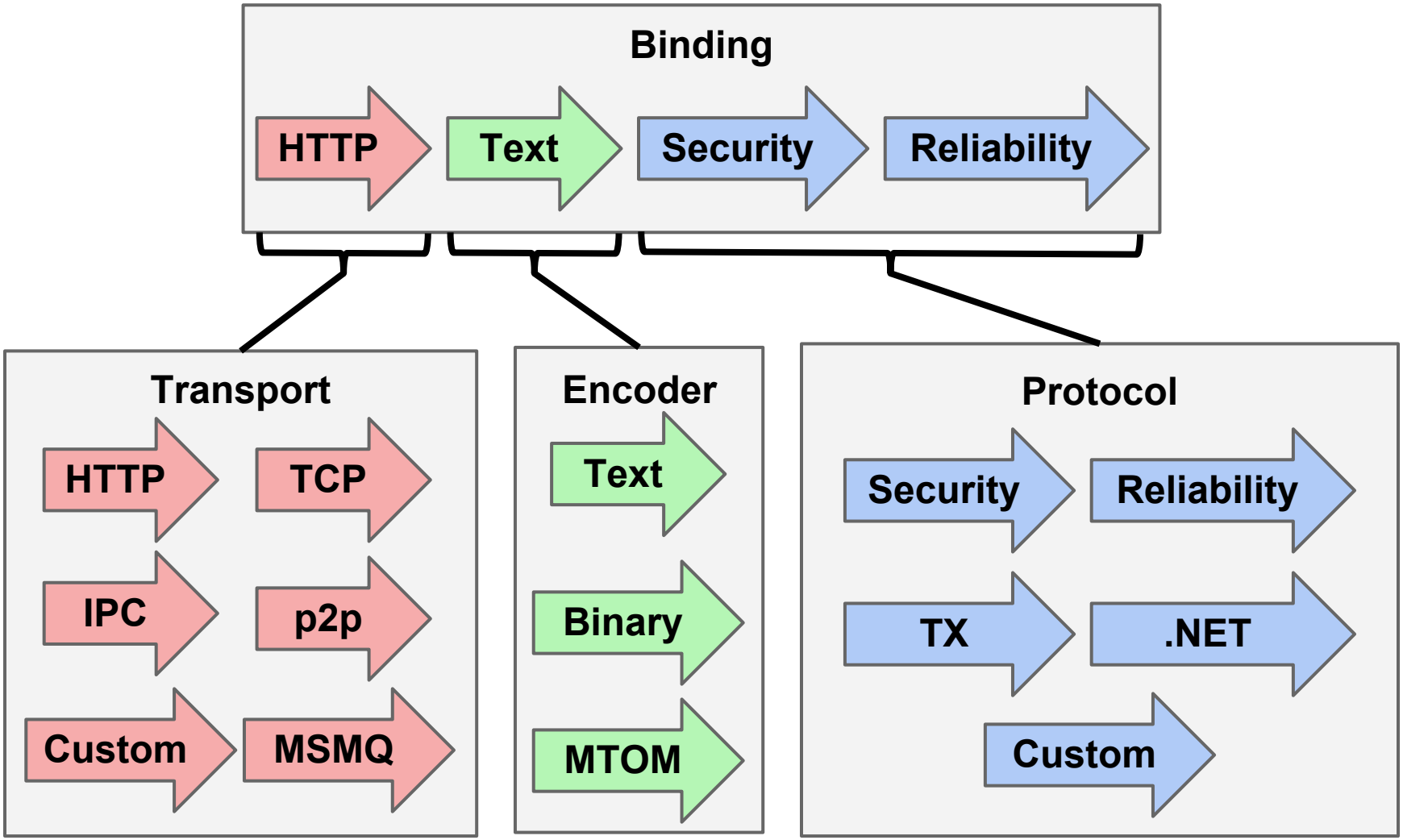
WAT?

- Привязка (**Binding**) **-** объект, используемый для задания сведений о связи, необходимых для подключения к конечной точке сервиса.

Структура привязки

- Привязка состоит из трех элементов
  - Протокол (**Protocol**)
  - Кодирование (**Encoder**)
  - Транспорт (**Transport**)

Структура привязки



Безопасность

Режимы безопасности

- Отсутствует (**None**)
- На уровне сообщений (**Message**)
- На уровне транспорта (**Transport**)
- Смешанный (**Mixed**)
- Двухуровневый (**Both**)

Адреса

Адреса служб

<transport>://<domain>[:<port>][/<uri>]

Транспорты:

- HTTP
- TCP
- p2p
- IPC
- MSMQ

Хостинг

Автохостинг

- Сервис располгается в процессе приложения

Автохостинг

+) Простота использования
+) Гибкость
+) Простота отладки
+) Простота развертывания
+) Поддержка всех привязок

-) Ограниченная доступность
-) Ограниченная функциональность

Хостинг в службе Windows

+) Автозапуск
+) Восстановление
+) Управляемость
+) Безопасность
+) Поддержка всех привязок

-) Развертывание

Хостинг с использованием IIS

+) Простота развертывания
+) Надежность
+) Разделяемый хостинг
+) Динамическая компиляция

-) Ограничения по использованию привязок
-) Размещение

Example time!

Контракт данных

```
1  [DataContract]
2  public class Greeting
3  {
4      private string stringValue;
5
6      [DataMember]
7      public string Text
8      {
9          get { return stringValue; }
10         set { stringValue = value; }
11     }
12 }
```

Контракт сервиса

```
1  [ServiceContract]
2  public interface IGreetingSrv
3  {
4      [OperationContract]
5      string GetGreeting(string name);
6
7      [OperationContract]
8      Greeting GetObjGreeting(string name);
9  }
```

Реализация сервиса

```
1   public class GreetingSrv : IGreetingSrv
2   {
3       public string GetGreeting(string name)
4       {
5           return string.Format("Hello, {0}", name);
6       }
7
8       public Greeting GetObjGreeting(string name)
9       {
10          Greeting g = new Greeting();
11          g.Text = String.Format("Hello, {0}", name);
12          return g;
13      }
14  }
```

Конфигурация приложения

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <system.serviceModel>
        <services>
            <service name="WCFGreetingSrv.GreetingSrv">
                <endpoint address="/GreetingSrv"
                          binding="basicHttpBinding"
                          contract="WCFGreetingSrv.IGreetingSrv" />
                <host>
                    <baseAddresses>
                        <add baseAddress="http://localhost:8733" />
                    </baseAddresses>
                </host>
            </service>
        </services>

        <behaviors>
            <serviceBehaviors>
                <behavior>
                    <serviceMetadata httpGetEnabled="True" httpsGetEnabled="False"/>
                </behavior>
            </serviceBehaviors>
        </behaviors>
    </system.serviceModel>
</configuration>
```

Конфигурация приложения

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <system.serviceModel>
        <services>
            <service name="WCFGreetingSrv.GreetingSrv">
                <endpoint address="/GreetingSrv"
                          binding="basicHttpBinding"
                          contract="WCFGreetingSrv.IGreetingSrv" />
                <host>
                    <baseAddresses>
                        <add baseAddress="http://localhost:8733" />
                    </baseAddresses>
                </host>
            </service>
        </services>

        <behaviors>
            <serviceBehaviors>
                <behavior>
                    <serviceMetadata httpGetEnabled="True" httpsGetEnabled="False"/>
                </behavior>
            </serviceBehaviors>
        </behaviors>
    </system.serviceModel>
</configuration>
```

Конфигурация приложения

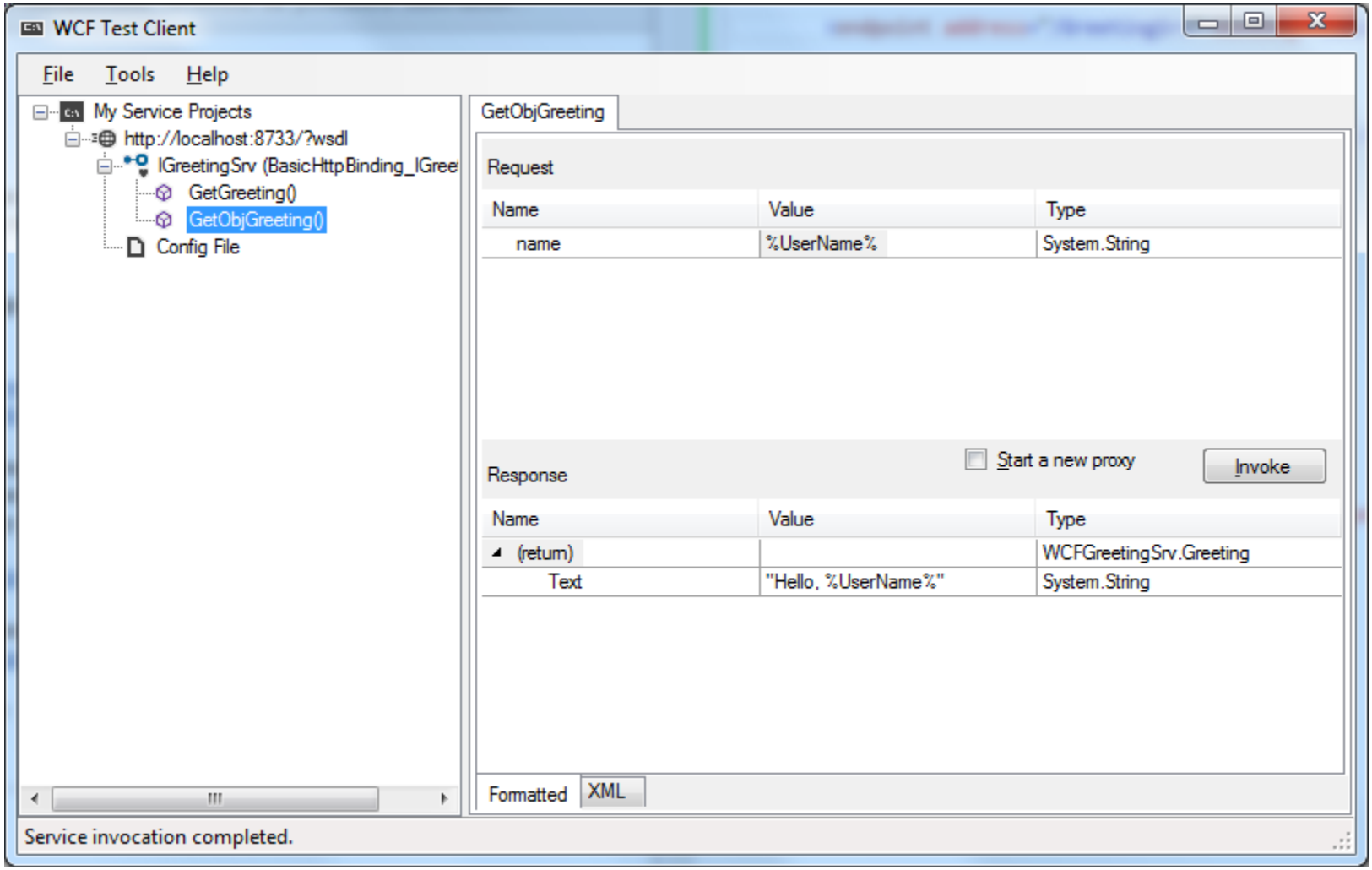
```
5           <service name="WCFGreetingSrv.GreetingSrv">
6               <endpoint address="/GreetingSrv"
7                         binding="basicHttpBinding"
8                         contract="WCFGreetingSrv.IGreetingSrv" />
9               <host>
10                  <baseAddresses>
11                      <add baseAddress="http://localhost:8733" />
12                  </baseAddresses>
13              </host>
14          </service>
```

Конфигурация приложения

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <system.serviceModel>
        <services>
            <service name="WCFGreetingSrv.GreetingSrv">
                <endpoint address="/GreetingSrv"
                          binding="basicHttpBinding"
                          contract="WCFGreetingSrv.IGreetingSrv" />
                <host>
                    <baseAddresses>
                        <add baseAddress="http://localhost:8733" />
                    </baseAddresses>
                </host>
            </service>
        </services>

        <behaviors>
            <serviceBehaviors>
                <behavior>
                    <serviceMetadata httpGetEnabled="True" httpsGetEnabled="False"/>
                </behavior>
            </serviceBehaviors>
        </behaviors>
    </system.serviceModel>
</configuration>
```

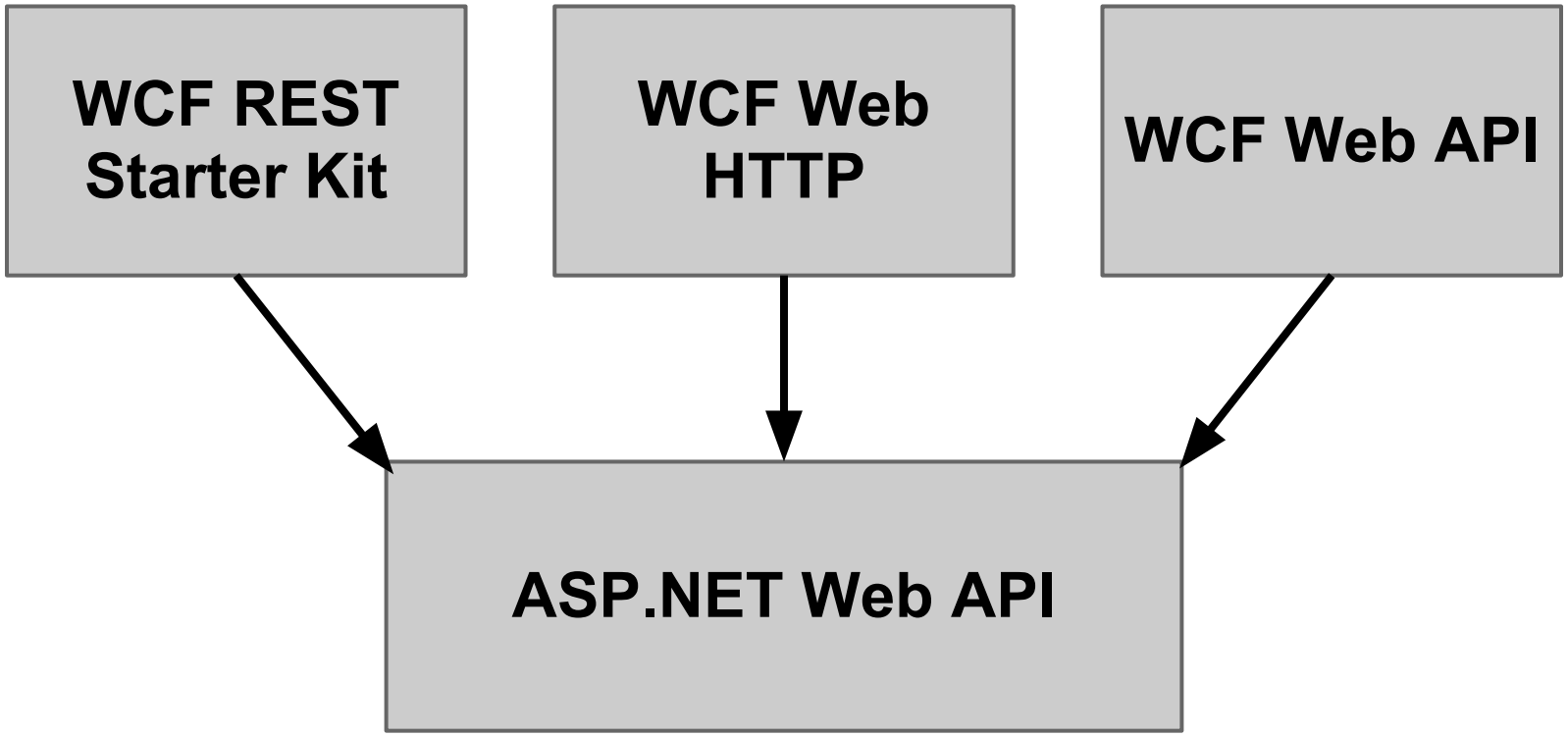
Тестовый клиент



WCF vs. ASP.NET Web API

WCF и REST

WCF и REST



Вопросы?