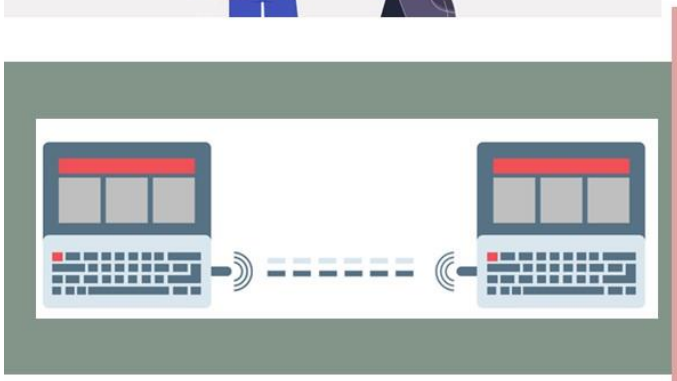
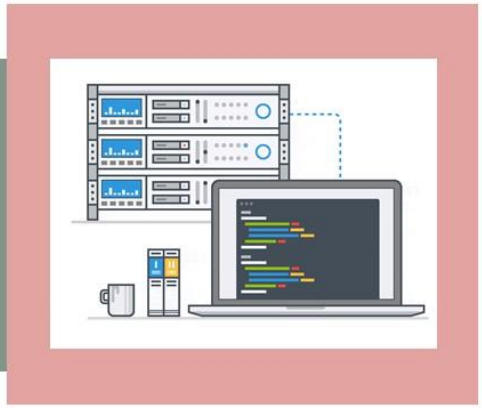
SOAP





SOAP

Представьте что у вас реализована или реализуется некая система, которая должна быть доступна извне. Т.е. есть некий сервер, с которым вам надо общаться. Например веб-сервер.

Этот сервер может выполнять множество действий, работать с базой, выполнять какие-то сторонние запросы к другим серверам, заниматься каким-то вычислениями и т.д. жить и возможно развиваться по ему известному сценарию (т.е. по сценарию разработчиков). С таким сервером общаться человеку неинтересно, потому что он может не уметь/не хотеть отдавать красивые странички с картинками и прочим юзер-френдли контентом. Он написан и работает чтобы работать и выдавать на запросы к нему данные, не заботясь, чтоб они были человекочитаемые, клиент сам с ними разберется.

Другие системы, обращаясь к этому серверу уже могут распоряжаться полученными от этого сервера данными по своему усмотрению - обрабатывать, накапливать, выдавать своим клиентам и т.д.

Ну вот, один из вариантов общения с такими серверами - это SOAP. SOAP протокол обмена xml-сообщениями.

SOAP

SIMPLE

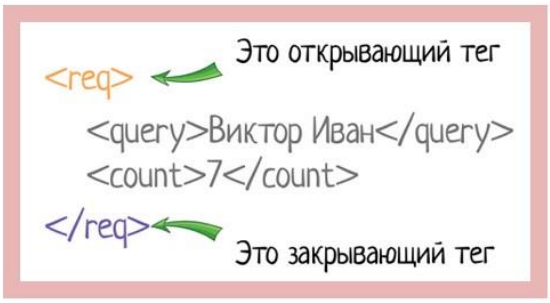
OBJECT

ACCESS

PROTOCOL

Давайте разберем что такое SOAP

Simple Object Access Protocol - протокол доступности объектов. Это называлось протоколом версии SOAP 1.1. Данное наименование отражает его значимость - обращение к различным методам для удаленных объектов или, другими словами, это определение того, как веб-сервисы взаимодействуют друг с другом или взаимодействуют с клиентскими приложениями, которые их вызывают.



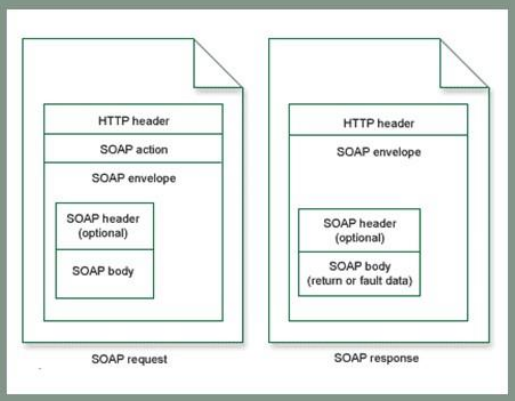
В самом общем виде SOAP это протокол обмена структурированными XML сообщениями в произвольной распределенной вычислительной среде. Отсюда следует, что мы должны куда-то передать XML и получить назад также XML.

Для передачи нам нужен транспорт. Или протокол передачи данных. Например SMTP, FTP, HTTP, HTTPS. Чаще всего HTTP, но это не принципиально.

В самом общем виде SOAP это протокол обмена структурированными XML сообщениями в произвольной распределенной вычислительной среде. Отсюда следует, что мы должны куда-то передать XML и получить назад также XML. Для передачи нам нужен транспорт. Или протокол передачи данных. Например SMTP, FTP, HTTP, HTTPS. Чаще всего HTTP, но это не принципиально.

SOAP используют HTTP как транспортный протокол, в то время как REST базируется на нем. Это означает, что все существующие наработки на базе протокола HTTP, такие как кеширование на уровне сервера, масштабирование, продолжают так же работать в REST архитектуре, а для SOAP необходимо искать другие средства. Взамен этого SOAP сервисы получают такое мифическое свойство, как возможность работать с любым протоколом транспортного уровня вместо HTTP

SOAP не может отличить вызовы от процедур и ответов, и его возможности включают определение форматов сообщений в виде конкретного XML-документа. Сообщение может содержать информацию о вызовах процедур, ответах, запросах и так далее. Сообщение SOAP - это документ XML, информация которого складывается из трех основных элементов: конверта, заголовка и тела.

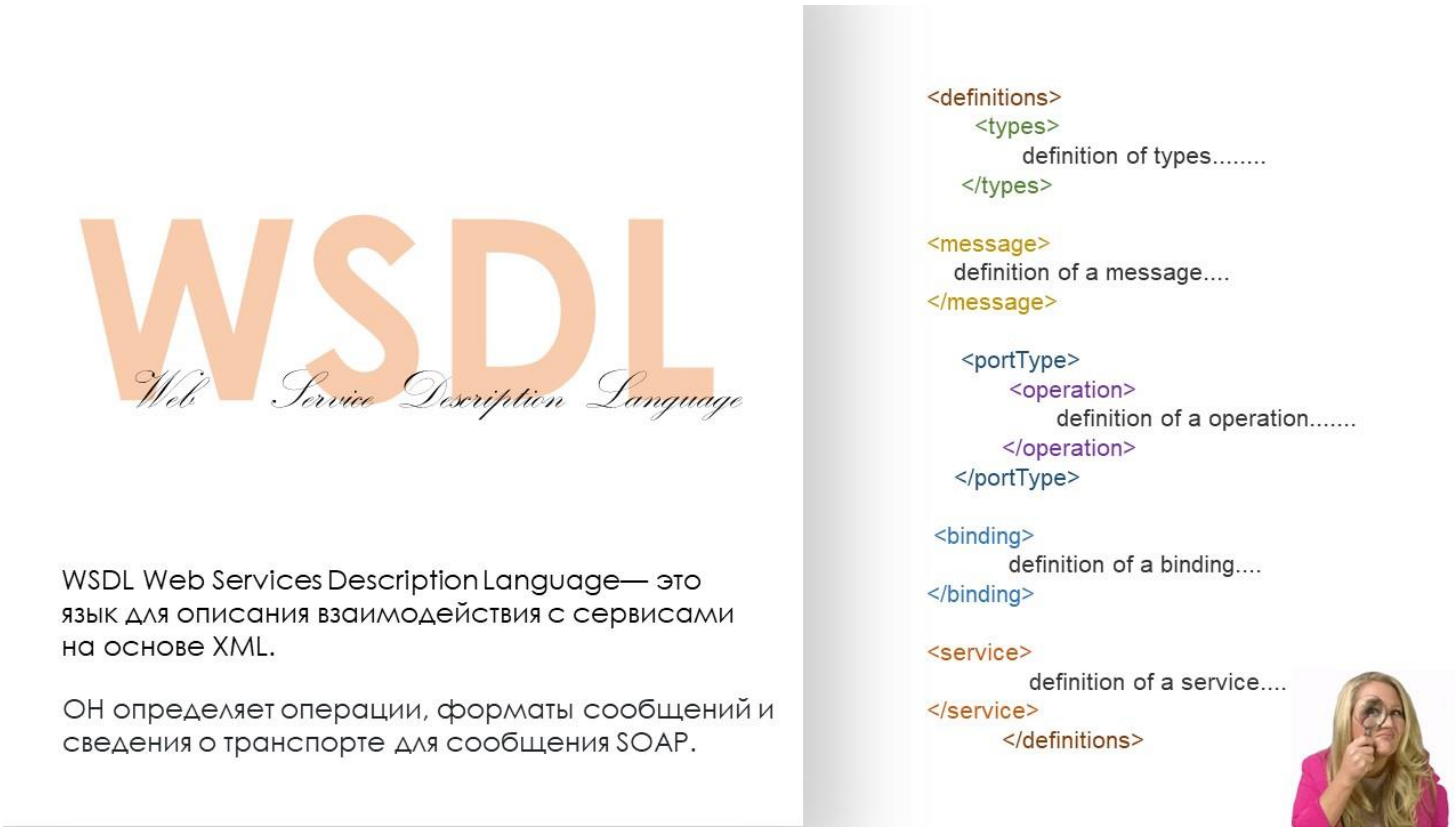


SOAP протокол web API сервисов. То есть способ взаимодействия Вашей информационной системы через web с другими информационными системами.

Сообщение SOAP - это документ XML, информация которого складывается из трех основных элементов: конверта, заголовка и тела. Сообщение может содержать информацию о вызовах процедур, ответах, запросах и так далее.

- Специфика SOAP — это формат обмена данными. С SOAP это всегда SOAP-XML, который представляет собой XML, включающий:

— Envelope (конверт) – корневой элемент, который определяет сообщение и пространство имен, использованное в документе,
— Header (заголовок) – содержит атрибуты сообщения, например: информация о безопасности или о сетевой маршрутизации,
— Body (тело) – содержит сообщение, которым обмениваются приложения,
— Fault – необязательный элемент, который предоставляет информацию об ошибках, которые произошли при обработке сообщений. И запрос, и ответ должны соответствовать структуре SOAP.



Веб-сервис (так называется то, что предоставляет сервер и то, что используют клиенты) дает возможность общения с сервером четко структурированными сообщениями. Дело в том, что веб-сервис не принимает абы какие данные. На любое сообщение, которое не соответствует правилам, веб-сервис ответит ошибкой. Ошибка будет, кстати, тоже в виде xml с четкой структурой (чего нельзя сказать правда о тексте сообщения).

Залогом успеха нашего обмена является способность правильно создать и передать XML запрос и правильно принять и разобрать XML ответ. Для того чтобы сделать все правильно нужно соблюдать (извиняюсь за тавтологию) правила. Эти правила описываются файлом специального формата: WSDL. WSDL - это язык описания правил использования web сервисов в том числе и сервиса SOAP.

WSDL (Web Services Description Language). Правила, по которым составляются сообщения для веб-сервиса описываются так же с помощью xml и также имеют четкую структуру. Т.е. если веб-сервис предоставляет возможность вызова какого-то метода, он должен дать возможность клиентам узнать какие параметры для данного метода используются. Если веб-сервис ждет строку для метода Method1 в качестве параметра и строка должна иметь имя Param1, то в описании веб-сервиса эти правила будут указаны.

В качестве параметров могут передаваться не только простые типы, но и объекты, коллекции объектов. Описание объекта сводится к описанию каждой составляющей объекта. Если объект состоит из нескольких полей, значит описывается каждое поле какой у него тип, название (какие возможные значения). Поля также могут быть сложного типа и так далее пока описание типов не закончится на простых - строка, булево, число, дата...

Впрочем простыми могут оказаться какие-то специфические типы, важно чтоб клиенты могли понять какие значения могут в них содержаться.

Для клиентов достаточно знать url веб-сервиса, wsdl всегда будет рядом, по которому можно получить представление о методах и их параметрах, которые предоставляет этот веб-сервис.

Минусов тоже полно:

- Неоправданно большой размер сообщений. Ну тут сама природа xml такова, что формат избыточный, чем больше тэгов, тем больше неполезной информации. Плюс soap добавляет своей избыточности.
- Автоматическая смена описания веб-сервиса может сломать все клиенты. Ну это как бы для любой системы так, если не поддерживается обратная совместимость со старыми методами, все отвалится...

Web Services Description Language

Определения WSDL описывают, как получить доступ к веб-службе и какие операции она будет выполнять.
WSDL — это язык для описания взаимодействия с сервисами на основе XML.

Собственно WSDL и XSD подробно описывают что и в каком виде слать на сервер, чтобы получить хороший ответ.

WSDL часто используется в сочетании с SOAP и XML-схемой для предоставления веб-сервисов через Интернет. Клиентская программа, подключающаяся к веб-службе, может прочитать WSDL, чтобы определить, какие функции доступны на сервере. Все используемые специальные типы данных встраиваются в файл WSDL в форме XML-схемы. Затем клиент может использовать SOAP для фактического вызова одной из функций, перечисленных в WSDL.

Структура документа WSDL

Основная структура документа WSDL выглядит следующим образом —

Документ WSDL также может содержать другие элементы, такие как элементы расширения и элемент службы, которые позволяют группировать определения нескольких веб-служб в одном документе WSDL.

Request

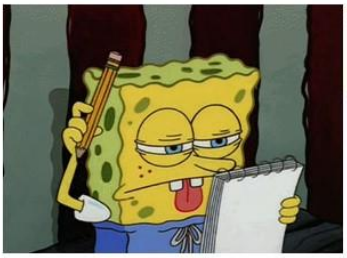
```
<soapenv:Envelope
Xmlns:soapenv="http://schemas.xmlsoap.org/soap/
envelope/
xmls;:wrap=http://foo.bar/wrappersoapserver>
<soapenv:Header/>
   <soapenv:Body>
      <wrap:doRegister>
         <email>nast@gmail.com</email>
         <name>Настюшка</name>
         <password>123eEg</password>
      </wrap:doRegister>
   </soapenv:Body>
</sopenv:Envelope>
```

SoapUI

SoapUI
Supported by SMARTBEAR

SoapUI — приложение с открытым исходным кодом для тестирования веб-сервисов сервис-ориентированных архитектур (SOA) и передачи состояний представлений (REST).

SOAP API



SOAP это протокол, в простонародье - Simple Object Access Protocol (Простой Протокол Доступа к Объектам), А API это Application Programming Interface - программный интерфейс приложения. А интерфейс это совокупность методов и правил взаимодействия. Собственно все вместе - SOAP API - это веб-сервис, созданный на основе SOAP, использующий преимущества создания веб-протоколов, таких как HTTP и его XML, которые работают во всех операционных системах, поэтому его разработчики могут легко манипулировать веб-сервисами и получать ответы, не заботясь о языке и платформах.

Как я уже писала выше, протокол SOAP помогает клиенту и серверу обмениваться достаточно произвольными (но в заранее оговоренной форме) сообщениями в формате XML, причем это обмен не только простыми типами, но и коллекциями, объектами, коллекциями объектов. Т.е. в запросе может быть пять объектов типа "Человек", у каждого объекта несколько полей (пол, возраст, глубина внутреннего мира в метрах и т.п.), часть этих полей будет обязательной, часть нет, но все это будет описано в схеме.

Что хорошего в SOAP? Несколько видов автоматической валидации, которую даже проверять не придется, так как сервис будет падать если что не так (и падать с понятными ошибками). Стандарты - если ты посылаешь подходящий по стандарту запрос, то ты получишь (при условии правильно работающей логики) стандартный ответ.

Что плохого в SOAP? Количество вариантов описания схем - на одном проекте это делается по одному шаблону и вы привыкаете к этому, вы точно знаете, что, как, куда слать. А на другом проекте все будет настолько подругому, что вы почувствуете себя полным джуном.

Что мы проверяем в SOAP - бизнес-логику и то, что схема валидируется сервером (а так же, что она принимает на вход параметры правильного формата). Собственно все, что касается схемы, проверяется на этапе разработки, а после, только бизнес-логика (до того момента, пока опять не начнутся изменения в схеме).

	SOAP	REST
	С SOAP это всегда SOAP-XML, который представляет собой XML	Специфика REST — использование HTTP в качестве транспортного протокола.
Формат обмена сообщениями	XML для запросов и ответов	XML, JSON или любого другого удобного формата
Определения услуг	WSDL	Swagger или Open API
Транспорт	Нет ограничений	HTTP(HTTPS)
Простота реализации	Необходимо определить свой сервис с использованием WSDL, и при обработке и анализе сообщений SOAP-XML возникают большие накладные расходы	REST обычно использует JSON

Специфика SOAP — это формат обмена данными. С SOAP это всегда SOAP-XML, который представляет собой XML

Специфика REST — использование HTTP в качестве транспортного протокола. Он подразумевает наилучшее использование функций, предоставляемых HTTP — методы запросов, заголовки запросов, ответы, заголовки ответов и т. д.

SOAP – это целое семейство протоколов и стандартов, откуда напрямую вытекает, что это более тяжеловесный и сложный вариант с точки зрения машинной обработки. Поэтому REST работает быстрее.

REST обычно использует JSON, который легче анализировать и обрабатывать. В дополнение к этому, REST не требует наличия определения службы для предоставления веб-службы.

Однако в случае SOAP вам необходимо определить свой сервис с использованием WSDL, и при обработке и анализе сообщений SOAP-XML возникают большие накладные расходы.

SOAP не накладывает никаких ограничений на тип транспортного протокола. Вы можете использовать либо Web протокол HTTP, либо MQ.

REST подразумевает наилучшее использование транспортного протокола HTTP

SOAP используют HTTP как транспортный протокол, в то время как REST базируется на нем. Это означает, что все существующие наработки на базе протокола HTTP, такие как кеширование на уровне сервера, масштабирование, продолжают так же работать в REST архитектуре, а для SOAP необходимо искать другие средства. Взамен этого SOAP сервисы получают такое мифическое свойство, как возможность работать с любым протоколом транспортного уровня вместо HTTP, однако практической пользы от него зачастую не больше, чем сотрудникам Челябинского трубопрокатного завода от большого количества статей в википедиях на мертвых языках

- Есть мнение, что разработка RESTful сервисов намного проще. Это не так
- В первом гугловском результате по запросу «REST vs SOAP» акцентируется внимание на том, что ответ REST может быть представлен в различных форматах, а SOAP привязан к XML. Это действительно важный фактор, достаточно представить себе вызов сервиса из javascript, ответ на который мы определенно хотим получать в JSON.
- «REST vs SOAP» можно перефразировать в «Простота vs Стандарты», что проявляется в том, что для SOAP мы имеем протокол WSDL для исчерпывающего описания веб-сервиса, который с использованием все тех же чудесных средств разработки прото-таки волшебным образом делает почти всю работу за нас. Со стороны REST мы имеем загадочный и неиспользуемый протокол WADL, который в принципе и не нужен – он мешает простоте.
- Второй аспект предыдущего пункта – обработка ошибок. В SOAP она полностью стандартизована, а REST может использовать давно известные коды ошибок HTTP (если здесь Вас посетила мысль, что это же очевидно и зачем я это пишу, то значит Вы внимательно читаете статью).
- SOAP работает с операциями, а REST – с ресурсами. Этот факт в совокупности с отсутствием клиентского состояния у RESTful сервисов приводит нас к тому, что такие вещи как транзакции или другая сложная логика должна реализовываться «SOAP-но». Приведу пару примеров на понимание разницы между подходами. Букмекерская контора заказала сервис для работы с футбольной статистикой. Пользовательский функционал – получить список матчей, получить детали о матче. Для редакторов – редактировать (Create, Edit, Delete) список матчей, редактировать детали матча. Для такой задачи однозначно надо выбирать подход REST и получать бенефиты от его простоты и естественности во взаимодействии с HTTP. Не нужны нам здесь SOAP. Нам всего лишь надо реализовать следующее:

Что происходит при обращении с использованием методов HTTP

URL GET POST PUT DELETE

example.com/matches Получаем список матчей Создаем заново список матчей Обновляем список матчей Мстим директору букмекерской конторы

example.com/matches/28 Получаем детали матча с ID=28 Создаем информацию о матчеОбновляем детали матча Удаляем матч

Все очень просто! Теперь пример посложнее. Та же букмекерская контора захотела API для ставок на live матчи. Эта процедура включает в себя многочисленные проверки, например, продолжает ли ставка быть актуальной, не изменился ли коэффициент, не превышена ли максимальная сумма ставки для маркета. После этого происходит денежная транзакция, результаты которой записываются в основную и в резервные базы данных. Лишь после этого клиенту приходит ответ об успешности операции. Здесь явно прослеживается ориентация на операции, имеются повышенные требования к безопасности и устойчивости приложения, поэтому целесообразно использовать SOAP.

Источники

https://ru.stackoverflow.com/questions/257184/%D0%A7%D1%82%D0%BE-%D1%82%D0%B0%D0%BA%D0%BE%D0%B5-soap

https://infostart.ru/public/716743/

https://www.youtube.com/watch?v=C2TMZeRdLKw&t=8s&ab_channel=okiseleva

https://dataart.com.ua/news/rest-i-soap-pochuvstvuyte-raznitsu/

http://webcache.googleusercontent.com/search?q=cache:eid2AqzH1TwJ:https://qahacking.ru/lessons/chto-takoe-soap-api&hl=uk&gl=ua&strip=1&vwsrc=0

https://habr.com/ru/post/524288/

https://wiki.merionet.ru/servernye-resheniya/28/soap-protokol-imeni-mylnoj-opery/

https://coderlessons.com/tutorials/xml-tekhnologii/uznaite-wsdl/wsdl-kratkoe-rukovodstvo