universität wien

Web Services

Faculty of Computer Science
Workflow Systems and Technologies

Interoperability SS 2023

Amolkirat Singh Mangat
Matthias Ehrendorfer
Florian Stertz

Distributed Systems

- Various software agents *work together to accomplish some tasks*.
- These agents *do not* necessarily *operate in the same computing environment* - communication must occur over the network.
- Architectural challenges of distributed systems include e.g.:
  - Lack of shared memory between caller and object
  - Concurrent access to remote resources
  - Latency and unreliability caused by underlying transport
  - Issues due to partial failures
  - Issues due to incompatible updates introduced to participants

Web Services

*"... are **self-contained**, **modular** business applications that have **open**, **internet-oriented**, **standards-based interfaces** ... communicate directly with other Web services via standards-based technologies"*
(UDDI Consortium)

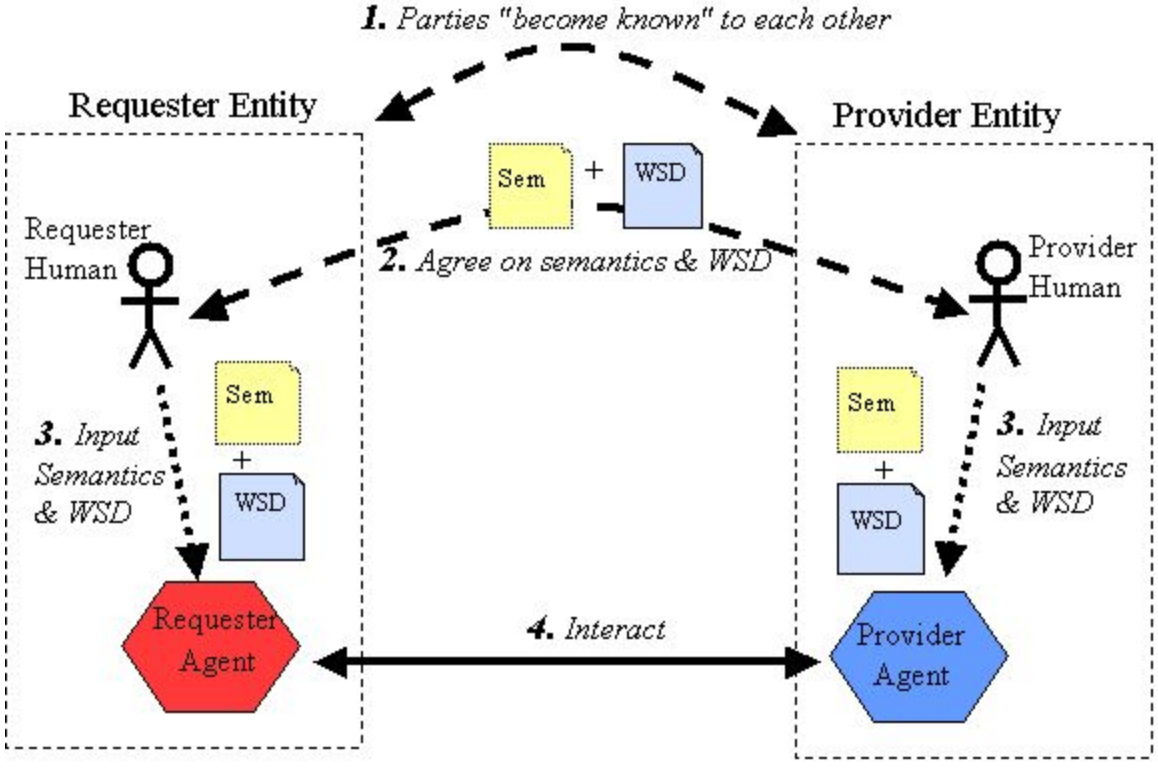
*"... is a software system designed to support **interoperable machine-to-machine interaction** over a network. It has an **interface** described **in a machine-processable format** ... Other **systems interact** with the Web service **in a manner prescribed by its description** ... "*
(W3C Consortium)

Web Service Architecture (W3C)

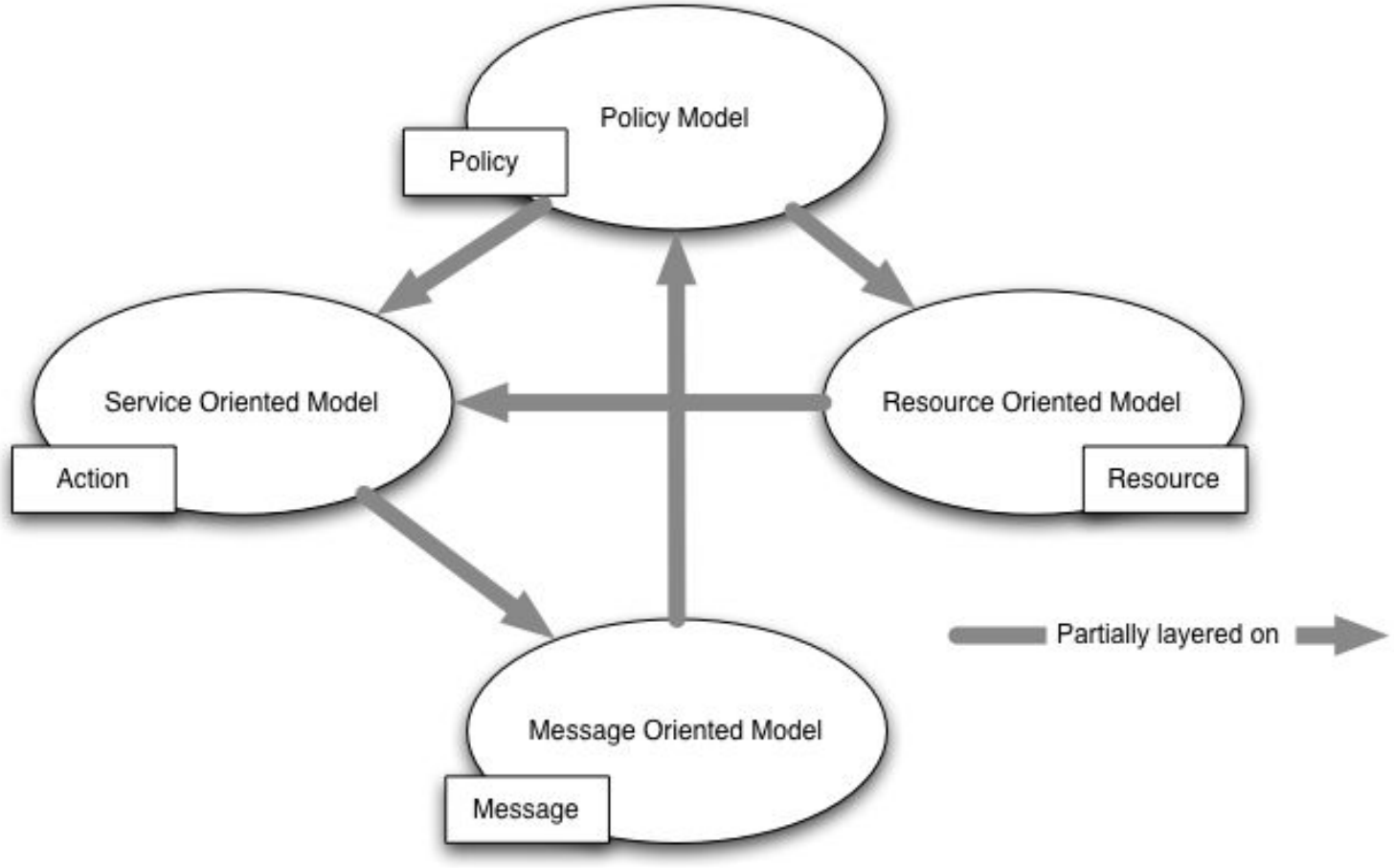
Collection of required *concepts* that enable interoperability between web services:

- ***Service*** Abstract definition of a web service - implementation agnostic
- **Agent** Concrete piece of software that implements a service which send and receives messages.
- **Provider** Person or organization that provides a service via an appropriate agent.
- **Requestor** Person or organization that consumes a provider's service
- **Service Description** Defines the mechanics of the message exchange between two parties.
- **Semantics** Shared expectation about the behavior of the service. It can be a formal or informal agreement - expresses a form of 'contract'.

Web Service Interaction

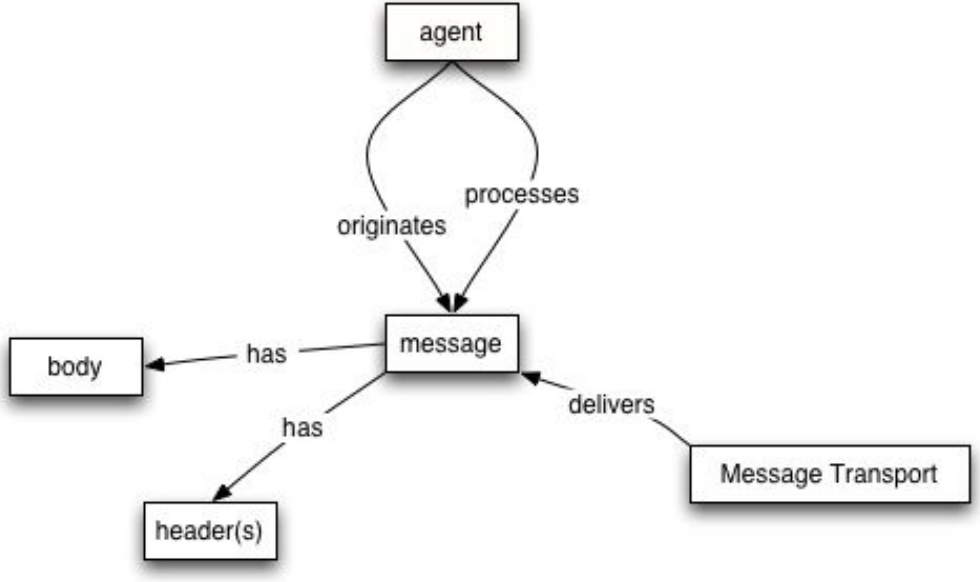


Architectural Models of Web Services



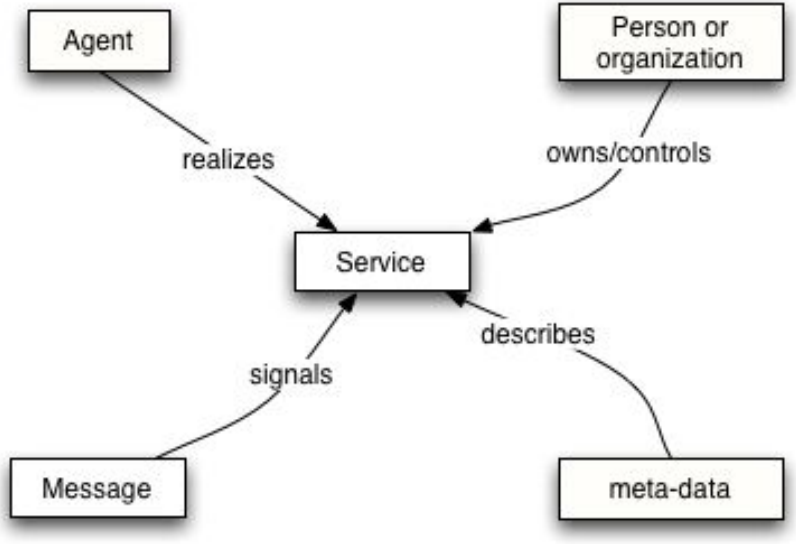
Architectural Models - Message Oriented Model

- Revolves around messages,
- the *structure of messages* with regards to the message headers and bodies,
- and around the *delivery mechanisms* for messages.



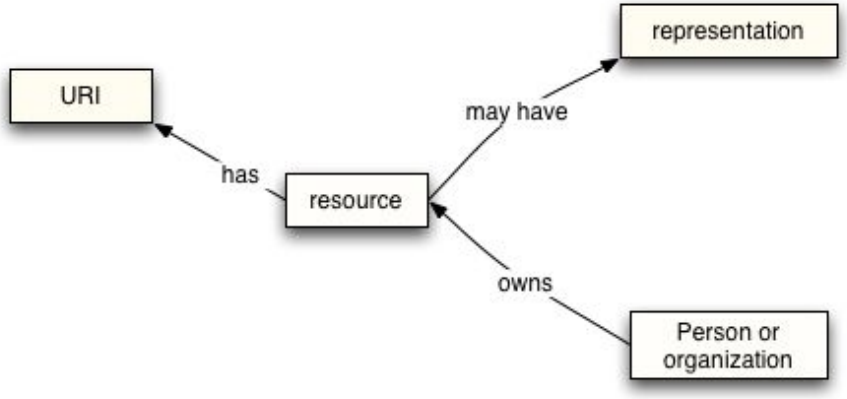
Architectural Models - Service Oriented Model

- Revolves around the concept of services.
- *Meta-data* is an integral part to *document* several aspects of services including interface and transport binding details, semantics and policy restrictions; key for the deployment and use of services.
- *Ownership* expresses the notion of *responsibility* for the provided functionality.



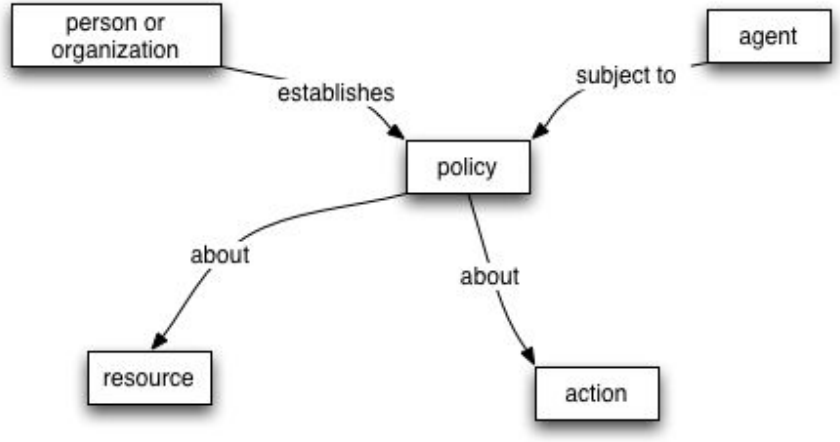
Architectural Models - Resource Oriented Model

- Revolves around the notion of a *resource* similar to the resource concept in the Web Architecture, i.e., *each URI identifies one resource* (e.g., web pages, images, multimedia-files etc.).
- Representations *reflect the state* of resources, however a representation mustn't be the same as the resource.



Architectural Models - Policy Model

- *Policies* are concerned with resources and are enacted to represent security, quality of service management and application concerns.
- Emphasis on *constraints* on the *behaviour of agents* (and services).
- Constraints are imposed on agents by the entity responsible for the resource.



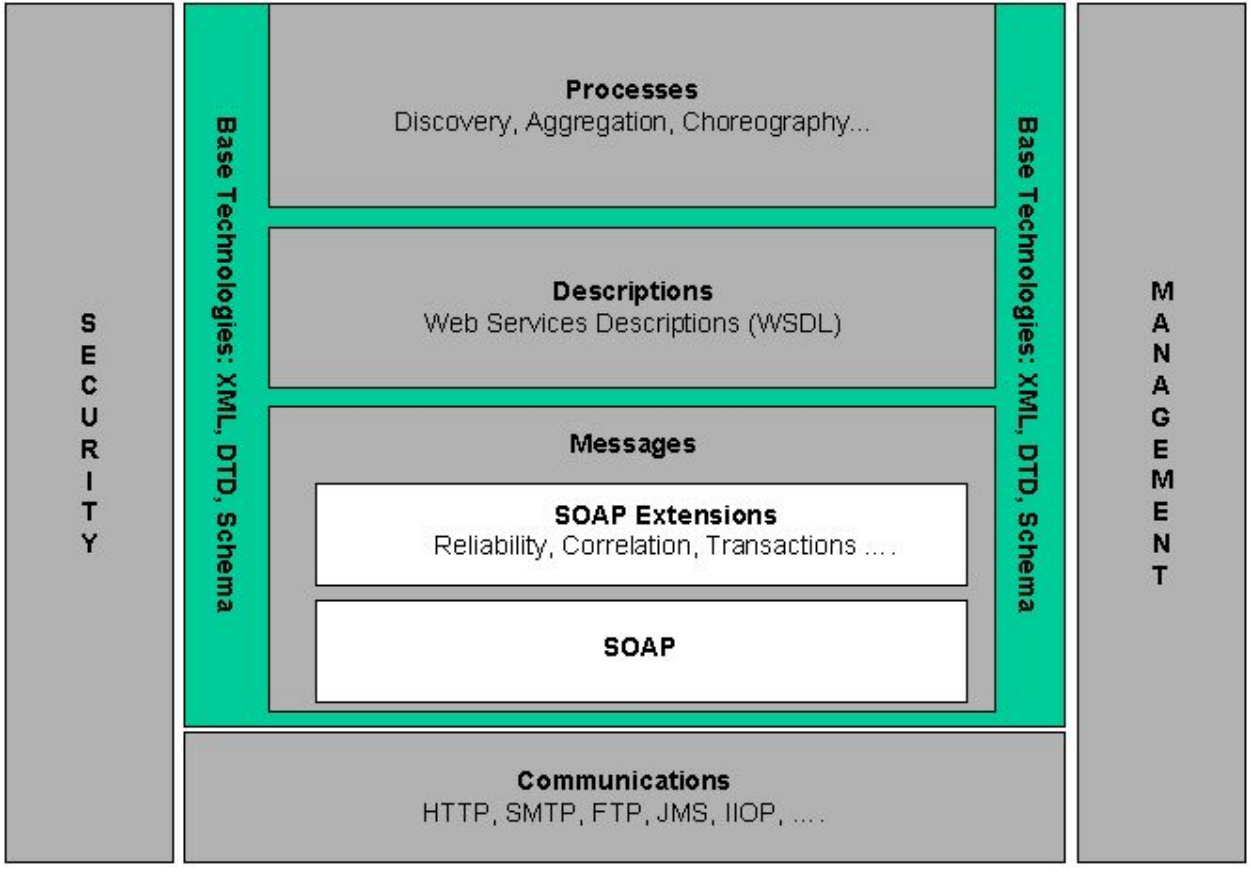
Service Oriented Architecture (SOA)

SOA is one form of distributed systems architecture that is characterized by following properties:

- **Logical View** Services are defined by *what they do*.
- **Message orientation** Services are defined by the messages exchanged - implementation details of the agents must be abstracted away.
- **Description orientation** Description of services by machine-processable meta-data. Semantics should also be included in the description.
- **Granularity** Tendency towards coarse-grained services and thus larger and complex messages.
- **Network orientation** Use of services foremost occurs over the network - is not an absolute requirement.
- **Platform neutrality** Messages are exchanged in platform-neutral and standardised format via interfaces.

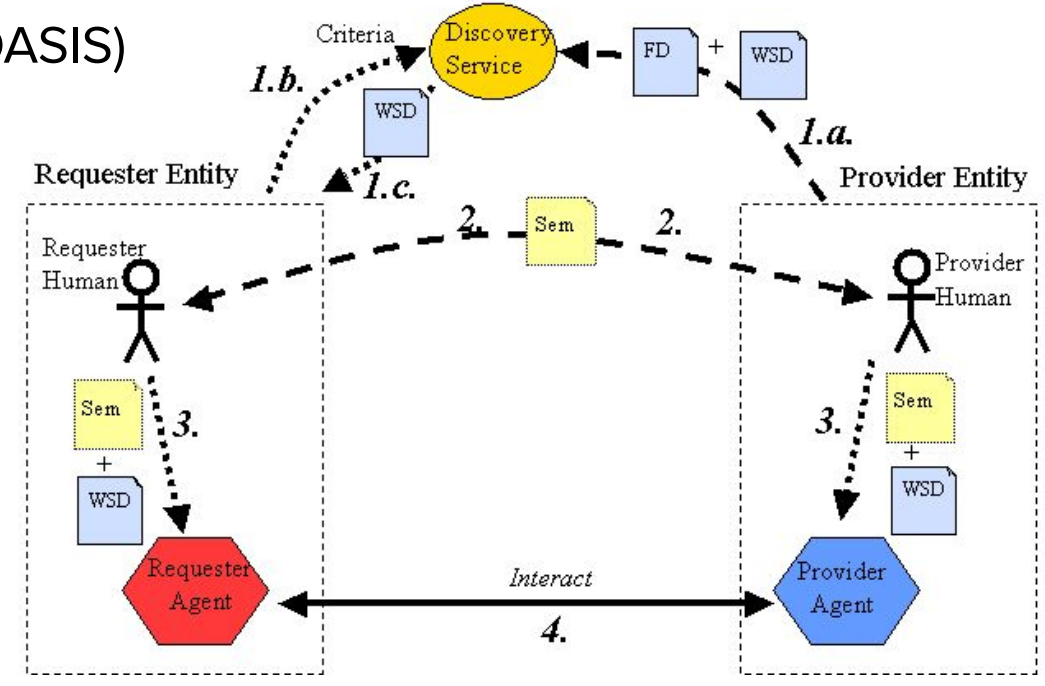
Web Service Perspectives

(Classic) Web Service Architecture Stack



Web Service Discovery

- Location of service (descriptions) one wants to engage with.
- Service location approaches
  - Registry (e.g. UDDI by OASIS)
  - Index
  - Peer-to-Peer



Web Service Semantics

- A successful interaction between systems demands a shared agreement about ***form***, ***structure*** *and* ***meaning*** of messages.
- This shared agreement governs the ***visibility*** of the *message semantics*.
- Use of *standards* facilitates acquiring insights about the *intent* of messages through the inspection of the flow of messages and their content.
    - E.g. SOAP defines the format and structure of the header and bodies of the messages.
    - Meaning expressed via meta-data (e.g. OWL, RDF)

Additional Perspectives

- Web Services Security aims at securing components involved in the point-to-point communication (e.g. transport encoding via SOAP, schema validation, message integrity checks, message confidentiality via encryption etc.)
- Web Services Reliability revolves around reliable and predictable delivery of messages and interactions of services.
- *Web Service Management* focuses on enabling monitoring, controlling, and reporting of service qualities and usage.

Web Services with SOAP and WSDL

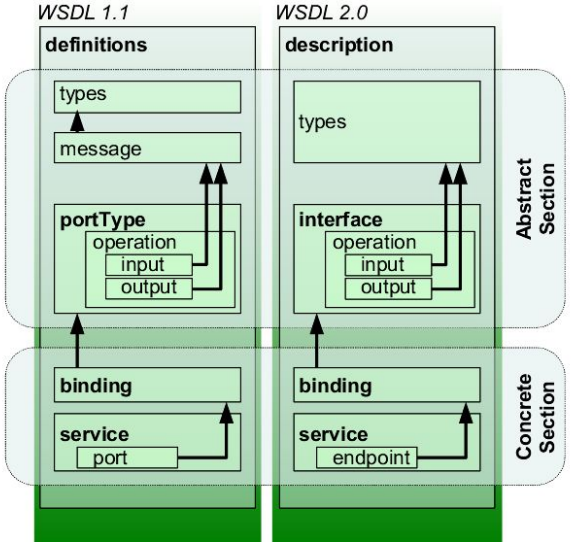
SOAP

- Represents a messaging framework to enable exchanging structured information among distributed systems [2].
- Version 1.1 and 1.2 are W3C standards.
- Enables remote invocation of services specified by a WSDL document.
- Uses XML and HTTP as the underlying technologies.
- Messages are wrapped in an Envelope consisting of an optional SOAP Header and a mandatory SOAP Body.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:ns1="urn:ec.europa.eu:taxud:vies:services:checkVat:types">
    <SOAP-ENV:Body>
        <ns1:checkVat>
            <ns1:countryCode>AT</ns1:countryCode>
            <ns1:vatNumber>U37586901</ns1:vatNumber>
        </ns1:checkVat>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Web Services Description Language (WSDL)

- Abstractly defines operations and messages of a web service [5].
- WSDL Version 1.1 and 2.0 are W3C standards
- Binding style determines how SOAP encodes the invoked operations - available options include RPC/literal, RPC/encoded and Document/literal (see [4]).



Building Blocks of WSDL [3]

Example: VIES Service with WSDL and SOAP

- VIES is service provided by the EU to verify the validity of a VAT number issued by any Member State.
- It provides a WSDL document for the service.

```
<?php //Run on almighty.cs.univie.ac.at via wwwlab.cs.univie.at/~a<student_id>/<vies>.php

$client = new SoapClient("https://ec.europa.eu/taxation_customs/vies/checkVatService.wsdl",
                         array('trace' => 1) ); // Enables us to view the last SOAP request/response

$res =$client->checkVat(
  array("countryCode" => "AT",
        "vatNumber" => "U37586901"));

header('content-type: text/plain');

echo "\n\n== Request SOAP Envelope\n";
print_r($client->__getLastRequest());

echo "\n\n== Response SOAP Envelope\n";
print_r($client->__getLastResponse());

echo "\n\n== Response (unmarshalled)\n";
print_r($res);
?>
```

Example: VIES Service with WSDL and SOAP

== Request SOAP Envelope

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:ns1="urn:ec.europa.eu:taxud:vies:services:checkVat:types">
    <SOAP-ENV:Body>
        <ns1:checkVat>
            <ns1:countryCode>AT</ns1:countryCode>
            <ns1:vatNumber>U37586901</ns1:vatNumber>
        </ns1:checkVat>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

== Response SOAP Envelope

```
<env:Envelope
    xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
    <env:Header/>
    <env:Body>
        <ns2:checkVatResponse
            xmlns:ns2="urn:ec.europa.eu:taxud:vies:services:checkVat:types">
            <ns2:countryCode>AT</ns2:countryCode>
            <ns2:vatNumber>U37586901</ns2:vatNumber>
            <ns2:requestDate>2023-05-04+02:00</ns2:requestDate>
            <ns2:valid>true</ns2:valid>
            <ns2:name>Universität Wien</ns2:name>
            <ns2:address>Universitätsring 1
                         AT-1010 Wien
            </ns2:address>
        </ns2:checkVatResponse>
    </env:Body>
</env:Envelope>
```

References

[1] https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/

[2] https://www.w3.org/TR/soap12/

[3] https://upload.wikimedia.org/wikipedia/commons/c/c2/WSDL_11vs20.png

[4] https://developer.ibm.com/articles/ws-whichwsdl/

[5] https://www.w3.org/TR/wsdl.html