45asp

> Дима Заянковский: ASP LAB 4  
  
UDDI - структура  
  
**UDDI — Universal Description, Discovery and Integration**  
  
Задача UDDI — предоставить механизм для обнаружения веб-сервисов. UDDI задает бизнес-реестр, в котором провайдеры веб-сервисов могут регистрировать сервисы, а разработчики — искать необходимые им сервисы. Компании IBM, Microsoft и Ariba создали собственные UDDI-реестры (в скором времени эти реестры будут объединены в веб-реестр), в одном из которых разработчики могут зарегистрировать свои веб-сервисы, после чего данные будут автоматически реплицированы в другие реестры.  
  
UDDI базируется на элементах четырех типов: Business Entity, Business Service, Binding Template и Technology Model.  
  
Элемент Business Entity описывает индустрию, предоставляющую данный веб-сервис. Этот элемент может включать описания категорий для данной индустрии, облегчающие более детальный поиск сервисов.  
  
Business Service — это класс сервисов в рамках определенной отрасли промышленности или услуг. Каждая отрасль принадлежит определенному элементу Business Entity.  
  
Вместе Binding Template и Technology Model определяют веб-сервис. Technology Model содержит абстрактное описание, а Binding Template — конкретную спецификацию сервиса. Каждый элемент Binding Template принадлежит определенному элементу Business Service, но несколько элементов Binding Template могут ссылаться на один элемент Technology Model.  
  
Бизнес-реестр UDDI сам является SOAP веб-сервисом. Он поддерживает операции создания, модификации, удаления и поиска элементов всех четырех рассмотренных выше типов.  
  
  
  
BasicProfile  
  
WS-I Basic Profile 1.1 - это набор стандартов и рекомендаций, разработанный Всемирной инициативой веб-сервисов (Web Services Interoperability Organization, WS-I), с целью обеспечения совместимости и интероперабельности между веб-сервисами, разработанными на основе протокола SOAP (Simple Object Access Protocol) и других важных веб-сервисных стандартов.  
WS-I была создана с целью устранения различных интерпретаций и неясностей в спецификациях SOAP, WSDL (Web Services Description Language) и UDDI (Universal Description, Discovery, and Integration), что позволило разработчикам создавать веб-сервисы, которые могли взаимодействовать между собой более надежно.  
WS-I Basic Profile 1.1 определяет набор требований и рекомендаций, включая следующие:  
 •   Использование XML 1.0 для представления данных в SOAP-сообщениях.  
 •   Определение структуры SOAP-сообщений и использование стандартных элементов в заголовке и теле сообщения.  
 •   Обязательное использование XML-схем для определения формата данных, передаваемых через веб-сервис.  
 •   Указание правил кодирования данных, включая обработку дат, чисел и строк.  
 •   Определение обязательных HTTP-заголовков, таких как "Content-Type" и "SOAPAction", для правильной передачи SOAP-сообщений через HTTP.  
  
  
Структура Port (WSDL 1.1)  
<wsdl:service>  
<wsdl:port name="PortName" binding="tns:BindingName">  
<soap:address location="<https://example.com/ServiceEndpoint>" />  
</wsdl:port>  
</wsdl:service>  
  
Структура Endpoint (WSDL 2)  
  
<endpoint name="SoapEndpoint" binding="m:SoapBinding"  
address="<http://ws.accurateweather.example/set>"/>  
  
 • <endpoint>: Это элемент, представляющий конечную точку веб-службы.  
 • name="EndpointName": Это атрибут, который указывает имя данной конечной точки. Имя конечной точки может быть уникальным в контексте службы.  
 • binding="BindingName": Этот атрибут указывает, какая привязка будет использоваться для данной конечной точки. Привязка определяет протоколы, которые будут использоваться для взаимодействия с конечной точкой (например, HTTP, TCP, и т. д.).  
 • address="EndpointAddress": Здесь указывается адрес (URL) конечной точки, по которому клиенты смогут обращаться к службе. Этот адрес может быть в виде URL-адреса, URI или другой формы, зависящей от используемой привязки.  
  
  
Endpoint описывается в теге Сервис. Сервис может включать несколько портов. У порта есть имя и привязка. Привяза происходит с отъявленными привязками ранее. Тег порт внутри содержит тег адрес с атрибутом Location. Location содержит адрес к сервису. > Дима Заянковский: Структура SOAP сообщения  
Вот общая структура SOAP сообщения:  
 • **Заголовок (Header)**: Заголовок SOAP сообщения содержит метаданные и информацию о самом сообщении. Здесь могут быть размещены различные элементы, такие как аутентификация, управление транзакциями, маршрутизация, дополнительные информационные теги и другие дополнительные данные. Заголовок является необязательным и может отсутствовать в SOAP сообщении.  
 • **Тело (Body)**: Тело SOAP сообщения содержит фактические данные, которые передаются между клиентом и сервером. Это место, где размещаются запросы и ответы на операции, которые клиент и сервер могут выполнять друг над другом. Тело является обязательной частью SOAP сообщения.  
1. **SOAP Fault** (может в Header или Body):   env:Сode – код сообщения; env:Reason – пояснение к коду; env:Node – узел назначения; env: Role – роль назначения; env: Detail – детальная информация.        –  Error message.   
  
  
  
  
<!-- структура wsdl-документа  -->  
<definition>  
    
  <types>  
    <!-- оперделение типов, используемых web-сервисом -->  
    <!-- для пл.-независимого WSDL используется синтаксис XML Schema  -->  
  </types>  
    
  <message>  
    <!-- сообщения, используемые web-сервисом -->  
    <!-- сообщений, может быть несколько  -->  
    <!-- каждое сообщение может состоять из нескольких частей  -->  
      
  </message>  
    
  <portType>  
    <!-- методы, предоставляемые web-сервисом -->  
    <!-- может быть несколько портов -->  
    <!-- определены операции web-сервиса и используемые сообщения   -->  
      
  </portType>  
    
  <binding>  
    <!-- протоколы связи, используемые web-сервисом -->  
    <!-- форматы сообщений и детали протокола для каждого порта -->  
      
  </binding>  
    
  <service>  
    <!— набор портов связанных с сервисом  -->  
  </service>  
   
</definition>

**ASP LAB 5**  
1. **WCF:** определение WCF-сервиса, коммуникационная модель, WCF-контракты, WCF-хостинг, конечные точки, стандартные привязки, основные отличия от ASMX-сервисов, поведение  и безопасность WCF-сервиса, порядок разработки WCF/RPC и WCF/REST-сервисов, разработка WCF-сервиса с несколькими конечными точками.  
  
1. **WCF:** Windows Communication Foundation; технология основанная на .NET FRAMEWORK для разработки приложений SOA-архитектуры, первая версия 2006 (Indigo).  
2. **WCF:** основные принципы  
-   разработка сервиса должна быть простой и иметь способность к расширению его функциональных возможностей;  
-   один API для всех коммуникационных протоколов;  
-   сервис должен быть интероперабельным (функционировать по отрытым телекоммуникационным стандартам);  
-   сервис должен поддерживать стандарты WS-\*;  
-   сервис должен поддерживать REST, RPC и др.   
Коммуникационая модель

**WCF:** контракты  
-   контракт службы;  
-   контракт данных;  
-   контракт сообщений.   
  
1. **WCF:** хост**–** контейнер для сервиса (любое С#-приложение). Основное назначение хоста: присоединить WCF-инфраструктуру, создать список конечных точек.  
1. **WCF:** конечная точка – адрес, привязка, контракт.  
1. **WCF:** привязка – инкапсулирует все технологии, позволяющие переправить сообщение от одной конечной точке к другой (транспорт, кодировка, протоколы (безопасность,сессия), правила обмена сообщениями (размер, timeout)).

Основные отличия от ASMX:  
 ⁃ Поддерживает больше транспортных сред http, tcp, msmq, namedpipe  
 ⁃ Более широкие возможности обеспечения безопасности  
 ⁃ Более расширяемы и масштабируемы  
 ⁃ Предоставляет более высокий уровень абстракции за счет контрактов  
 ⁃ Легко интегрируется с EntityFramework и Linq  
  
Привязки для обеспечения безопасности:  
1. **WCF:** привязка: безопасность:WS-ReliableMessaging;  
2. **WCF:** привязка: безопасность:WS-Security  
  
1. **WCF:** привязка:кодирование:  
-   текст (ASCII, UTF-8, UTF-16);  
-   двоичная (проприетарные алгоритмы);  
-   МТОМ.   
2. **WCF:** привязка:транспорт:  
-   HTTP;  
-   TCP;  
-   IPC Named Pipes;  
-   MSMQ;  
-   Custom.

4 лаба  
Asmx   
Soap, структура сообщения  
Связывание soap и http   
Rpc это  
Wsdl, структура (definitions, types, message и т.д., зачем эти основные элементы нужны)  
4 технологии, на которых основан веб-сервис (xml, soap, wsdl, uddi)   
Binding в wsdl   
Способы создания asmx приложения (1. Создаём класс, а на его основе генерируется wsdl; 2. Из готовой wsdl схемы генерируем прокси-класс)  
Структура эндпоинта (это в wsdl)  
Структура uddi  
Что такое профиль веб службы, пояснить тот, что написан у вас (атрибут WebServiceBinding класса Simplex, у меня это профиль BasicProfile1\_1)  
Что такое прокси класс, как его сгенерировать с помощью утилиты wsdl.exe (это я демонстрировала при ней)  
Где найти wsdl схему (в браузере открываем веб-службу (Simplex наш) и дописываем ?wsdl в url, это в лекции Смелова есть)  
Как из винформы использовать службу в том же решении (в контекстном меню connected services винформ приложения нужно добавить ссылку на службу; это тоже при ней показывала)

ASMX - ASMX предоставляет возможность создавать веб-службы, которые отправляют сообщения с помощью протокола SOAP. SOAP — это независимый от платформы и независимый от языка протокол для создания веб-служб и доступа к ней.  Основана на XML, WSDL, SOAP.

SOAP - Simple Object Access Protocol – простой протокол доступа к объекту. Первоначально - реализация RPC, расширение протокола XML-RPC, позже - произвольный обмен XML-сообщениями. Используется над HTTP, может также SMTP,TCP,UDP.

Версии: 1.1,1.2

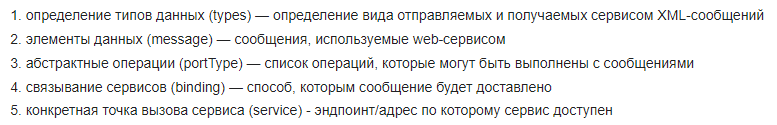
Структура сообщения:

* Корневой элемент *с именем Envelope* , который идентифицирует XML-документ как сообщение SOAP.
* Необязательный элемент *Header* , содержащий сведения для конкретного приложения, например данные проверки подлинности. Если элемент *Header* присутствует, он должен быть первым дочерним элементом элемента *Envelope* .
* Обязательный элемент *Body* , содержащий сообщение SOAP, предназначенное для получателя.
* Необязательный элемент *Fault* , используемый для указания сообщений об ошибках. Если элемент *Fault* присутствует, он должен быть дочерним элементом элемента *Body* .

RPC – Remote Procedure Call – технология для создания распределенных клиентских и серверных программ. Наиболее популярны JSON и XML форматы. Формат определяется сервером и устанавливается для всех процедур. Позволяет вызывать удаленные функции и процедуры так, как будто они вызываются локально.

WSDL – язык описания веб сервисов, основанный на языке xml.

Структра wsdl документа:



Структура Endpoint (WSDL 2)  
  
<endpoint name="SoapEndpoint" binding="m:SoapBinding"  
            address="<http://ws.accurateweather.example/set>"/>  
  
 • <endpoint>: Это элемент, представляющий конечную точку веб-службы.  
 • name="EndpointName": Это атрибут, который указывает имя данной конечной точки. Имя конечной точки может быть уникальным в контексте службы.  
 • binding="BindingName": Этот атрибут указывает, какая привязка будет использоваться для данной конечной точки. Привязка определяет протоколы, которые будут использоваться для взаимодействия с конечной точкой (например, HTTP, TCP, и т. д.).  
 • address="EndpointAddress": Здесь указывается адрес (URL) конечной точки, по которому клиенты смогут обращаться к службе. Этот адрес может быть в виде URL-адреса, URI или другой формы, зависящей от используемой привязки.

**UDDI — Universal Description, Discovery and Integration**  
  
Задача UDDI — предоставить механизм для обнаружения веб-сервисов. UDDI задает бизнес-реестр, в котором провайдеры веб-сервисов могут регистрировать сервисы, а разработчики — искать необходимые им сервисы. Компании IBM, Microsoft и Ariba создали собственные UDDI-реестры (в скором времени эти реестры будут объединены в веб-реестр), в одном из которых разработчики могут зарегистрировать свои веб-сервисы, после чего данные будут автоматически реплицированы в другие реестры.  
  
UDDI базируется на элементах четырех типов: Business Entity, Business Service, Binding Template и Technology Model.  
  
Элемент Business Entity описывает индустрию, предоставляющую данный веб-сервис. Этот элемент может включать описания категорий для данной индустрии, облегчающие более детальный поиск сервисов.  
  
Business Service — это класс сервисов в рамках определенной отрасли промышленности или услуг. Каждая отрасль принадлежит определенному элементу Business Entity.  
  
Вместе Binding Template и Technology Model определяют веб-сервис. Technology Model содержит абстрактное описание, а Binding Template — конкретную спецификацию сервиса. Каждый элемент Binding Template принадлежит определенному элементу Business Service, но несколько элементов Binding Template могут ссылаться на один элемент Technology Model.  
  
Бизнес-реестр UDDI сам является SOAP веб-сервисом. Он поддерживает операции создания, модификации, удаления и поиска элементов всех четырех рассмотренных выше типов.

Профиль

BasicProfile  
  
WS-I Basic Profile 1.1 - это набор стандартов и рекомендаций, разработанный Всемирной инициативой веб-сервисов (Web Services Interoperability Organization, WS-I), с целью обеспечения совместимости и интероперабельности между веб-сервисами, разработанными на основе протокола SOAP (Simple Object Access Protocol) и других важных веб-сервисных стандартов.  
WS-I была создана с целью устранения различных интерпретаций и неясностей в спецификациях SOAP, WSDL (Web Services Description Language) и UDDI (Universal Description, Discovery, and Integration), что позволило разработчикам создавать веб-сервисы, которые могли взаимодействовать между собой более надежно.  
WS-I Basic Profile 1.1 определяет набор требований и рекомендаций, включая следующие:  
 •   Использование XML 1.0 для представления данных в SOAP-сообщениях.  
 •   Определение структуры SOAP-сообщений и использование стандартных элементов в заголовке и теле сообщения.  
 •   Обязательное использование XML-схем для определения формата данных, передаваемых через веб-сервис.  
 •   Указание правил кодирования данных, включая обработку дат, чисел и строк.  
 •   Определение обязательных HTTP-заголовков, таких как "Content-Type" и "SOAPAction", для правильной передачи SOAP-сообщений через HTTP.

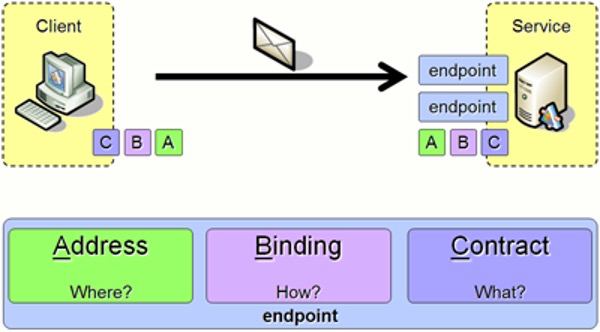
5 лаба

1. **WCF:** определение WCF-сервиса,
2. коммуникационная модель,
3. WCF-контракты,
4. WCF-хостинг,
5. конечные точки,
6. стандартные привязки,
7. основные отличия от ASMX-сервисов,
8. поведение и безопасность WCF-сервиса,
9. порядок разработки WCF/RPC и WCF/REST-сервисов,
10. разработка WCF-сервиса с несколькими конечными точками.

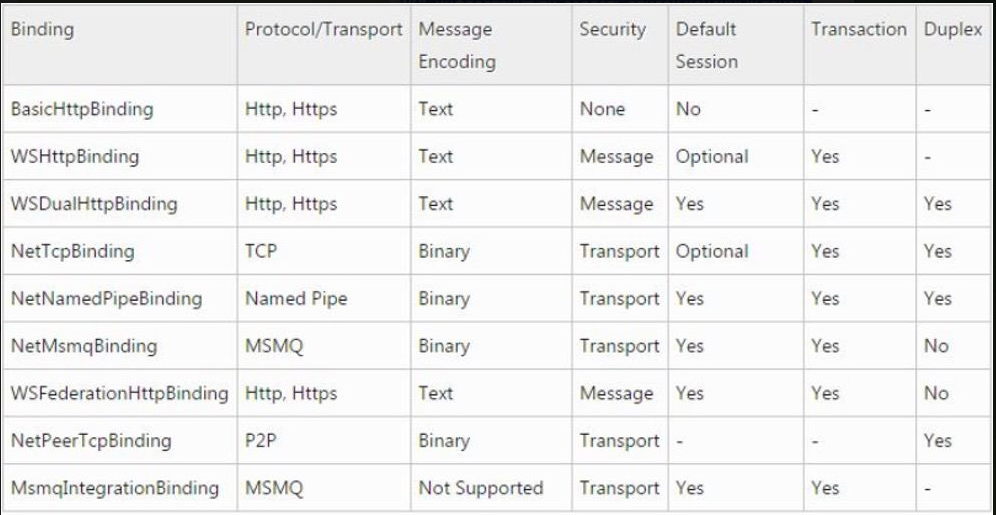
1. Windows Communication Foundation (WCF) — это платформа для создания приложений, ориентированных на обслуживание. С помощью WCF можно отправлять данные в виде асинхронных сообщений из одной конечной точки службы в другую.

**WCF:** основные принципы  
-   разработка сервиса должна быть простой и иметь способность к расширению его функциональных возможностей;  
-   один API для всех коммуникационных протоколов;  
-   сервис должен быть интероперабельным (функционировать по отрытым телекоммуникационным стандартам);  
-   сервис должен поддерживать стандарты WS-\*;  
-   сервис должен поддерживать REST, RPC и др.

2. Коммуникационная модель



3. **WCF:** контракты  
-   контракт службы;  
-   контракт данных;  
-   контракт сообщений.   
4.  
1. **WCF:** хост**–** контейнер для сервиса (любое С#-приложение). Основное назначение хоста: присоединить WCF-инфраструктуру, создать список конечных точек.  
1. **WCF:** конечная точка – адрес, привязка, контракт.  
1. **WCF:** привязка – инкапсулирует все технологии, позволяющие переправить сообщение от одной конечной точке к другой (транспорт, кодировка, протоколы (безопасность,сессия), правила обмена сообщениями (размер, timeout)).



7. Основные отличия от ASMX:  
 ⁃ Поддерживает больше транспортных сред http, tcp, msmq, namedpipe  
 ⁃ Более широкие возможности обеспечения безопасности  
 ⁃ Более расширяемы и масштабируемы  
 ⁃ Предоставляет более высокий уровень абстракции за счет контрактов  
 ⁃ Легко интегрируется с EntityFramework и Linq

Привязки для обеспечения безопасности:  
1. **WCF:** привязка: безопасность:WS-ReliableMessaging;  
2. **WCF:** привязка: безопасность:WS-Security  
  
1. **WCF:** привязка:кодирование:  
-   текст (ASCII, UTF-8, UTF-16);  
-   двоичная (проприетарные алгоритмы);  
-   МТОМ.   
2. **WCF:** привязка:транспорт:  
-   HTTP;  
-   TCP;  
-   IPC Named Pipes;  
-   MSMQ;  
-   Custom.

Безопасность:

