

# WEB SERVICES



[training@infotek-solutions.com](mailto:training@infotek-solutions.com)

# Module agenda

---

1. Intro to business to business communication
2. Give an example why and when we need to exchange data between systems (Expedia, weather on BBC and Yahoo, using ATM machine from different banks)
3. Explain structure of web services
4. Give definition of web services
5. Two types of web services
6. REST (JSON format using as an example how to store books)
7. SOAP (XML format using as an example how to store books. WSDL with structure explanation)
8. Difference between two types of web services (\*) Synchronous and asynchronous web services
9. What to test in web services (format, structure, data, performance, security)
10. Tools to test web services: Postman for REST, SoapUI for both.
11. Create test cases and assertions. As examples use get country by country code for REST and Medicare for SOAP

# Examples why and when we need to exchange data between systems

Expedia

Account My Lists 1 My Trips Support Español 简体中文

Home Bundle and Save Hotels Cars Flights Cruises Things to Do Vacation Rentals Deals Rewards Mobile Collections

Roundtrip One Way

www.bbc.com

Washington, DC, Unit

Nearby airports

1 Traveler, All Airlines, Econon

Select your depart

WASHINGTON DULLES WEATHER

EDIT

FRI -7°C

SAT 4°C

SUN 4°C

MON 3°C

Prices are roundtrip per person, inc.

Stops

Nonstop (1)

1 Stop (19)

Airlines included

American Airlines (3)

British Airways (3)

Delta (3)

United (3)

Alitalia (1)

Brussels Airlines (1)

Finnair (1)

Show all

Departure time - Washington

Morning (5:00am - 11:59am)

Afternoon (12:00pm - 5:59pm)

Evening (6:00pm - 11:59pm)

9:55pm - 5:50pm +1 13h 55m 1 stop 3h 40m in CMN

Royal Air Maroc IAD - FCO

Finalizing prices... Select

11:00pm - 6:50pm +1 13h 50m 1 stop 1h in IST

Turkish Airlines IAD - FCO

Finalizing prices... Select

5:15pm - 12:35pm +1 13h 20m 1 stop 2h 50m in CPH

SAS IAD - FCO

Finalizing prices... Select

5:10pm - 8:10am +1 9h 0m Nonstop

United IAD - FCO

Finalizing prices... Select

YAHOO!

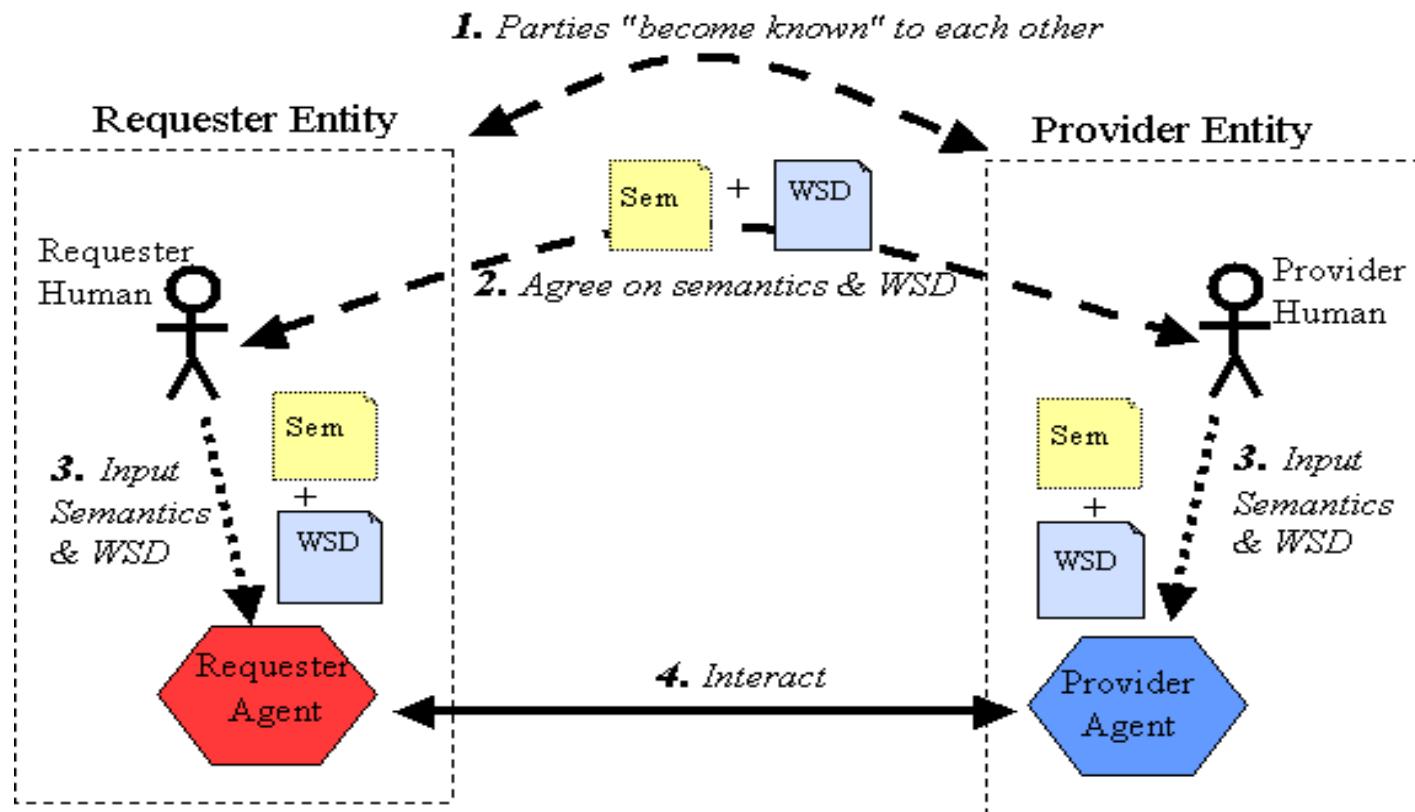
Mail

Herndon, VA

Today Sat Sun Mon

40° 17° 37° 16° 38° 31° 37° 22°

# Intro to business to business communication

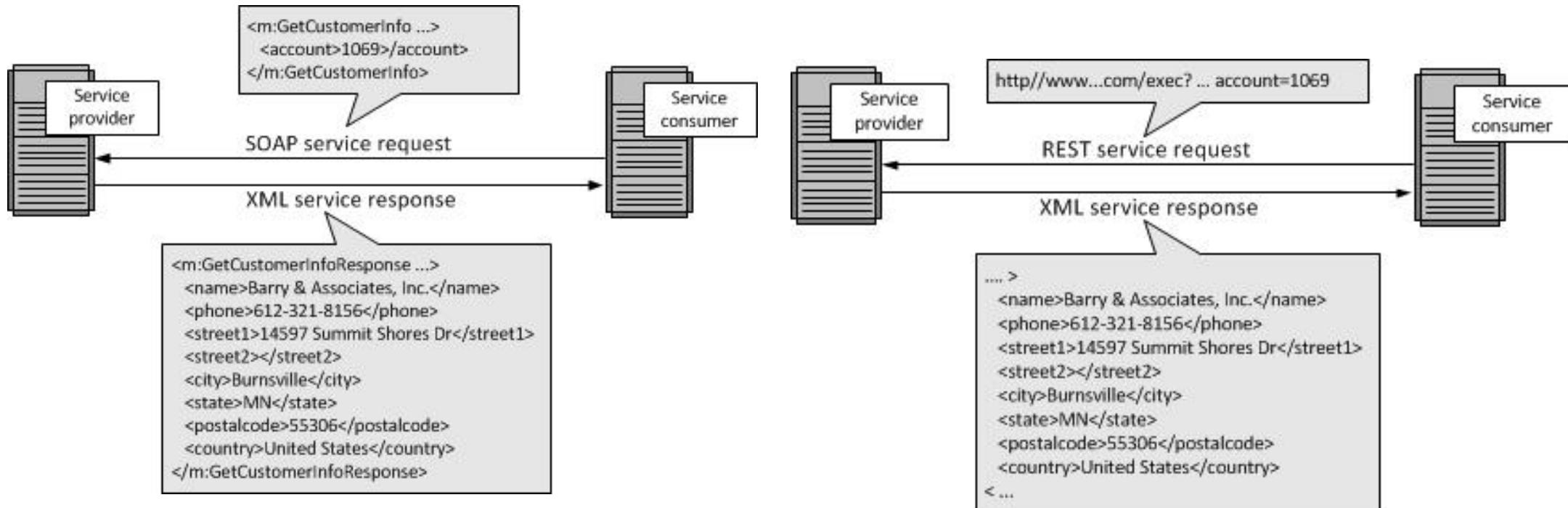


# Give definition of web services

---

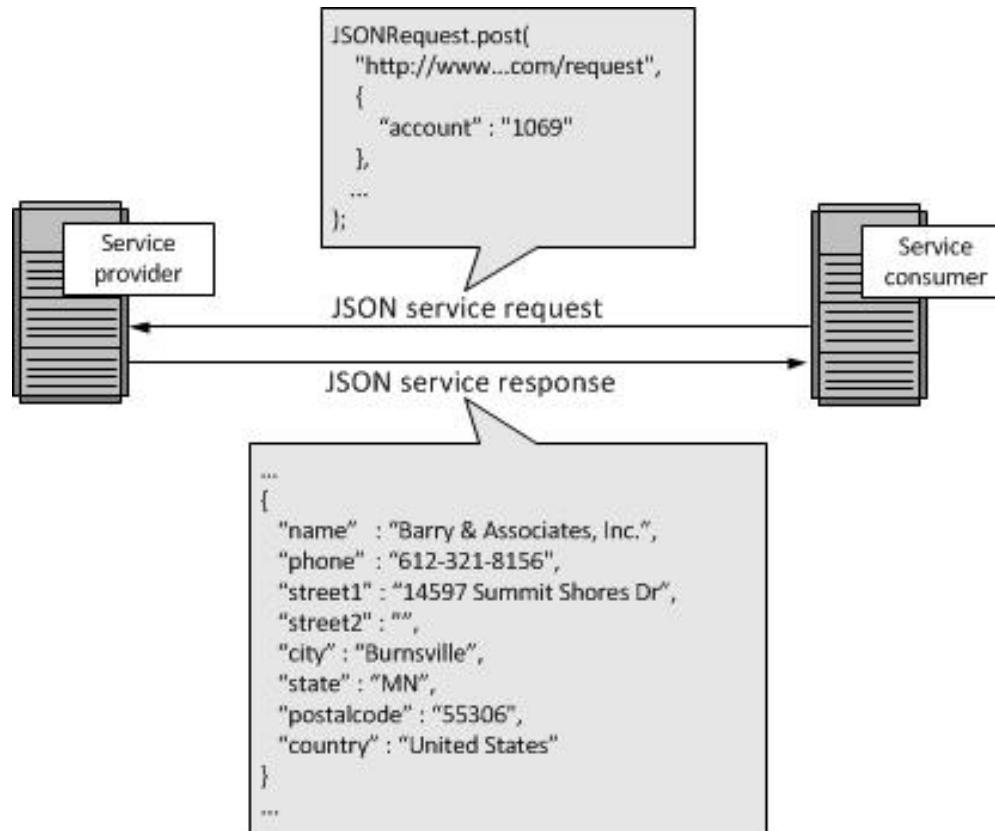
- The term *Web services* describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL standards over an Internet protocol backbone.
- XML is used to tag the data, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI is used for listing what services are available.
- Used primarily as a means for businesses to communicate with each other and with clients, Web services allow organizations to communicate data without intimate knowledge of each other's IT systems behind the firewall.
- Unlike traditional client/server models, such as a Web server/Web page system, Web services do not provide the user with a GUI. Web services instead share business logic, data and processes through a programmatic interface across a network. The applications interface, not the users.
- Web services allow different applications from different sources to communicate with each other without time-consuming custom coding, and because all communication is in XML, Web services are not tied to any one operating system or programming language.
- For example, Java can talk with Perl, Windows applications can talk with UNIX applications.

# Two types of web services



# REST

---

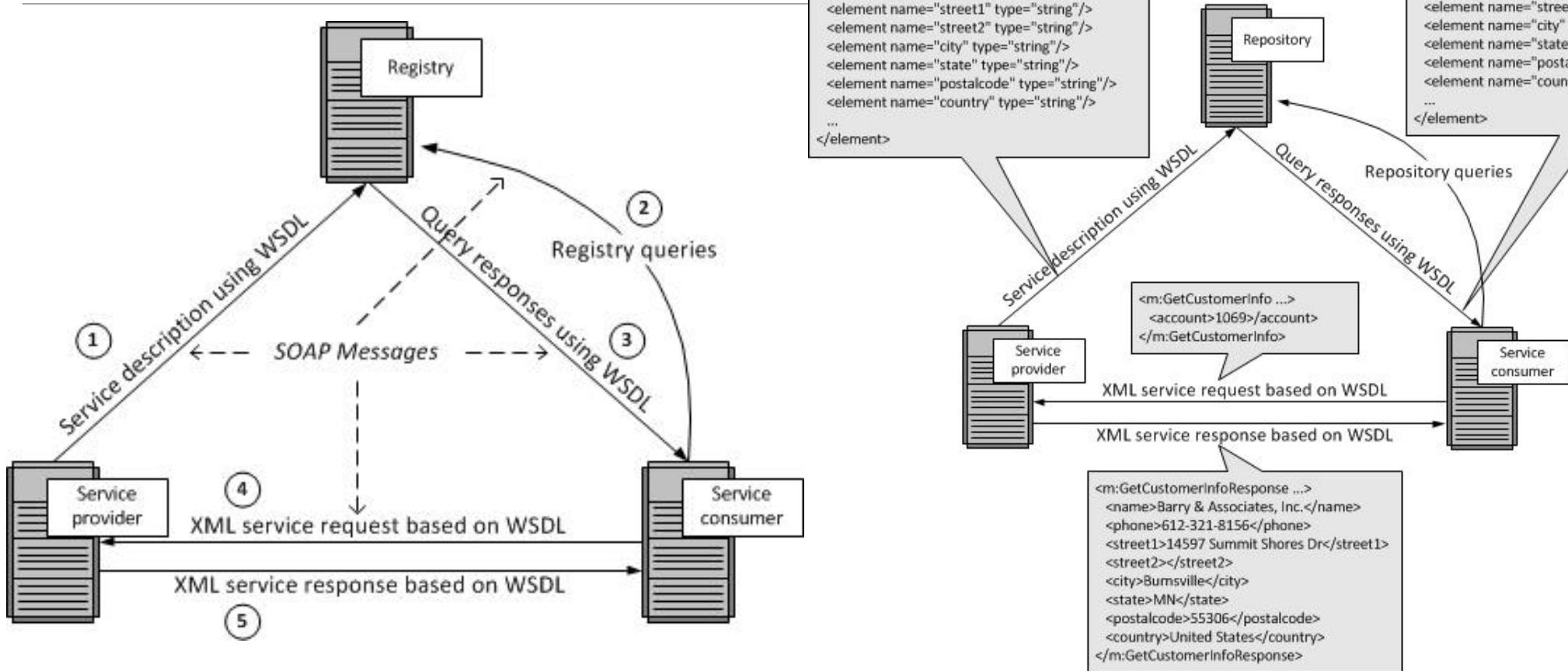


[books.json](#)

Raw

```
1  {
2    "books": [
3      {
4        "isbn": "9781593275846",
5        "title": "Eloquent JavaScript, Second Edition",
6        "subtitle": "A Modern Introduction to Programming",
7        "author": "Marijn Haverbeke",
8        "published": "2014-12-14T00:00:00.000Z",
9        "publisher": "No Starch Press",
10       "pages": 472,
11       "description": "JavaScript lies at the heart of almost every modern web application, from social apps to the newest browser-based games. Eloquent JavaScript, Second Edition, is your guide to the language that powers them. This updated edition features a new chapter on asynchronous programming, and includes a foreword by Mozilla's Brian Grinstead, who discusses how the book has influenced his work on the Firefox developer tools. It's packed with practical examples, and clear explanations of the most important concepts, making it ideal for both novices and experienced developers looking to extend their knowledge of JavaScript.", "website": "http://eloquentjavascript.net/"}
13     },
14     {
15       "isbn": "9781449331818",
16       "title": "Learning JavaScript Design Patterns",
17       "subtitle": "A JavaScript and jQuery Developer's Guide",
18       "author": "Addy Osmani",
19       "published": "2012-07-01T00:00:00.000Z",
20       "publisher": "O'Reilly Media",
21       "pages": 254,
22       "description": "With Learning JavaScript Design Patterns, you'll learn how to write beautiful, structured, and maintainable JavaScript code. By applying design patterns and best practices, you'll learn how to avoid common pitfalls, and how to easily maintain and extend your code over time. You'll also learn how certain patterns can help you work more effectively with the language's core features, such as prototypal inheritance, generators, and closures.", "website": "http://www.addyosmani.com/resources/essentialjsdesignpatterns/book/"}
24   },
25   {
26     "isbn": "9781449365035",
```

# SOAP



# XML format

XML

```
<?xml version="1.0"?>
<catalog>
  <book id="bk101">
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications
      with XML.</description>
  </book>
  <book id="bk102">
    <author>Ralls, Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-12-16</publish_date>
    <description>A former architect battles corporate zombies,
      an evil sorceress, and her own childhood to become queen
      of the world.</description>
  </book>
  <book id="bk103">
    <author>Corets, Eva</author>
    <title>Maeve Ascendant</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-11-17</publish_date>
    <description>After the collapse of a nanotechnology
      society in England, the young survivors lay the
      foundation for a new society.</description>
  </book>
  <book id="bk104">
    <author>Corets, Eva</author>
```

# SOAP (WSDL with structure explanation)

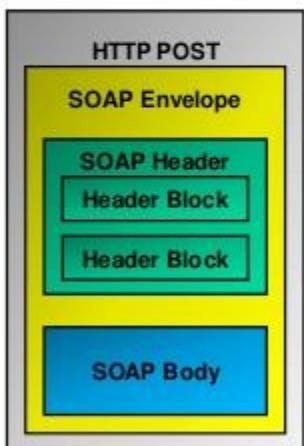
## SOAP - WSDL - UDDI

### 4. SOAP (4/12)

#### SOAP message exchange mechanism (3/5):

SOAP defines a generic structure of an abstract message.

Elements of a SOAP message:



#### SOAP Envelope:

The envelope (=root element of SOAP message) is a for the optional SOAP header and the mandatory SOAP body element.

#### SOAP Header:

The SOAP header may carry control information whi not application payload. Such information is organized in header blocks, each its individual XML namespace defining the schema. The SOAP header is extensible, i.e. arbitrary namesp a particular way of message processing can be "wov the header.

#### SOAP Body:

The SOAP body carries the actual application inform encoded as an XML document. The schema of the bc defined by a WSDL document.

in

## WSDL FILE (FOUR ELEMENTS)

```
<definitions>
  <types>
    data type definitions.....
  </types>

  <message>
    definition of the data being communicated.
  </message>

  <portType>
    set of operations.....
  </portType>

  <binding>
    protocol and data format specification....
  </binding>
</definitions>
```

**Types:** XML data types for all input and output messages in the service

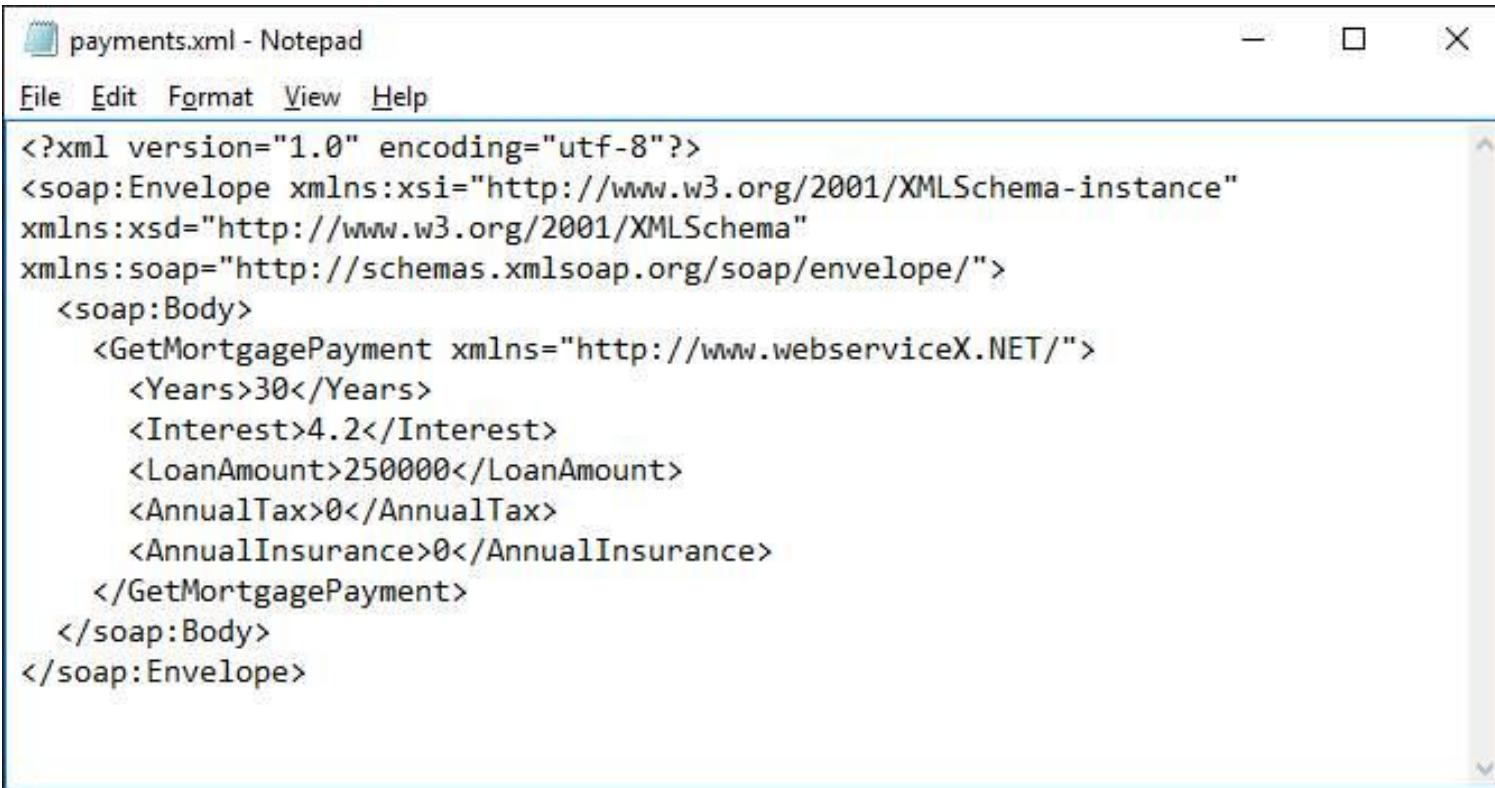
**Message:** Defines the input & output parameters for each operation

**PortType:** Operations (functions) offered by the web service

**Binding:** Defines protocol and data format for each port type

# SOAP (WSDL with structure explanation)

---



The screenshot shows a Windows Notepad window titled "payments.xml - Notepad". The window contains a SOAP XML message. The XML structure includes an envelope with a body containing a specific service operation, GetMortgagePayment, which takes several parameters: Years, Interest, LoanAmount, AnnualTax, and AnnualInsurance.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <GetMortgagePayment xmlns="http://www.webserviceX.NET/">
            <Years>30</Years>
            <Interest>4.2</Interest>
            <LoanAmount>250000</LoanAmount>
            <AnnualTax>0</AnnualTax>
            <AnnualInsurance>0</AnnualInsurance>
        </GetMortgagePayment>
    </soap:Body>
</soap:Envelope>
```

# Difference between two types of web services

---

#	SOAP	REST
1	A XML-based message protocol	An architectural style protocol
2	Uses WSDL for communication between consumer and provider	Uses XML or JSON to send and receive data
3	Invokes services by calling RPC method	Simply calls services via URL path
4	Does not return human readable result	Result is readable which is just plain XML or JSON
5	Transfer is over HTTP. Also uses other protocols such as SMTP, FTP, etc.	Transfer is over HTTP only
6	JavaScript can call SOAP, but it is difficult to implement	Easy to call from JavaScript
7	Performance is not great compared to REST	Performance is much better compared to SOAP - less CPU intensive, leaner code etc.

# Difference between two types of web services

	SOAP	REST
Acronym	Simple Object Access Protocol	REpresentational State Transfer
Architecture	SOA based architecture that uses middleware interoperability.	Architecture style protocol designed for web based communication assuming only point to point communication.
Message Format	XML inside a SOAP Envelope	XML
In business logic	Uses services interfaces to expose the business logic.	Uses URI to expose business logic.
Standards	Defines standards to be strictly followed.	Does not define too much standards like SOAP.
Communication	Use WSDL for communication between consumer and provider.	Uses XML or JSON to send and receive data.
Transport	Transfer is over HTTP, Also uses other protocols such as SMTP, FTP, MIME, JMS, HHTP, etc.	Transfer is over HTTP only.
JavaScript calls	JavaScript can call SOPA but it is difficult to implement.	Easy to call from Javascript.
Security	SOAP defines its own security.	RESTful web services inherits security measures from the underlying transport.
Payload	Payload must support SOAP schema.	Payload can be of any format.
Error Handling	Doesn't support error handling.	Has built-in error handling.
External Calls	Invokes services by calling RPC method.	Simply calls services via URL path.

# What to test in web services

---

- FORMAT
- STRUCTURE
- DATA
- PERFORMANCE
- SECURITY

# Tools to test web services: Postman for REST, SoapUI for both.

---



# Create test cases and assertions. As examples use get country by country code for REST

The screenshot shows the Postman application interface. At the top, there are three tabs in the header bar: "http://services.groupkt.com" (selected), "https://randomuser.me/api", and "http://services.groupkt.com". To the right of the tabs is a dropdown menu set to "No Environment". Below the header, the URL "http://services.groupkt.com/country/get/all" is entered into the search bar. Underneath the search bar, there is a "Send" button. The main workspace contains a "Tests" tab which is currently selected. The test script is written in JavaScript:

```
1 //how elements were return
2 * pm.test("check that answer consists 249 elements", function () {
3     var body = JSON.parse(responseBody);
4     var list = body.length;
5     tests["251"] = typeof responseJson === 'undefined' || list;
6     //console.log(list);
7});
```

To the right of the test script, there is a sidebar with several links:

- Test scripts are written in JavaScript and run after the response is received.
- [Learn more about tests](#)
- SNIPPETS**
- [Clear a global variable](#)
- [Clear an environment variable](#)
- [Get a global variable](#)
- [Get a variable](#)
- [Get an environment variable](#)
- [Response body: Contains string](#)
- [Response body: Convert XML to Object](#)

At the bottom of the interface, there are tabs for "Body", "Cookies (1)", "Headers (12)", and "Test Results (2/2)". The "Test Results" tab is selected and shows two entries:

- All Passed Skipped Failed
- PASS check that answer consists 249 elements
- PASS 251

On the far right, status information is displayed: Status: 200 200 Time: 123 ms.

# HTTP Methods for RESTful Services

---

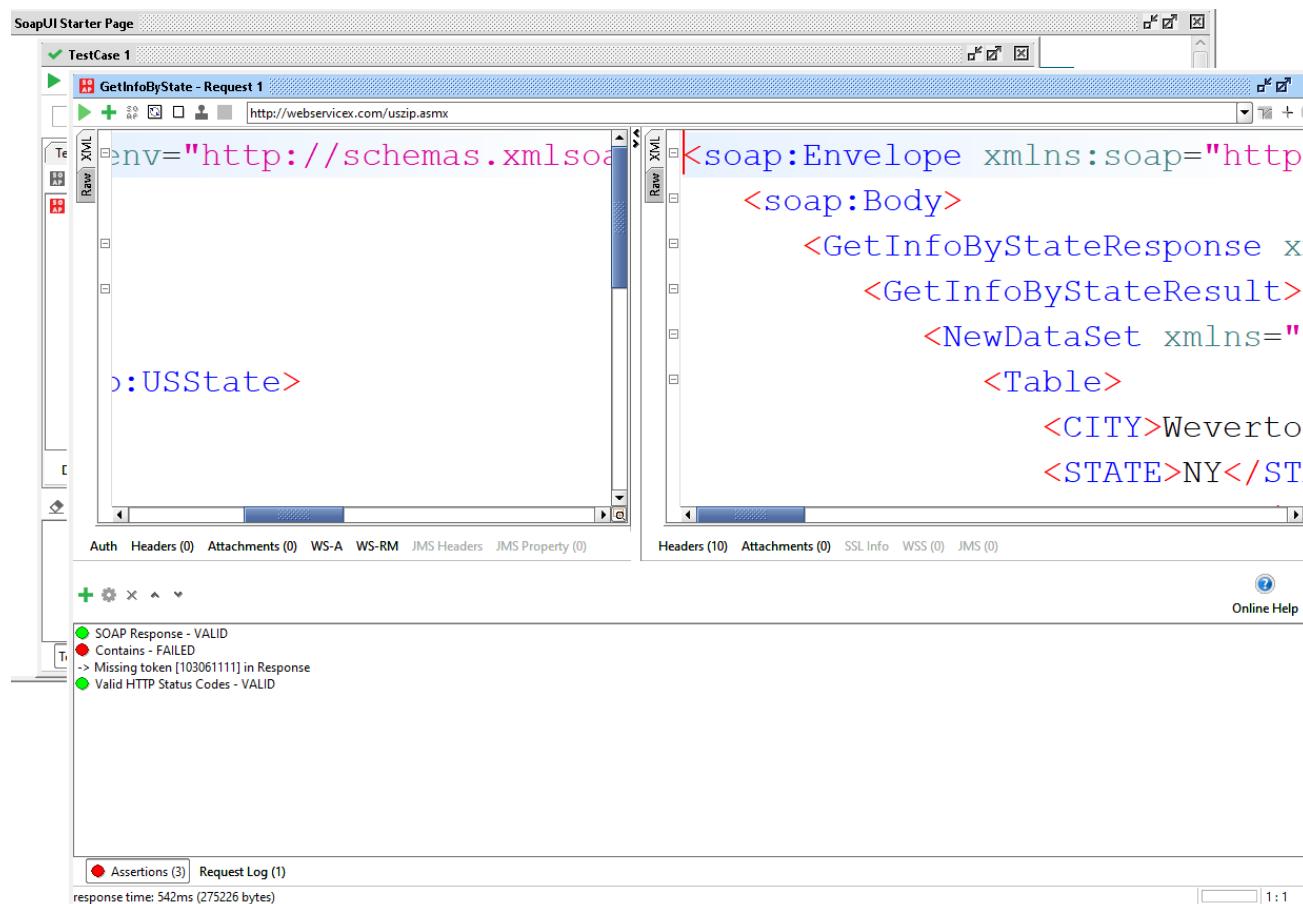
HTTP Verb	CRUD	Entire Collection (e.g. /customers)	Specific Item (e.g. /customers/{id})
POST	Create	201 (Created), 'Location' header with link to /customers/{id} containing new ID.	404 (Not Found), 409 (Conflict) if resource already exists..
GET	Read	200 (OK), list of customers. Use pagination, sorting and filtering to navigate big lists.	200 (OK), single customer. 404 (Not Found), if ID not found or invalid.
PUT	Update/Replace	405 (Method Not Allowed), unless you want to update/replace every resource in the entire collection.	200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid.
PATCH	Update/Modify	405 (Method Not Allowed), unless you want to modify the collection itself.	200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid.
DELETE	Delete	405 (Method Not Allowed), unless you want to delete the whole collection—not often desirable.	200 (OK). 404 (Not Found), if ID not found or invalid.

# HTTP Status Codes

HTTP Status Codes				
For great REST services the correct usage of the correct HTTP status code in a response is vital.				
1xx – Informational	2xx – Successful	3xx – Redirection	4xx – Client Error	5xx – Server Error
This class of status code indicates a provisional response, consisting only of the Status-Line and optional headers, and is terminated by an empty line  100 – Continue 101 – Switching Protocols 102 – Processing	This class of status code indicates that the client's request was successfully received, understood, and accepted.  200 – OK 201 – Created 202 – Accepted 203 – Non-Authoritative Information 204 – No Content 205 – Reset Content 206 – Partial Content 207 – Multi-Status	This class of status code indicates that further action needs to be taken by the user agent in order to fulfill the request.  300 – Multiple Choices 301 – Moved Permanently 302 – Found 303 – See Other 304 – Not Modified 305 – Use Proxy 307 – Temporary Redirect	The 4xx class of status code is intended for cases in which the client seems to have erred.  400 – Bad Request 401 – Unauthorised 402 – Payment Required 403 – Forbidden 404 – Not Found 405 – Method Not Allowed 406 – Not Acceptable 407 – Proxy Authentication Required 408 – Request Timeout 409 – Conflict 410 – Gone 411 – Length Required 412 – Precondition Failed 413 – Request Entity Too Large 414 – Request URI Too Long 415 – Unsupported Media Type 416 – Requested Range Not Satisfiable 417 – Expectation Failed 422 – Unprocessable Entity 423 – Locked 424 – Failed Dependency 425 – Unordered Collection 426 – Upgrade Required	Response status codes beginning with the digit "5" indicate cases in which the server is aware that it has erred or is incapable of performing the request.  500 – Internal Server Error 501 – Not Implemented 502 – Bad Gateway 503 – Service Unavailable 504 – Gateway Timeout 505 – HTTP Version Not Supported 506 – Variant Also Negotiates 507 – Insufficient Storage 510 – Not Extended
Examples of using HTTP Status Codes in REST				
201 – When doing a POST to create a new resource it is best to return 201 and not 200. 204 – When deleting a resources it is best to return 204, which indicates it succeeded but there is no body to return. 301 – If a 301 is returned the client should update any cached URI's to point to the new URI. 302 – This is often used for temporary redirect's, however 303 and 307 are better choices. 409 – This provides a great way to deal with conflicts caused by multiple updates. 501 – This implies that the feature will be implemented in the future.				
Special Cases				
306 – This status code is no longer used. It used to be for switch proxy. 418 – This status code from RFC 2324. However RFC 2324 was submitted as an April Fools' Joke. The message is <i>I am a teapot.</i>				
Key Description				
Black	HTTP version 1.0			
Blue	HTTP version 1.1			
Aqua	Extension RFC 2295			
Green	Extension RFC 2518			
Yellow	Extension RFC 2774			
Orange	Extension RFC 2817			
Purple	Extension RFC 3648			
Red	Extension RFC 4918			

# Create test cases and assertions. As examples use Medicare by zip for SOAP

---

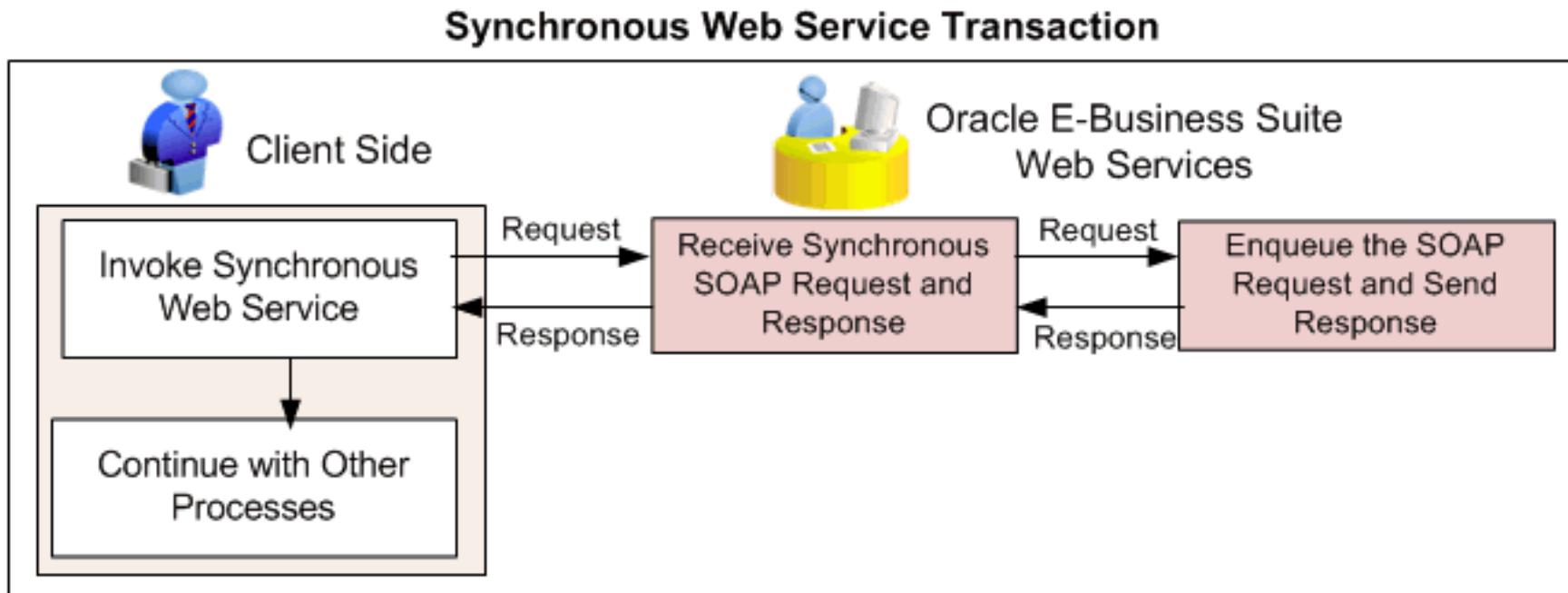


# Performance testing with SoapUI

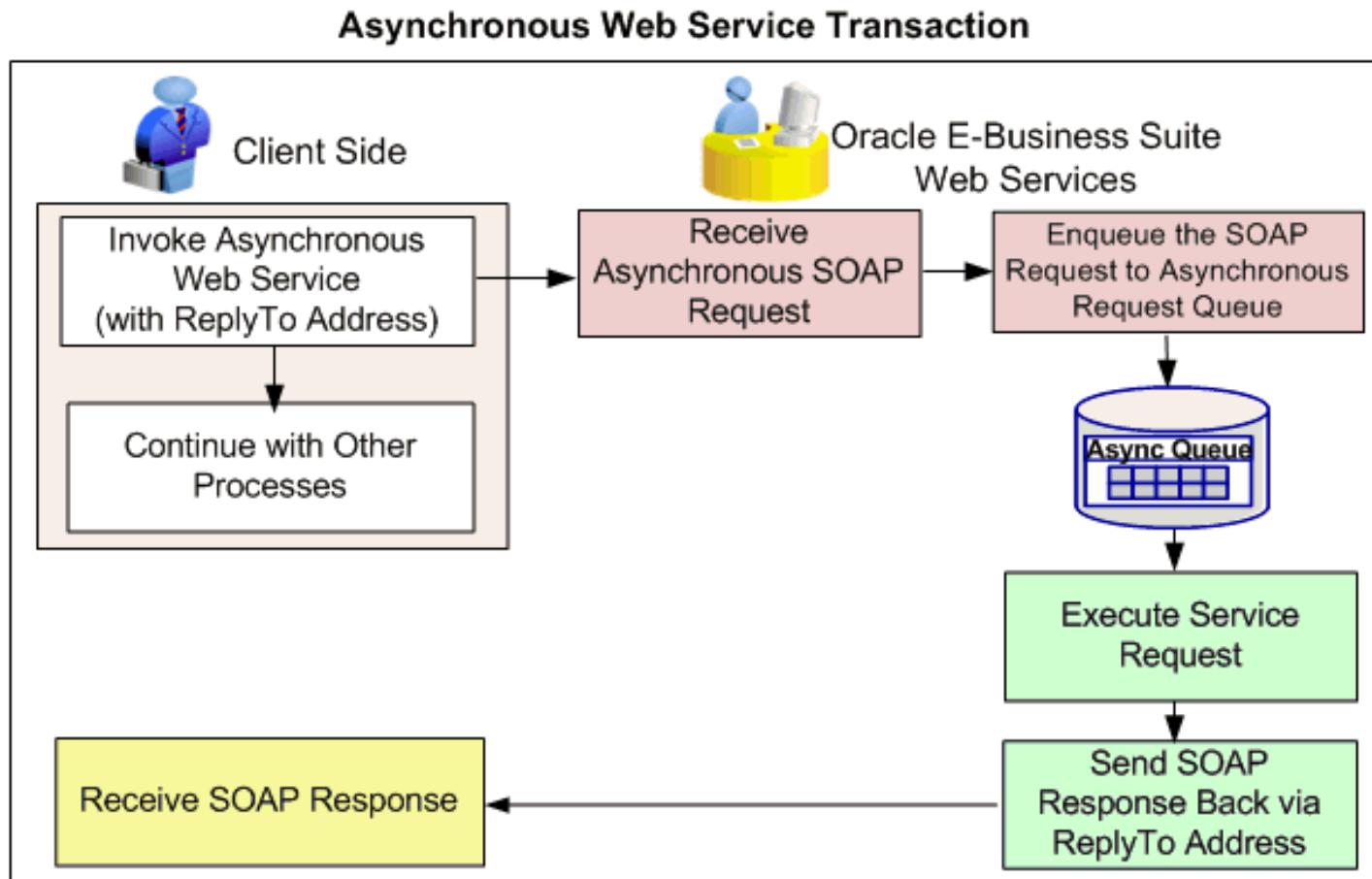
---

Only demonstration without testing (because we are using third parties web services)

# Synchronous web services



# Asynchronous web services



# Thank you

---

Questions

Suggestions