Учебный курс

Технологии и средства разработки корпоративных систем

Лекция 8

Создание приложений по технологии Windows Communications Foundation

Лекции читает

кандидат технических наук, доцент

Зыков Сергей Викторович

Содержание

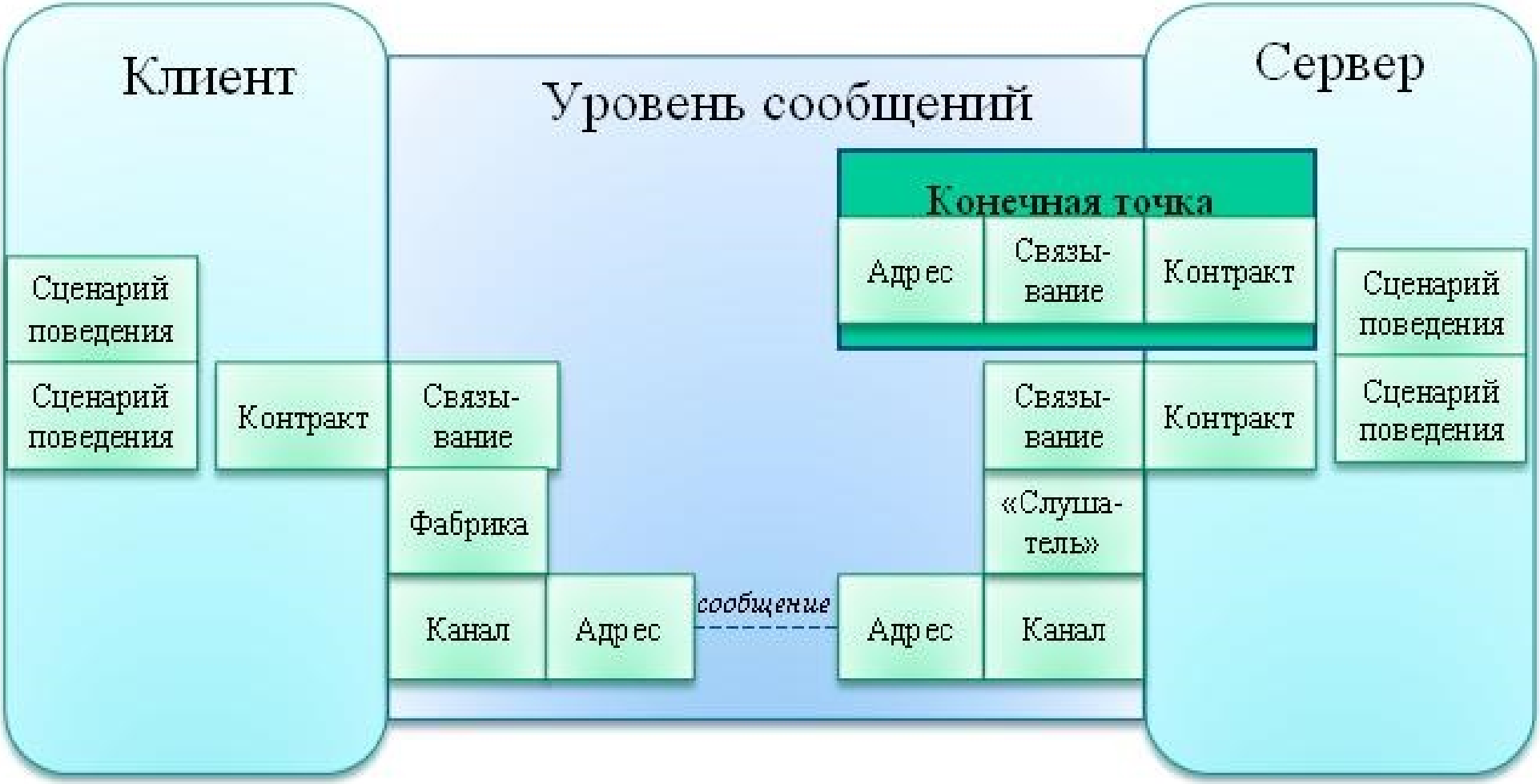
Сервисно-ориентированная архитектура

- **Сервисно-ориентированная архитектура** (SOA) – это архитектура, основанная на использовании сервисов, определяющая протоколы и методы их взаимодействия, а также методики реализации тех или иных решений на основе сервисов
- SOA осуществляет взаимодействие сервисов, оставляя их независимыми друг от друга
- Сервисы в SOA являются строительными блоками бизнес-процессов

Цели Windows Communication Foundation (WCF) – Microsoft-реализации SOA

Общая архитектурная схема WCF

Программная модель WCF



Контракты WCF

Контракты Сервисов	• Функциональные операции реализуемые данным сервисом • Используется WSDL
Контракты Данных	• Описывают структуры данных используемые при взаимодействии с сервисом • Используется XSD
Контракты Сообщений	• Определяют то, каким образом типы CLR будут отображаться на формат SOAP

Контракты сервисов (1)

1. При определении используют атрибуты:
   - `ServiceContract` (для классов и/или интерфейсов)
   - `OperationContract` (для методов)

2. Определяют направление взаимодействия в сервисах:
   - однонаправленное;
   - дуплексное

Контракт сервиса (2)

```
// Пример описания контракта сервиса
  [ServiceContract]
    public interface IService1
    {
        [OperationContract]
        string GetData(int value);
        [OperationContract]
        CompositeType
  GetDataUsingDataContract(CompositeType
  composite);
        // TODO: Вписать описания операций сервиса
    }
```

Контракт данных (1)

1. использует атрибуты для класса и его членов:
   - DataContract;
   - DataMember;
   - CollectionDataContract

2. обеспечивает сериализацию данных:
   – DataContractSerializer

Контракт данных (2)

```
// пример описания контракта данных
   [DataContract]
    public class CompositeType{
        bool boolValue = true;
        string stringValue = "Hello ";
        [DataMember]
        public bool BoolValue{
            get { return boolValue; }
            set { boolValue = value; }
        }
        [DataMember]
        public string StringValue{
            get { return stringValue; }
            set { stringValue = value; }
        }
    }
```

Контракт сообщения (1)

- Использует атрибуты:
  - `MessageContract`;
  - `MessageHeader`;
  - `MessageBodyMember`
- Особенности:
  - не более одного входного параметра;
  - не более одного возвращаемого значения;
  - альтернатива контракту данных;
  - не используется совместно с контрактом данных

Контракт сообщения (2)

```
// пример контракта сообщения
[MessageContract]
  public class ItemMesage
 {
      [MessageHeader]
       public SomeProtocol ProtocolHeader;
      [MessageBody]
       public Item Content;
   }
```

Каналы WCF

- служат для подготовки и доставки сообщений
- представляются в виде стека
- делятся на:
  - транспортные;
  - протокольные
- Возможные конфигурации:
  - однонаправленный;
  - дуплексный;
  - запрос-ответ
- Каналы, фабрики каналов и «слушатели» наследуют единый интерфейс `ICommunicationObject`

Связывание

- Связыванием (binding) – называется сконфигурированный стек каналов WCF
- Связывания образуют стек каналов с помощью специальных элементов
- В WCF по умолчанию определены 9 типов связываний

Связывание


Начало
Требуется интероперабельность?
Локальное?
netNamedPipeBinding
Уровень интероперабельности?
basicHttpBinding
Требуются очереди?
Требуются унаследованные MSMQ?
msmqIntegrationBinding
netMsmqBinding
Нужен дуплексный режим?
wsDualHttpBinding
Требуется сеть «точка-точка»?
netPeerTcpBinding
Требуется выделенная безопасность?
wsFederationHttpBinding или ws2007FederationHttpBinding
wsHttpBinding или ws2007HttpBinding
netTcpBinding

Сценарий поведения (behavior) -

класс WCF, влияющий на поведение системы во время выполнения

- **Виды**:
  - сервисный (service) - работает на уровне сервиса, имеет доступ ко всем его конечным точкам, отвечает за транзакции, авторизацию, создание объектов, аудит
  - конечных точек (endpoints) - работает на уровне конечных точек, реализует инспекцию и обработку входящих/исходящих сообщений
  - операционный (operational) - используется на уровне отдельных операций, управляет сериализацией, транзакционным потоком
  - обратного вызова (callback) – аналог сценария сервисов, используется для элементов, функционирующих на клиенте при дуплексной коммуникации.

Сценарии поведения для управления потоком данных

На клиенте:

- Инспекция параметров
- Инспекция сообщений
- Форматирование сообщений

На сервере:

- Инспекция параметров
- Инспекция сообщений
- Форматирование сообщений
- Выбор операции (метода)
- Вызов операции (метода)

Сервисные сценарии поведения

Класс WCF	Функциональность
`InstanceContextMode`	Определяет , сколько экземпляров сервиса будут обслуживать запросы (возможные значения `Single, PerCall, PerSession`)
`ConcurrencyMode`	Определяет число потоков на экземпляр сервиса (возможные значения: `Single, Reentrant, Multiple`)
`ServiceMetadataBehavior`	Экспорт и публикация данных о сервисе

Операционные сценарии поведения: транзакции

- Виды транзакций:
  - многошаговые бизнес-процессы;
  - короткие транзакции (быстро завершаются, затрагивают небольшое число объектов и зависимостей)

- Для определения транзакционного поведения в WCF необходимо использовать атрибут

```
[OperationBehavior
    (TransactionScopeRequired=true)]
```

Сериализация и кодировка в WCF

- ***Сериализация*** – процесс преобразования графа обьектов в формат `XML Information Set (XML Infoset)`
- ***Кодировка*** – процесс преобразования сообщения WCF в поток байт
- **Виды кодировки:**
  - двоичный;
  - текстовый;
  - Message Transmission Optimization Mechanism (MTOM);
  - JavaScript Object Notation;
  - Plain-Old-XML (POX)

Сравнение методов сериализации WCF

Сериализатор	Характеристики
`DataContract Serializer`	Сериализатор по умолчанию WCF. Основан на схемах XSD. Может использоваться для взаимодействия с внешними приложениями
`NetDataConract Serializer`	Отличается возможностью добавления информации о типах CLR. Используется только внутри конкретного .NET приложения
`XmlSerializer`	Стандартный сериализатор .NET. Совместим со встроенными типами .NET, позволяет конфигурировать выходной XML, работает с Web-сервисами ASP.NET
`DataContractJs onSerializer`	Поддерживает нотацию объектов JavaScript.

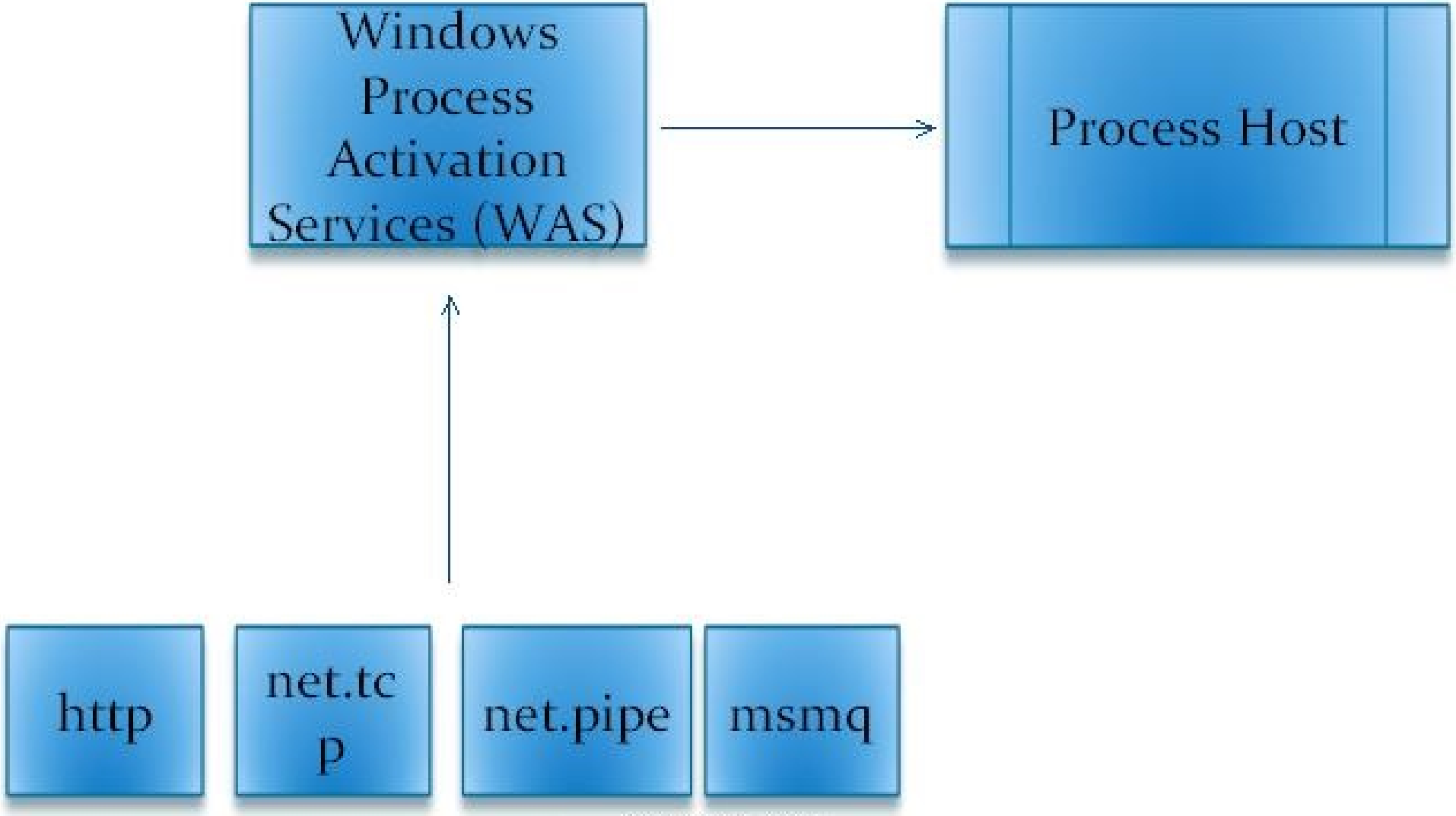
Хостинг

- ***Service Host*** (***хост***) – процесс, ответственный за работу и контекст выполнения сервиса
- В качестве хоста WCF может выступать любой процесс;
- IIS (Internet InformationServices) и WAS (Windows Process Activation Services) имеют встроенную поддержку хостинга
- Для любого хоста необходимо реализовать:
  - создание объекта класса ServiceHost;
  - добавление конечной точки;
  - запуск прослушивания сообщений

Среда хостинга WAS

- WAS (Windows Process Activation Services) - среда хостинга;
- WAS встроена в Windows Vista и Windows Server 2008;
- WAS позволяет размещать сервисы в гибкой среде, которая не требует обязательного использования протокола обмена `http`

Схема реализации хостинга в среде WAS



Пример сервиса WCF (1)

- **Описание** - сервис, реализующий функции калькулятора.

```
// Контракт сервиса
[ServiceContract(Namespace=
      "http://Microsoft.ServiceModel.Samples")]
         public interface ICalculator{
            [OperationContract]
            double Add(double n1, double n2);
            [OperationContract]
            double Subtract(double n1, double n2);
            [OperationContract]
            double Multiply(double n1, double n2);
            [OperationContract]
            double Divide(double n1, double n2);
         }
```

Пример сервиса WCF (2)

```
// Класс, реализующий контракт:
public class CalculatorService : ICalculator{
    public double Add(double n1, double n2){
                    return n1 + n2;
    }
    public double Subtract(double n1, double n2){
                    return n1 - n2;
    }
    public double Multiply(double n1, double n2){
                    return n1 * n2;
    }
    public double Divide(double n1, double n2){
                    return n1 / n2;
    }
}
```

Пример сервиса WCF (3)

```
//конфигурационный файл  сервера (начало):
<system.serviceModel>
    <services>
      <service
   name="Microsoft.ServiceModel.Samples.CalculatorService"
         behaviorConfiguration="CalculatorServiceBehavior">
        <!-- ICalculator is exposed at the base address
   provided by host:
   http://localhost/servicemodelsamples/service.svc  -->
        <endpoint address=""
                  binding="wsHttpBinding"
   contract="Microsoft.ServiceModel.Samples.ICalculator" />
        <!-- the mex endpoint is explosed at
   http://localhost/servicemodelsamples/service.svc/mex -->
        <endpoint address="mex"
                  binding="mexHttpBinding"
                  contract="IMetadataExchange" />
      </service>
    </services>
```

Пример сервиса WCF (4)

```
//конфигурационный файл  сервера (окончание):

<!--For debugging purposes set the
  includeExceptionDetailInFaults attribute to true--
  >
    <behaviors>
      <serviceBehaviors>
        <behavior name="CalculatorServiceBehavior">
          <serviceMetadata httpGetEnabled="True"/>
          <serviceDebug
  includeExceptionDetailInFaults="False" />
        </behavior>
      </serviceBehaviors>
    </behaviors>
  </system.serviceModel>
```

Пример сервиса WCF (5)

Код на клиенте (с сокращениями - начало):

```
 [System.Diagnostics.DebuggerStepThroughAttribute()]
    [System.CodeDom.Compiler.GeneratedCodeAttribute
        ("System.ServiceModel", "3.0.0.0")]
    public partial class CalculatorClient:
      System.ServiceModel.ClientBase<Microsoft.ServiceModel.Samples.
        ICalculator>, Microsoft.ServiceModel.Samples.ICalculator{
          public CalculatorClient(){}
          public CalculatorClient(string endpointConfigurationName):
             base(endpointConfigurationName){}
          public CalculatorClient(string endpointConfigurationName,
            string remoteAddress):
             base(endpointConfigurationName, remoteAddress){}
          public CalculatorClient(string endpointConfigurationName,
            System.ServiceModel.EndpointAddress remoteAddress):
             base(endpointConfigurationName, remoteAddress){}
          public CalculatorClient(System.ServiceModel.Channels.Binding
            binding,System.ServiceModel.EndpointAddress remoteAddress):
             base(binding, remoteAddress){}
```

Пример сервиса WCF (5) *(продолжение)*

```
    public double Add(double n1, double n2)
    {
        return base.Channel.Add(n1, n2);
    }
    public double Subtract(double n1, double n2)
    {
        return base.Channel.Subtract(n1, n2);
    }
    public double Multiply(double n1, double n2)
    {
        return base.Channel.Multiply(n1, n2);
    }
    public double Divide(double n1, double n2)
    {
        return base.Channel.Divide(n1, n2);
    }
}
```

Пример сервиса WCF (6)

Код на клиенте (с сокращениями – окончание):

```
        public double Add(double n1, double n2){
            return base.Channel.Add(n1, n2);
        }

        public double Subtract(double n1, double n2){
            return base.Channel.Subtract(n1, n2);
        }

        public double Multiply(double n1, double n2){
            return base.Channel.Multiply(n1, n2);
        }

        public double Divide(double n1, double n2){
            return base.Channel.Divide(n1, n2);
        }
    }
```

Библиография (1)

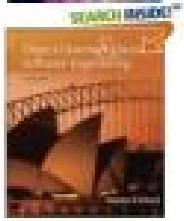
Основная литература:

http://msdn.microsoft.com/en-us/netframework/aa663324.aspx

- S. Resnick, R. Crane, C. Bowen. Essential Windows Communication Foundation. Pearson Education, Inc., 2008
- C.Peiris, D. Mulder Pro WCF: Practical Microsoft SOA Implementation. A-Press, 2007

Библиография (2)

Дополнительная литература:



- Соммервилл И. Инженерия программного обеспечения (6-е изд.), м.: Вильямс, 2002.- 624 с., ил.
- Schach S.R.: Object-Oriented and Classical Software Engineering (5 ed.) McGraw-Hill, 2001, 744 pp.
- Зыков С.В. Проектирование корпоративных порталов.– М.: МФТИ, 2005.– 258 с.
- Буч Г., Рамбо Дж., Джекобсон А. Язык UML. Руководство пользователя: Пер. с англ.- изд. 2-е, М.:ДМК Пресс, С.-Пб.: Питер, 2004.- 432 с.

Библиография (3)

Дополнительная литература:

- Жизненный цикл программного обеспечения ИС. http://www.tver.mesi.ru/e-lib/res/661/2/devis_2.html
- Martin Fowler, The New Methodology http://www.martinfowler.com/articles/newMethodology.html

Задания для самостоятельной работы

Реализовать Web-сервис, который…

1. … реализует функции простого калькулятора: сложение, вычитание, умножение, деление
2. … конвертирует валюты (рубли в доллары и т.п.). Курс – случайная величина с периодом обновления 3 минуты.
3. … принимает как параметр имя пользователя и определяет, сколько раз его вызывал пользователь с этим именем
4. … вычисляет квадратный корень в форме консольного приложения-клиента.
5. … определяет знак зодиака, по введенной дате

Благодарю за внимание!

Вопросы?

- http://zykov.altweb.ru
- szykov@hotmail.com
- sergey.zykov@tekama.com