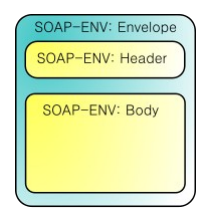
1. **SOAP:** определение, структура Envelop-сообщений, пространства имен, роли, принципы расширения.

Simple Object Access Protocol – простой протокол доступа к объекту.первоначально - реализация RPC, расширение протокола XML-RPC, позже - произвольный обмен XML-сообщениями. чаще всего используется над HTTP, но не обязательно**.** спецификация поддерживается W3C, последняя версия 1.2, в версии 1.2. аббревиатура SOAP не расшифровывается.

1. формат сообщений: SOAP Envelope, SOAP Header (необязательный), SOAP Body, SOAP Fault.

+



Структура Envelop-сообщения может включать следующие элементы:

1. Заголовок (Header): включает метаданные о сообщении, такие как идентификатор сообщения, тип сообщения, время создания и другие параметры, которые могут быть полезны при обработке сообщения.

2. Тело (Body): содержит данные сообщения, которые передаются между сервисами для обработки. Это может быть любая информация, включая запросы, ответы, события и т. д.

3. Футер (Footer): может содержать дополнительные метаданные или контрольные суммы для проверки целостности сообщения или аутентификации.

Envelop-сообщения могут быть организованы в пространствах имен, чтобы группировать сообщения, связанные с определенными областями или функциональностями.

**Структура SOAP-СООБЩЕНИЯ**

<Envelope> - корневой тег, кот опред сообщение и пр-во имет, исп-ное в доке. Обязателен толко тег <body> (это запрос от К к С, предполаг что будет вызыв процедура GetFoto). Есть еще <soap:header> (служ инфа для программистов), <soap:Fault> (ошибки)

Префиксы пространства имен "env", "enc", "rpc", "rep", "xop" и "xmime", используемые в разделах prose этого документа, связаны с именами пространства имен

Пространства имён

Пространство имён — это конструкция, группирующая в себе элементы и параметры (атрибуты) с целью реализации структурной модели и обеспечения уникальности элемента и атрибута, что позволяет избежать конфликта имён.  
  
Для объявления пространства имён используется запись в формате:

<тег xmlns:приставка="URI пространства имён">

Приставку (он же префикс) можно указывать любой, так как она действует локально внутри текущего документа. Но есть общепринятые символьные коды приставок, соотвествующие определённому пространству имён (см. таблицу ниже).  
  
Если не указывать приставку, тогда заданное пространство будет считаться стандартным для всех элементов, где оно (пространство имён) принудительно не указано.  
  
Пример объявления:

<корневой-тег xmlns:tns=" http://test.ru/vb\_chat.wsdl" <!-- подключаем пространство tns -->

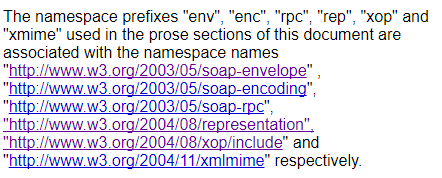
xmlns:xsd="http://www.w3.org/2000/10/XMLSchema" <!-- подключаем пространство xsd -->

xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" <!-- подключаем пространство soap -->

xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" <!-- подключаем пространство soapenc -->

xmlns="http://schemas.xmlsoap.org/wsdl/"> *<!-- пространство имён по-умолчанию -->*

1. **SOAP:** стандартные пространства имен

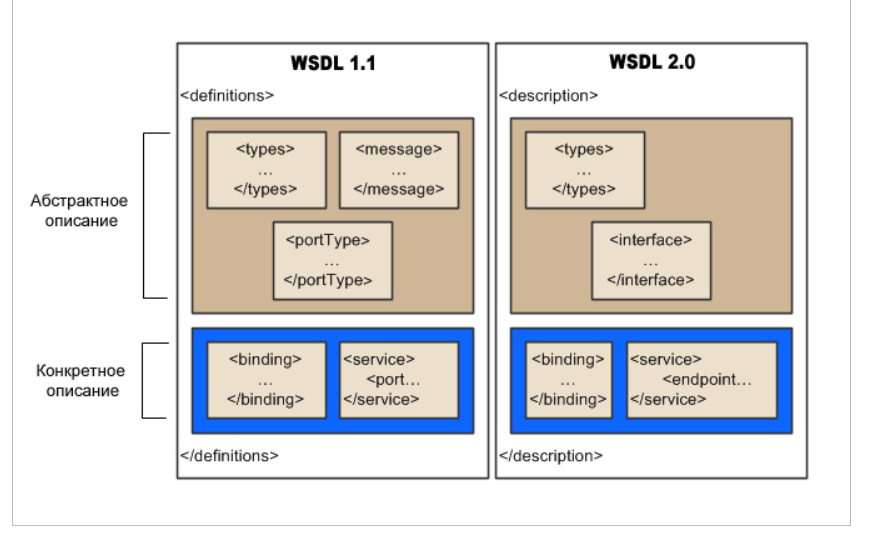


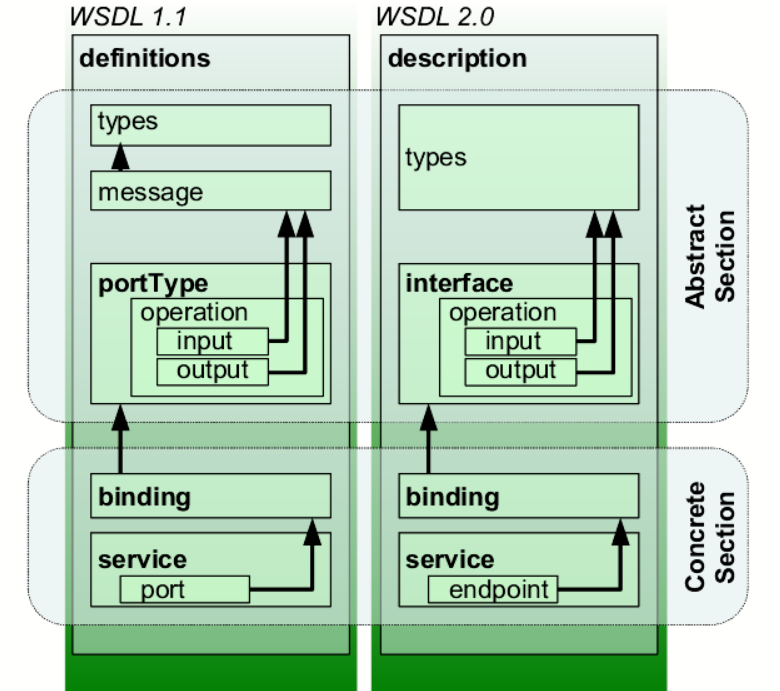
1. **SOAP: Processing Model**
2. **SOAP:** узел – компонент SOA-архитектуры, который может отправлять или получать SOAP-сообщения.
3. **SOAP:** роль - набор правил, определяющих поведение узла, задается атрибутом env:role.
4. **SOAP:** тристандартные роли для узла (нет роли, промежуточный узел, конечный получатель), роль может быть пользовательской (задаваться собственным URI)
5. **SOAP:** роль может быть указана во всех блоках и указывает кому предназначена информация блока. Обычно, обработанный блок извлекается из конверта, если не указано другого.
6. **SOAP:** атрибут **env: mustUnderstand = true** требует обязательную обработку; если обработчик не разобрал блок, то обработка прерывается и формируется сообщение с блоком Fault.
7. **SOAP:** атрибут **env: mustUnderstand = false**, если обработчик не разобрал блок, то ошибка может быть проигнорирована.
8. **SOAP:** атрибут **env: relay = true**, если обработчик не разобрал блок, то блок передается дальше.
9. **SOAP:** атрибут **env: relay = false**, если обработчик не разобрал блок, то блок не передается дальше.

**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.

1. **WSDL:** определение, стандарты и версии, концептуальная модель, пространства имен, структура WSDL-документа.
2. **Web Services Description Language**
3. **WSDL:** разработанаIBM (NASSL), Microsoft (SCL), webMethods (WIDL).

**WSDL:** средство XML-описания интерфейса для доступа к web-службе; платформонезависимое описание;

1. **WSDL:** версия 1.1(2001), **версия 2.0**(2007, статус рекомендации).
2. **Стандарты и версии WSDL:**
3. **- WSDL 1.1: Определен стандартный формат для описания веб-сервисов.**
4. **- WSDL 2.0: Внесено множество улучшений и дополнений, позволяющих более гибко описывать веб-сервисы.**
5. **- WSDL 1.2: Основан на WSDL 1.1, но включает дополнительные функциональные возможности для работы с веб-сервисами.**
6. **1.1, 2.0:**
7. ****

****

Пространства имён для WSDL

По аналогии с предыдущим описанием, пространства имён подключаются через параметр xmlns.  
  
Пример подключения:

<definitions name="vb\_chat"

targetNamespace="http://test.ru/vb\_chat.wsdl" <!-- указываем адрес собственного пространства имён, к которому относится текущий файл -->

xmlns:tns=" http://test.ru/vb\_chat.wsdl" <!-- подключаем пространство tns -->

xmlns:xsd="http://www.w3.org/2000/10/XMLSchema" <!-- подключаем пространство xsd -->

xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" <!-- подключаем пространство soap -->

xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" <!-- подключаем пространство soapenc -->

xmlns="http://schemas.xmlsoap.org/wsdl/"> *<!-- пространство имён по-умолчанию -->*

В примере есть новый параметр targetNamespace, который служит для указания URI адреса пространства имён, к которому относится текущий документ. Важно не путать назначение targetNamespace и xmlns - первый относит файл к пространству имён, а второй (xmlns) служит для работы с пространствами имён внутри файла.

Структура WSDL-документа:

1. Определение пространств имен и их привязка к префиксам.

2. Определение элемента Definitions с указанием версии WSDL, информации о имени и пространстве имен.

3. Определение элемента Types, который содержит описания типов данных с использованием XML Schema.

4. Определение элемента Message для каждого сообщения, с указанием его структуры и типов данных.

5. Определение элемента PortType, который содержит определение доступных операций и ссылки на соответствующие сообщения.

6. Определение элемента Binding, который связывает операции из элемента PortType с протоколами передачи данных и форматами сообщений.

7. Определение элемента Port, который указывает конечную точку доступа (URL) и протокол передачи данных для конкретного привязки.

8. Определение элемента Service, который связывает конечные точки доступа с конкретными протоколами и портами.

1. **ASMX:** определение ASMX-сервиса, порядок разработки, принципы применения, утилита WSDL.EXE.
2. **Active Server Method Extended;** технология Microsoft для разработки web-сервисов(2002, 2007), основанная на XML, WSDL, SOAP; официальное название в MSDN «XML Web Services»; легкая технология: для работы с ней не обязательно знать XML, SOAP и WSDL.
3. **ASMX:** WSE(Web Services Enhancements – улучшение)- дополнение к ASMX основанное на спецификациях WS-\*.
4. **ASMX:** поглощена WCF в 2007г., но поддерживается в виду ее популярности.
5. **ASMX:** Visual Studio, к любому проекту добавить ASM-службу

утилита WSDL.EXE, генерация PROXY

Принципы применения ASMX-сервисов:

1. Интероперабельность: ASMX-сервисы используют открытые стандарты (XML, SOAP, HTTP), что позволяет клиентам разных платформ и технологий взаимодействовать с веб-сервисом.

2. Расширяемость: ASMX-сервисы могут быть легко расширены и изменены с помощью добавления новых операций или параметров.

3. Использование существующей инфраструктуры: ASMX-сервисы могут использовать существующую HTTP-инфраструктуру и развертываться на стандартных веб-серверах.

Утилита WSDL.EXE (Web Services Description Language) - это инструмент командной строки, предоставляемый .NET Framework для генерации WSDL-документов на основе типов данных и интерфейсов .NET. Она позволяет легко создавать и обновлять WSDL-документы для ASMX-сервисов. Утилита WSDL.EXE может быть использована для автоматической генерации клиентского кода на основе WSDL-документа.