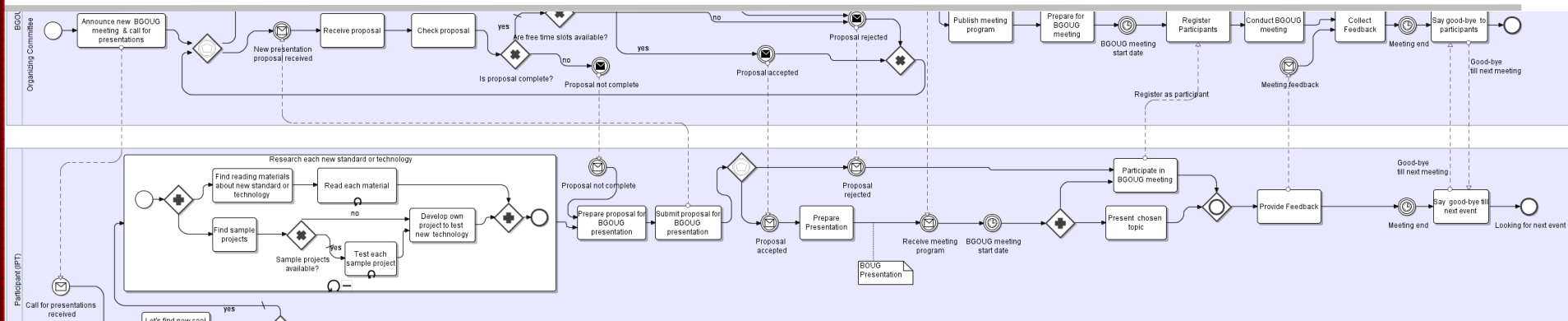


## XML. Уеб услуги и SOA. Simple Object Access Protocol (SOAP). Web Services Description Language (WSDL)



Траян Илиев

IPT – Intellectual Products & Technologies  
e-mail: [tiliev@iproduct.org](mailto:tiliev@iproduct.org)  
web: <http://www.iproduct.org>

Oracle®, Java™ and EJB™ are trademarks or registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. Oracle®, Java™ и EJB™ са търговски марки на Oracle и/или неговите подразделения. Всички други търговски марки са собственост на техните притежатели.

## Съдържание

1. Extensible Markup Language (XML)
2. XML Namespaces
3. XML схема дефиниции (XSD)
4. XML технологии и приложения
5. SOA и основни стандарти за уеб услуги
6. Simple Object Access Protocol (SOAP)
7. Web Services Description Language (WSDL)
8. Оптимизация на SOAP преноса – XML-binary Optimized Packaging (XOP), Message Transmission Optimization Mechanism (MTOM)

## Extensible Markup Language (XML)

- Официален стандарт – **XML 1.0** и XML 1.1 на World Wide Web Consortium - W3C
- Двоични и текстови файлове, метаданни
- Standard Generalized Markup Language – **SGML**
- HyperText Markup Language – **HTML**
- Extensible Markup Language – **XML**
- Extensible HyperText Markup Language – **XHTML**
- XML – стандарт за създаване на маркиращи езици за конкретни области на приложение
- Какво значи разширяем - “Extensible”?
- XML парсъри

## Сравнение между HTML и XML

### HTML:

```
<html>  
  <head>  
    <title>Page of John  
    Smith</title>  
  </head>  
  <body>  
    <p>John Smith</p>  
  </body>  
</html>
```

### XML:

```
<?xml version="1.0"  
  encoding="UTF-8"?>  
  <name>  
    <first>John</first>  
    <last>Smith</last>  
  </name>
```

## Информационни йерархии – обектни модели

- Сравнение между XML и HTML – различно предназначение:
  - HTML – специфично приложение: форматиране и визуализиране на информацията
  - XML – няма специфично приложение – произволни информационни структури
- HTML Document Object Model - DOM
- XML Document Object Model – DOM

## Extensible Markup Language (XML) – елементи

- Тагове, елементи и атрибути
- Дърво на документа – корен, клони, листа, типове възли
- Елементно съдържание – просто, смесено
- Документен тип – речник. XML валидация:
  - Document Type Definition (DTD)
  - XML Schema Definition (XSD)
- Визуализация на XML в уеб браузър:
  - CSS
  - XSLT
- XML Namespaces
- CDATA секции



## Основни елементи на XML (1)

- XML маркър и съдържание – маркър частта се намира между символите **<** и **>** (тагове) или между **&** и **;** (entities), всичко останало е **съдържание**
- XML процесор (XML parser) и потребителско приложение – анализира и обработва маркър и изпраща структурирана информация на потребителското приложение
- Тагове: **<mytag>**, **</mytag>**, **<mytag />**
- XML елемент – логически елемент на дървото на документа започващ с отварящ таг и завършващ със затварящ, може да включва съдържание и/или вложени под-елементи:

**<mytag>**I am**<name>**George**</name>****</mytag>**

## Основни елементи на XML (2)

- Атрибут – представлява двойка име/стойност, която може да се включи в отварящ или празен таг:  
`<book id='15' isbn="817525766-0">`  
    Научи Java за един ден  
`</book>`
- XML декларация:  
`<?xml version="1.0" encoding="ISO-8859-1"?>`
- `encoding` атрибутът може да има различни стойности: Windows-1252, ISO-8859-1, UTF-8, UTF-16 и др.
- Добре структуриран XML – един коренен елемент и др.
- Може да бъде валидиран с DTD, XML Schema, RELAX NG...



## XML Namespaces

- Пространствата от имена (Namespaces) позволяват да използваме тагове с **едно и също име за различни цели** (подобно на пакетите в Java)
- Декларират се с помощта на атрибута **xmlns** – например:  
**xmlns="http://schemas.xmlsoap.org/wsdl/"** - пространство от имена по подразбиране за елемента и всички вложени в него под-елементи  
**xmlns:tns="http://iproduct.org/usermanager"** - пространство от имена, което се идентифицира с префикса **tns** в елемента и всички вложени в него под-елементи
- Валидацията на елементите в едно и също пространство от имена обикновено става с XML Schema дефиниция (XSD)

## XML Namespaces – пример (SOAP Request)

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  
  xmlns:q0="http://iproduct.org/usermanager"  
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  
  <soapenv:Body>  
    <q0:addUser>  
      <arg0>  
        <q0:id>4</q0:id>  
        <q0:username>trayan</q0:username>  
        <q0:password>changeit</q0:password>  
        <q0:name>Trayan Iliev</q0:name>  
        <q0:email>tiliev@iproduct.org</q0:email>  
      </arg0>  
    </q0:addUser>  
  </soapenv:Body>  
</soapenv:Envelope>
```

## XML Namespaces - валидация с XML Schema

```
<definitions name="UserManagerService"
targetNamespace="http://iproduct.org/usermanager"
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://iproduct.org/usermanager"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://iproduct.org/usermanager"
        schemaLocation="UserManagerService_schema1.xsd"/>
    </xsd:schema>
    <xsd:schema>
      <xsd:import namespace="http://iproduct.org/models"
        schemaLocation="UserManagerService_schema2.xsd"/>
    </xsd:schema>
  </types>
```

## Технологии свързани с XML (1)

- **XHTML**
- XML Document Object Model (**DOM**)
- Extensible Style Sheet Language (**XSL**)
- XSL Transformations (**XSLT**)
- XSL Formatting Objects (**XSL-FO**)
- **XPath** – за селектиране на елементи и атрибути в дървото на документа
- Document Type Definition (**DTD**)
- XML Schema (**XSD**)

## Технологии свързани с XML (2)

- XML Linking Language (**XLink**) и XML Pointer Language (**XPointer**) – за свързване на XML документи
- **XQuery** – за осъществяване на заявки (подобни на SQL) към XML данни, включително бази от данни
- **XForms** – ново поколение HTML форми за въвеждане на данни, независими от конкретната платформа и устройство, част от стандарта XHTML 2.0



## Приложения на XML

- Нов начин за създаване на модулни уеб сайтове с възможност за еволюция
- Семантична мрежа – RDF, OWL
- Уеб 2.0 – социални технологии, блог, RSS, mashup
- Многослойни уеб приложения
- Уеб услуги
- Електронен бизнес и търговия – B2B, B2C



## Езици базирани на XML

- Mathematical Markup Language (MathML)
- Resource Description Framework (RDF)
- Web Ontology Language (OWL)
- Really Simple Syndication (RSS)
- Synchronized Multimedia Integration Language (SMIL)
- Scalable Vector Graphics (SVG)
- Wireless Application Protocol (WAP)
- **Simple Object Access Protocol (SOAP)**
- **Web Services Description Language (WSDL)**

## Пример: MathML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1 plus MathML 2.0//EN"
"http://www.w3.org/Math/DTD/mathml2/xhtml-math11-f.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
  <head>
    <title>Example of MathML embedded in an XHTML file</title>
  </head>
  <body>
    <h1>Пример за използване на MathML в XHTML документ</h1>
    <p>Площта на кръга се изчислява по следната формула:
      <math xmlns="http://www.w3.org/1998/Math/MathML">
        <mi>&#x03C0;</mi> <!-- π -->
        <mo>&#x2062;</mo> <!-- Умножение -->
        <msup>
          <mi>r</mi>
          <mn>2</mn>
        </msup>
      </math>.
    </p>
  </body>
```

## XML схема дефиниции (XSD) (1)

- Обикновено схемата дефинира **“структурата и организацията на базата от данни”**, резултат от процес на моделиране на данните в определена приложна област
- Схемата дефинира типовете на данните (прости, съставни) и ограниченията за валидност при тяхното комбиниране и използване (integrity constraints)
- Предимства пред DTD – явява се валиден XML и може да се обработва (парсва, генерира, валидира) използвайки стандартни технологии, поддържа типизация и наследяване на типовете данни
- Могат да се комбинират различни схеми, всяко от които да валидира собствено пространство от имена

## XML схема дефиниция - пример

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:mod="http://iproduct.org/models" attributeFormDefault="unqualified"
elementFormDefault="qualified" targetNamespace="http://iproduct.org/models">
  <xs:element name="user">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="id" type="xs:long"/>
        <xs:element minOccurs="0" name="username" type="xs:string"/>
        <xs:element minOccurs="0" name="password" type="xs:string"/>
        <xs:element minOccurs="0" name="name" type="xs:string"/>
        <xs:element minOccurs="0" name="email" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

## XML схема дефиниции – типове данни (1)

- Елементни структури и типове данни
- Примитивни типове данни: `string`, `boolean`, `float` (32 bit), `double` (64 bit), `decimal` (nonlimited precision), `date`, `time`, `dateTime`, `duration`, `binary`, `uriReference`, `ID`, `IDREF`, `NMTOKEN`, `NMTOKENS`, `ENTITY`, `NOTATION`, `QNAME` и др.
- Производни типове данни – получават се от съществуващ тип чрез комбинация или рестрикция
- Атомарни (`IDREF`, `NMTOKEN`) и списъчни типове (`IDREFS`, `NMTOKENS` – списък от токени разделени с интервали – `derivedBy="list"`)
- По подразбиране `derivedBy="restriction"` - специализация



## XML схема дефиниции – типове данни (2)

- `<simpleType name="..." base="..." abstract="true/false" derivedBy="restriction/list">`  
    `<minLength value="4">` `<maxLength value="4">`  
    `<pattern value="[a-zA-Z][0-9][0-9][0-9]">`  
    `</simpleType>` – прост тип, който не може да съдържа вложени под-елементи или атрибути
- `<complexType>` – може да съдържа други под-елементи и/или атрибути
- `<simpleContent>` – може да съдържа extension/restriction на друг тип без под-елементи (може да има атрибути)
- `<complexContent>` – може да съдържа extension/restriction на друг тип с под-елементи или смесено съдържание



## XML схема дефиниции - примери

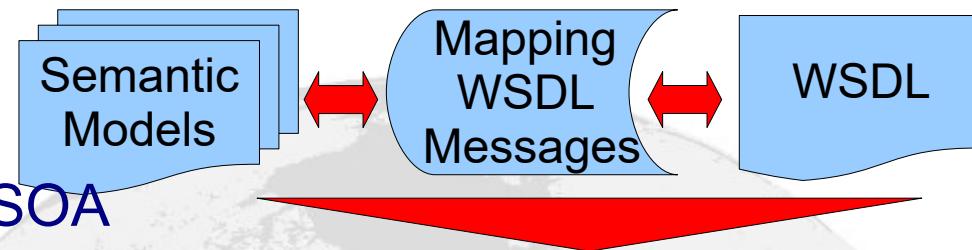
XSD примери във W3Schools:

<http://www.w3schools.com/schema/default.asp>

## SOA и Уеб услуги

Уеб услугите са:

- компоненти за изграждане разпределени приложения в SOA архитектурен стил
- комуникират използвайки отворени протоколи
- само-описателни и само-съдържащи се
- могат да бъдат откривани използвайки UDDI или ebXML регистри (и по-нови спецификации – WSIL & **Semantic Web Services**)

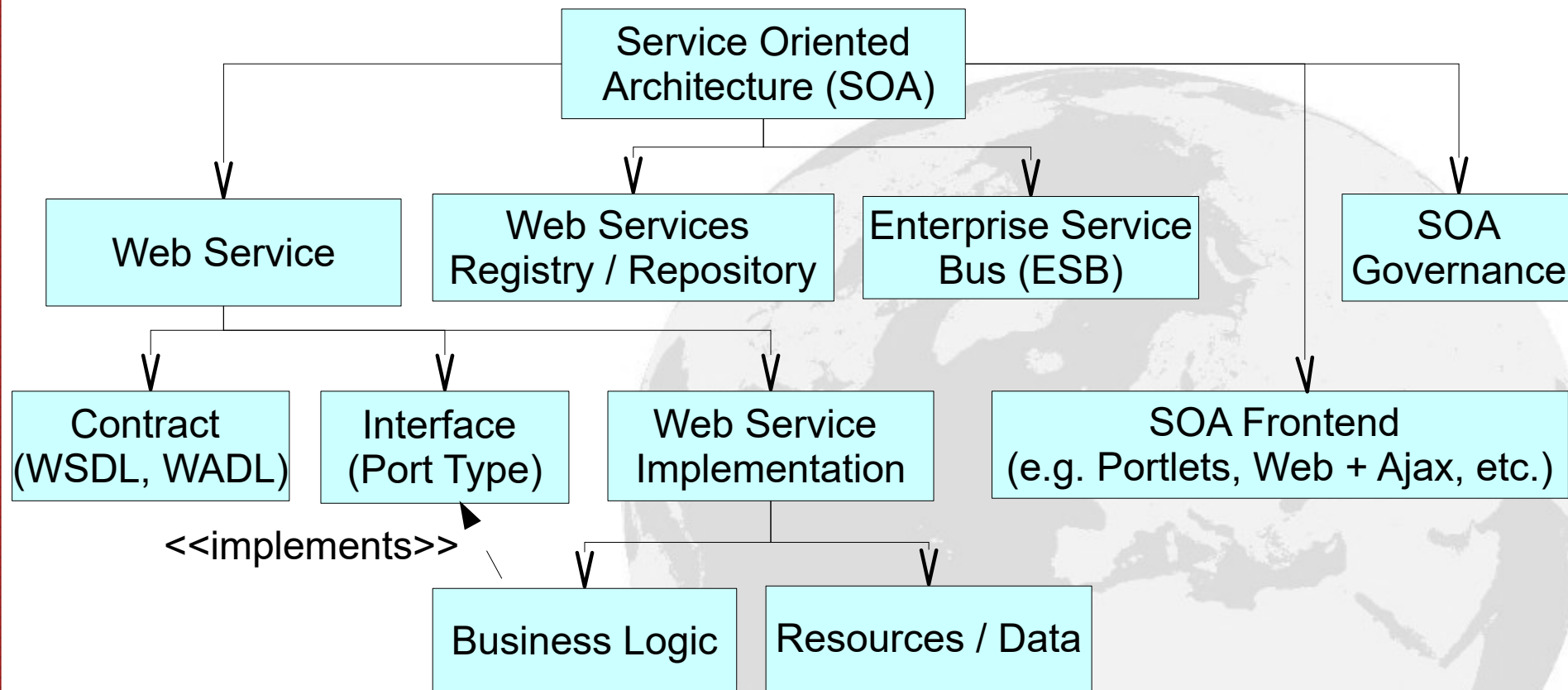


## SOA Main Principles (2)

According to <http://www.soapprinciples.com/> research site:

- Standardized Service Contract
  - Service Loose Coupling
  - Service Abstraction
  - Service Reusability
  - Service Autonomy
  - Service Granularity
  - Service Statelessness
  - Service Discoverability
  - Service Composability
- And more:
- Service Optimization
  - Service Relevance
  - Service Encapsulation
  - Service Location Transparency

## SOA – Main Concepts



By idea from: Dirk Krafzig, Karl Banke, and Dirk Slama. Enterprise SOA. Prentice Hall, 2005

## SOA Service Contract (1)

Според Wikipedia ([http://en.wikipedia.org/wiki/Service-oriented\\_architecture](http://en.wikipedia.org/wiki/Service-oriented_architecture)) програмният контракт на услугите включва:

- Name
- Version
- Owner
- Responsibility assignment (RACI):
  - Responsible
  - Accountable
  - Consulted
  - Informed
- Type:
  - Presentation
  - Process
  - Business
  - Data
  - Integration

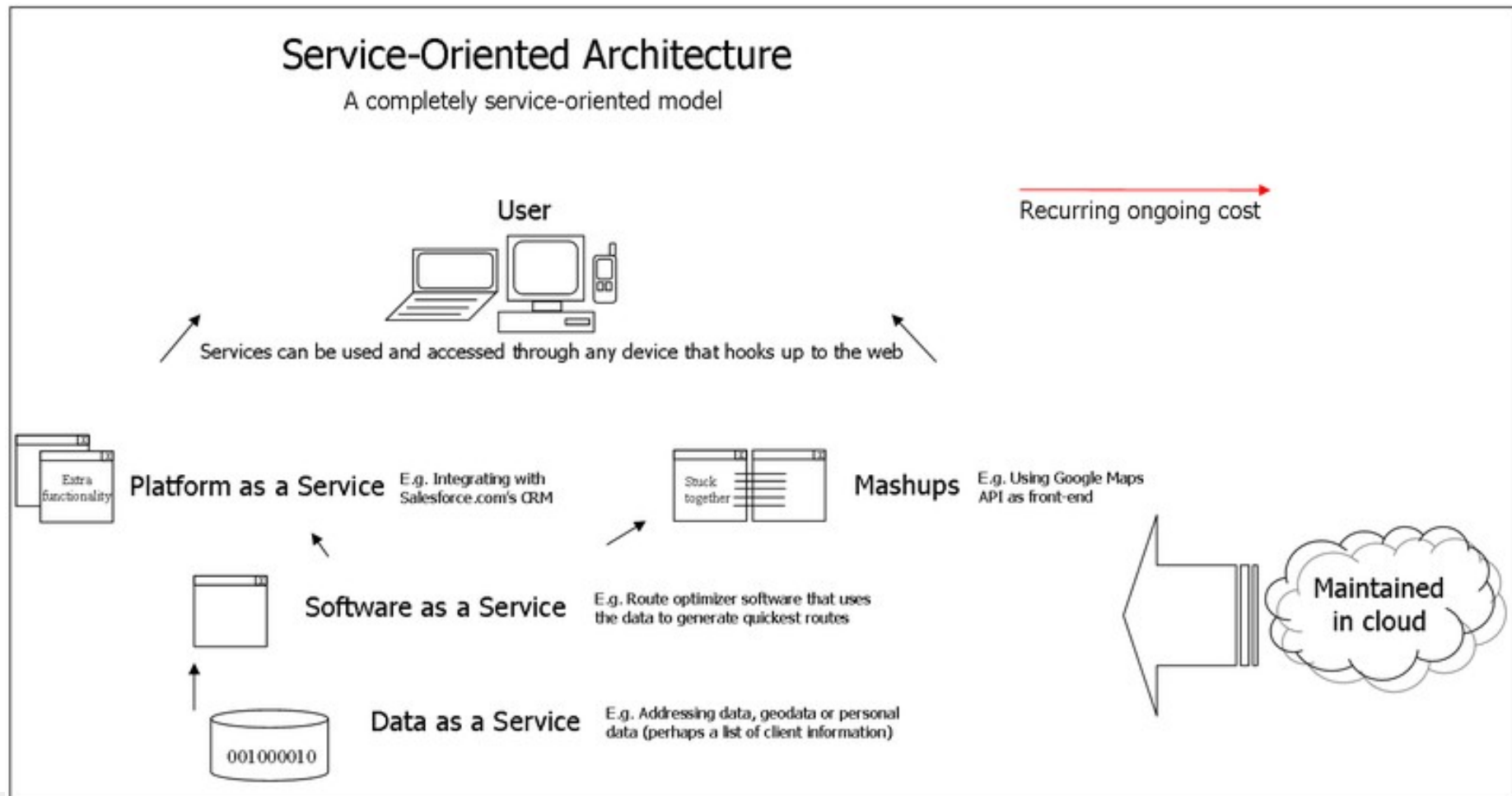
## SOA Service Contract (2)

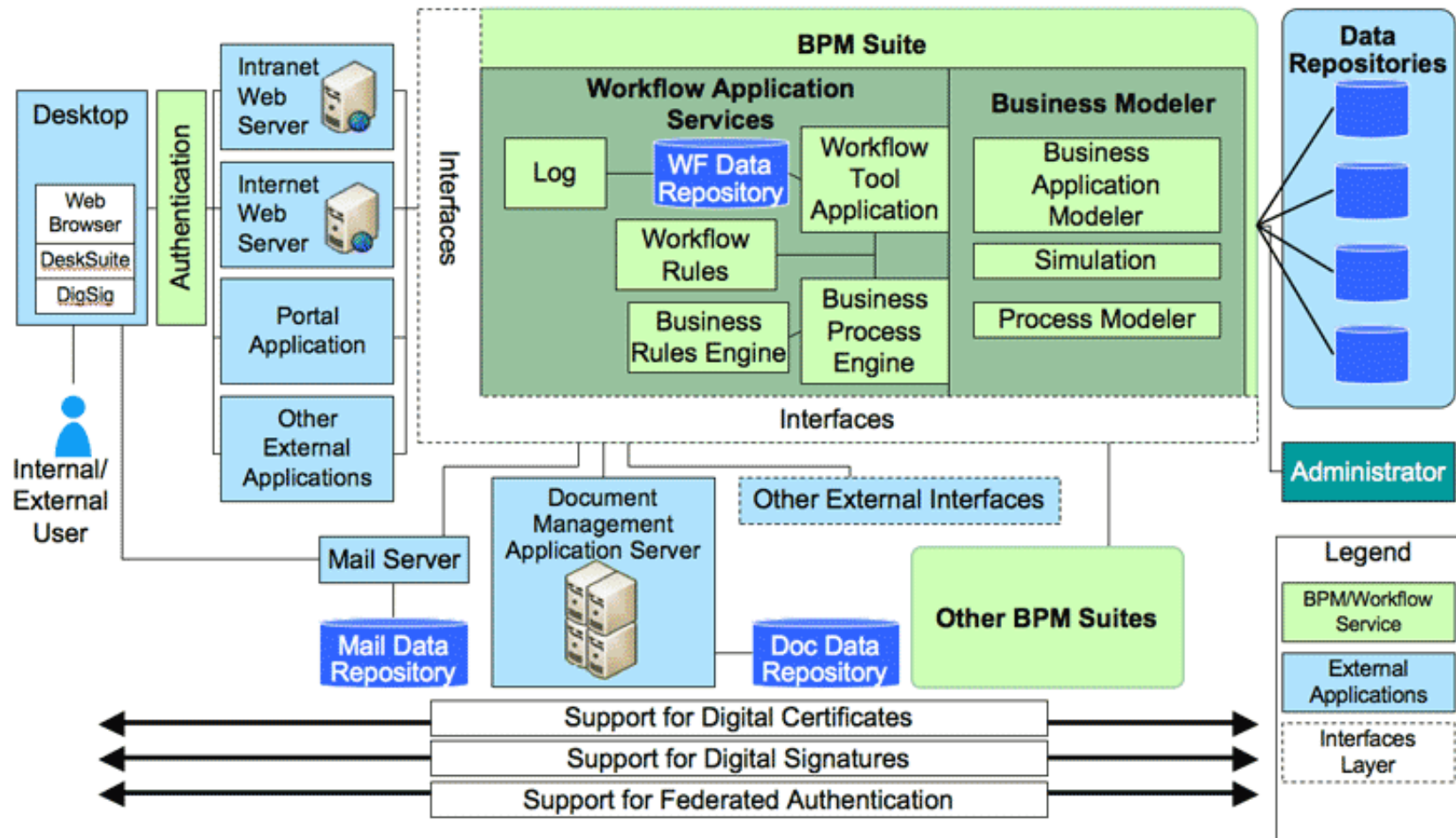
Според Wikipedia ([http://en.wikipedia.org/wiki/Service-oriented\\_architecture](http://en.wikipedia.org/wiki/Service-oriented_architecture)) програмният контракт на услугите включва:

- Functional requirement
- Service operations
- Invocation – e. g.:
  - SOAP
  - REST
  - Events triggers
- Security constraints
- Quality of service
- Transactional
- Service level agreement
- Semantics
- Process



# Service Oriented Architecture (SOA)





Sample business portal architecture that demonstrates how using Business Process Modeling (BPM) instruments can be built new executable business processes through orchestration & choreography of both human and software automated activities

**Source: National Institute of Health (2007). Business Process Management (BPM) Service Pattern –**

<http://enterprisearchitecture.nih.gov/ArchLib/AT/TA/WorkflowServicePattern.htm>

## SOA Manifesto (Октомври 2009)

- Business value over technical strategy.
- Strategic goals over project-specific benefits.
- Intrinsic interoperability over custom integration.
- Shared services over specific-purpose implementations.
- Flexibility over optimization.
- Evolutionary refinement over pursuit of initial perfection.

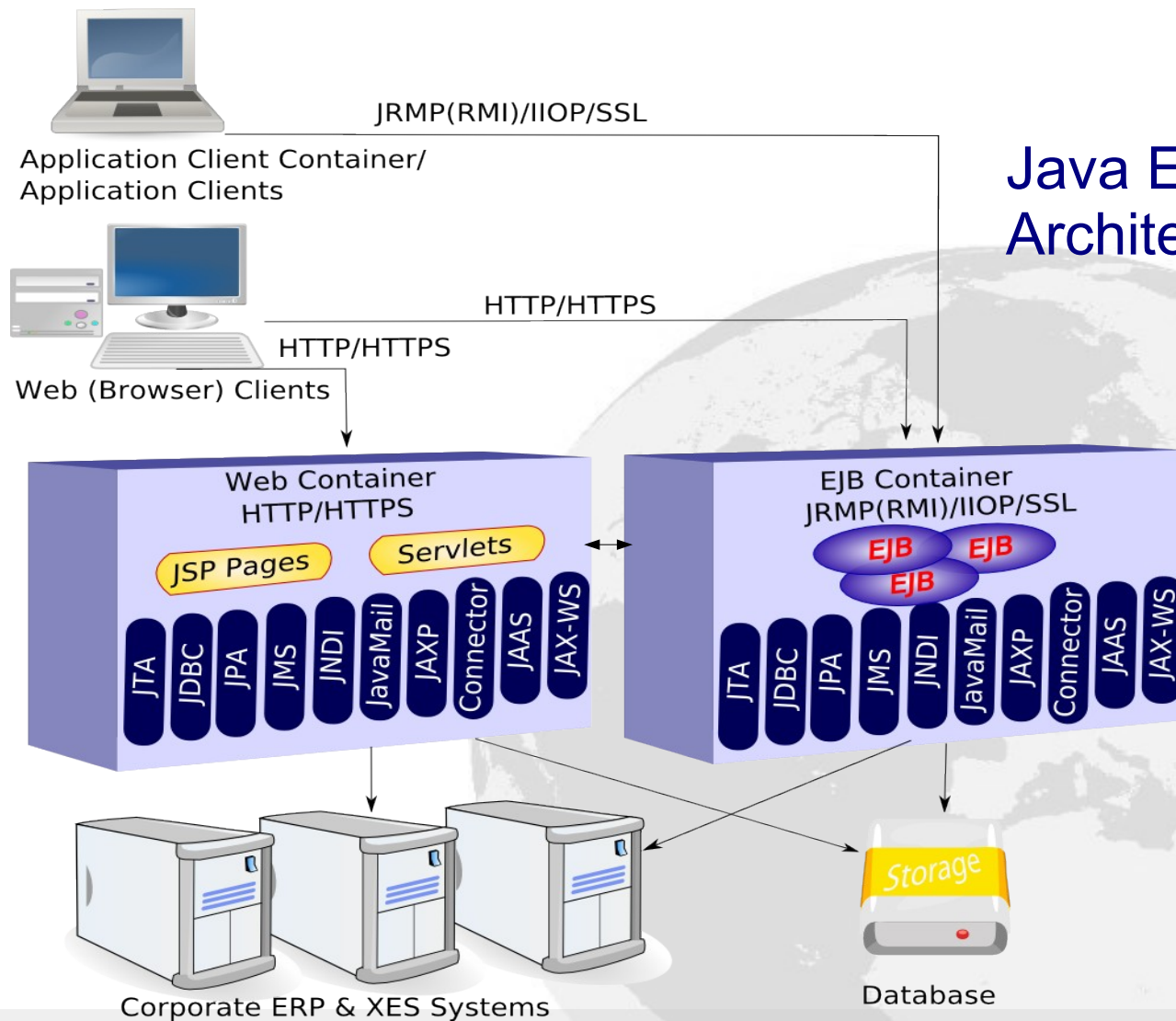
## SOA – базови стандарти и спецификации

- Reference Model for Service Oriented Architecture 1.0 – OASIS Standard, 12 October 2006
- OASIS Reference Architecture for Service Oriented Architecture Version 1.0 Public Review Draft 1, 23 April 2008
- OASIS Reference Architecture Foundation for Service Oriented Architecture Version 1.0 – Committee Draft 02, 14 October 2009
- SOA Practitioners Guide Parts 1 – 4 – достъпен на:  
<http://www.soablueprint.com/>

## SOA перспективи

- SOA & Web 2.0 Mashups
- Service-Oriented Business Applications (SOBA)
- Event-driven SOA (SOA 2.0) –  
[http://en.wikipedia.org/wiki/Event-driven\\_SOA](http://en.wikipedia.org/wiki/Event-driven_SOA)
- Internet of Services
- Digital Nervous System –  
[http://en.wikipedia.org/wiki/Digital\\_Nervous\\_System](http://en.wikipedia.org/wiki/Digital_Nervous_System)







## Java EE 6 / Java SE Standards APIs

Съгласно Java EE спецификацията:

- Web Services
  - ~~Java API for XML-based RPC (JAX-RPC)~~
  - Java API for XML Web Services (JAX-WS)
  - Java Architecture for XML Binding (JAXB)
  - SOAP with Attachments API for Java (SAAJ)
  - Java API for XML Registries (JAXR)
- RESTful Web Services
  - Jersey – RESTful Web Services - JAX-RS

## Поддръжка на веб услуги

- XML -базирани веб услуги:
  - Simple Object Access Protocol (SOAP)
    - XML-based envelope
    - XML-based encoding rules
    - XML-based request and response convention
  - Web Services Description Language (WSDL)
  - Universal Description, Discovery and Integration (UDDI) и ebXML Registries

## Simple Object Access Protocol (SOAP)

- Simple Object Access Protocol (SOAP) е W3C стандарт за XML-базирана, платформено и езиково независима комуникация communication между веб приложения и услуги, базирана върху размяна на съобщения
- SOAP е прост
- SOAP е разширяем

## SOAP - пример

- SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope
xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:add xmlns:ns2="http://calculator.me.org/">
      <i>5</i>
      <j>12</j>
    </ns2:add>
  </S:Body>
</S:Envelope>
```

## SOAP- пример

- SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope
xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:addResponse
xmlns:ns2="http://calculator.me.org/">
      <return>17</return>
    </ns2:addResponse>
  </S:Body>
</S:Envelope>
```

## Структура на SOAP съобщението

- SOAP Envelope
  - SOAP Header (optional)
  - SOAP Body
  - SOAP Fault
- ```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap=
"http://www.w3.org/2001/12/soap-
envelope"
soap:encodingStyle="http://
www.w3.org/2001/12/soap-encoding">
  <soap:Header> ... </soap:Header>
  <soap:Body> ...
    <soap:Fault> ...
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```



## SOAP Binding Styles

- **Remote procedure call (RPC):** структурата на `<soap:Body>` елемента в RPC стил е необходимо да съответства на правилата специфицирани в Section 7 на SOAP 1.1 спецификацията: `<soap:Body>` елемента може да съдържа **само един елемент** именуван с името на **операцията**, всички параметри трябва да бъдат представени като под-елементи на този “обвиващ” елемент. Типично се използва със SOAP encoding, който не съответства на изискванията на **WS-I** спецификацията.
- **Document Style:** `<soap:Body>` съдържанието се специфицира от XML Schema дефинирана в `<wsdl:type>` секция. То **не следва SOAP** конвенциите – SOAP съобщението в `<soap:Body>` се **изпраща като един документ** без специфично за SOAP форматиране. Документният стил се предпочита защото съответства на **WS-I** спецификацията.

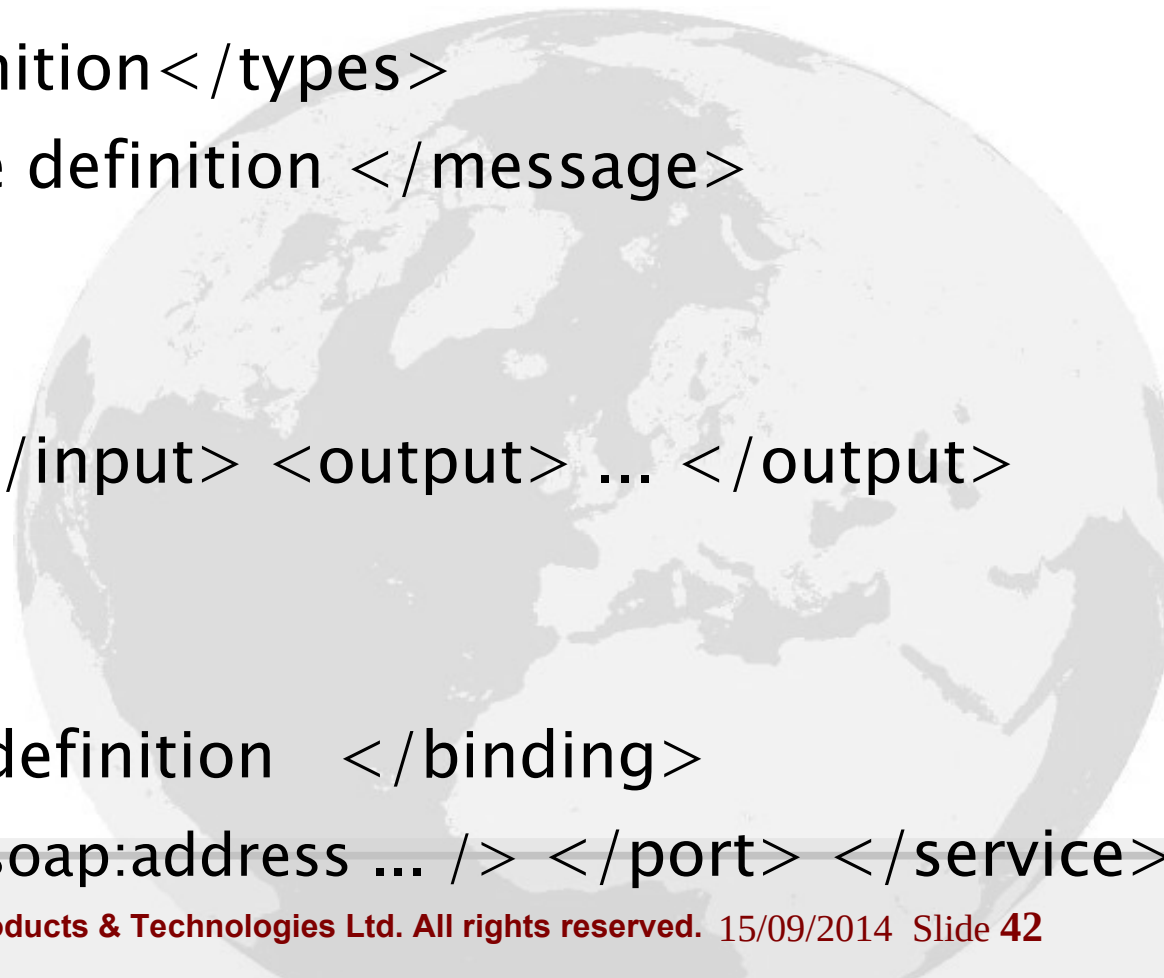
## SOAP Message Encodings

- **SOAP encoding (use="encoded")** – SOAP стила на кодиране позволява сериализация/ десериализация на стойностите на типовете данни от SOAP модела на данни съгласно стила на кодиране дефиниран в SOAP 1.1 спецификацията (рефериран от encodingStyle атрибута). Този стил не съответства на **WS-I** изискванията.
- **Literal XML encoding (use="literal")** – обозначава факта, че документът трябва да бъде прочетен такъв, какъвто е (as-is) или не-кодиран. Документът се сериализира като XML, който съответства на XML схемата представена в WSDL описанието (following the XML schema definitions literally). Всяко съобщение реферира конкретна схема дефиниция и съответства на **WS-I**.

## Web Services Description Language (WSDL)

- Web Services Description Language (WSDL) е W3C стандарт за XML-базирано описание на уеб услуги и също се използва за тяхното откриване и генериране на програмен код за достъп от клиента
- WSDL специфицира следната информация:
  - Името на уеб услугата и адресна информация
  - Комуникационния протокол върху който се пренася SOAP и стила на кодиране съобщения/операции на услугата
  - Информация за операции, параметри и типове на данните, представляващи интерфейса на уеб услугата

## WSDL 1.1 Basic Document Structure



```
<definitions>
  <types>    types definition</types>
  <message>  message definition </message>
  <portType>
    <operation>
      <input> ... </input> <output> ... </output>
    </operation>
  </portType>
  <binding>  binding definition  </binding>
  <service> <port> <soap:address ... /> </port> </service>
</definitions>
```

## Пример за WSDL описание на веб услуга (1)

```
<definitions
targetNamespace="http://endpoint.basicauth.helloservice/"
name="HelloService">
  <types>
    <xsd:schema>
      <xsd:import
namespace="http://endpoint.basicauth.helloservice/"
schemaLocation="http://localhost:8080/helloservice-basicauth/
HelloService?xsd=1"/>
    </xsd:schema>
  </types>
  <message name="sayHello">
    <part name="parameters" element="tns:sayHello"/>
  </message>
</definitions>
```



## Пример за WSDL описание на веб услуга (2)

```
</message>
<message name="sayHelloResponse">
  <part name="parameters" element="tns:sayHelloResponse"/>
</message>
<portType name="Hello">
  <operation name="sayHello">
    <input message="tns:sayHello"/>
    <output message="tns:sayHelloResponse"/>
  </operation>
</portType>
<binding name="HelloPortBinding" type="tns:Hello">
```



## Пример за WSDL описание на веб услуга (3)

```
<soap:binding
transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
<operation name="sayHello">
<soap:operation soapAction=""/>
<input>
<soap:body use="literal"/>
</input>
<output>
<soap:body use="literal"/>
</output>
</operation>
</binding>
```

## Пример за WSDL описание на веб услуга (4)

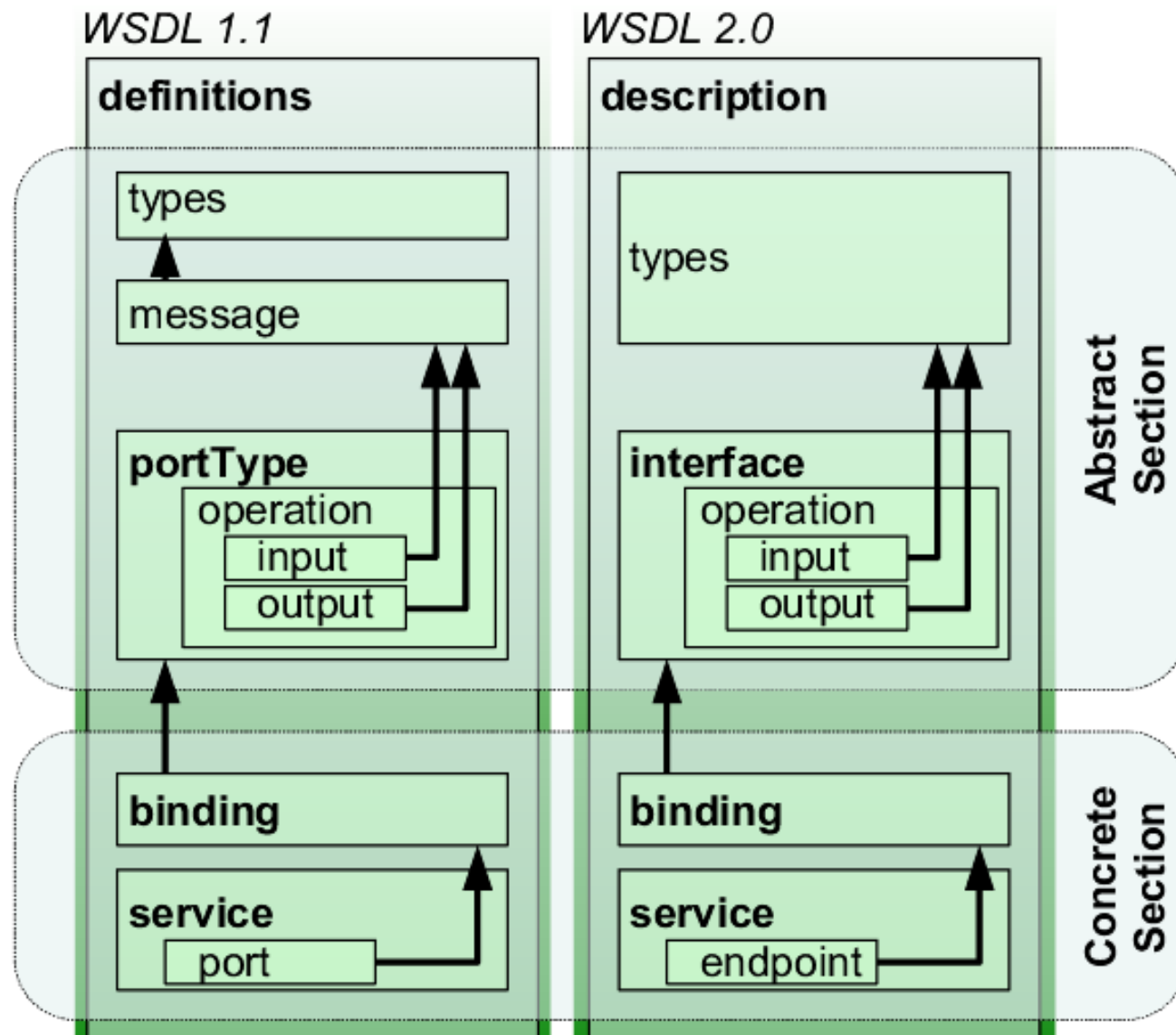
```
<service name="HelloService">  
  <port name="HelloPort" binding="tns:HelloPortBinding">  
    <soap:address location="http://localhost:8080/helloservice-  
      basicauth/HelloService"/>  
  </port>  
</service>  
</definitions>
```

## Феноменът Web 2.0

- Web 2.0 – Интернет като платформа за общуване, **създаване** и **споделяне** на съдържание – блогове, RSS/Atom, коментари, снимки, аудио, видео (**social media**)
- Колаборативно конструиране на колективно знание – **Participatory, Decentralized, Linked, Emergent**
- Нови ценности и нови възможности – мрежови ефект, постоянна еволюция, Taksonomy -> Folksonomy
- Отворени стандарти и свободен софтуер за създаване
- Service-Oriented Architectures (SOA)
- Разделяне на данни и презентация
- Богат, реагиращ потребителски интерфейс

# Representational State Transfer (REST)

- REpresentational State Transfer (REST) е една архитектура за достъп до разпределени, хипермедийни уеб услуги
- Ресурсите се идентифицират чрез URIs и се достъпват и манипулират използвайки основните методи на HTTP протокола (GET, POST, PUT, DELETE)
- Информацията се обменя с използване на представяния (representations) на тези ресурси
- По-олекотена алтернатива на SOAP+WSDL → HTTP



Разлики между  
WSDL 1.1  
и  
WSDL 2.0

## Web Application Description Language (WADL)

- XML-базиран файлов формат осигуряващ машинно-четимо описание за HTTP-базирани уеб приложения - типично REST web services.
- WADL е W3C Member Submission
  - Множество ресурси
  - Връзки между ресурсите
  - HTTP методи които могат да бъдат приложени за всеки ресурс
  - Очакван вход, изход и формати на данните
  - XML Schema типове данни за представяне на RESTful ресурсите



# Java API for XML Web Services (JAX-WS)

- Пакети: javax.jws, javax.jws.soap
- Основни анотации:
  - **@WebService** - маркира Java клас като имплементиращ веб услуга или Java интерфейс като дефиниращ интерфейс на веб услуга
  - **@WebMethod** – маркира методи които ще бъдат публикувани
  - **@WebParam** – мапинг на параметрите
  - **@WebResult** – мапинг на резултата
  - **@Oneway** – веб метод, който не връща резултат
  - **@SOAPBinding** – специфицира мапинг между Web Service --> SOAP message protocol

# Java API for RESTful Web Service (JAX-RS)

- Пакети: javax.ws.rs, javax.ws.rs.core, javax.ws.rs.ext
- Основни анотации:
  - @Path
  - @PathParam
  - @QueryParam
  - @FormParam
  - @HeaderParam
  - @CookieParam
  - @MatrixParam
  - @DefaultValue
  - @Encoded
  - @Context
  - @Produces
  - @Consumes,
  - @HttpMethod
  - @GET
  - @POST
  - @PUT
  - @DELETE

## Universal Description, Discovery, and Integration (UDDI)

- SOAP-базиран протокол, който дефинира комуникацията между клиентите на уеб услугата и UDDI Registry
- UDDI 2 предоставя възможност за 3 шаблона на запитвания:
  - Browse pattern
  - Drill-down pattern
  - Invocation pattern
- UDDI добре се допълва от семантични описания на метаданните и съответствията(mappings) с използване на RDF и OWL съхранени като tModels в UDDI Registry
- Web Services Inspection Language (WSIL) е разпределена алтернатива на UDDI фокусирана върху описанията на уеб услугите, а не на предоставящите ги бизнес организации

## Tools

- Admin Console
- asadmin
- asant
- appclient
- capture-schema
- package-appclient
- JavaDB
- verifier
- xjc
- schemagen
- **wsimport**
- **wsgen**

## Недостатъци от използването на XML за извършване на заявки към уеб услуги

- Неефективно представяне на двоични данни в XML формат (бавно, голям размер – Base64 => + 33%)
- Неефективен пренос през мрежата в текстов формат
- Бавно парсване на XML документите, ZIP-ването увеличава времето за обработка
- Необходимост от парсване на целия документ за достъп до определен елемент – невъзможност за пряк достъп
- Неефективна реализация на XPath търсене в големи документи
- Голямо количество памет е необходима за изграждане на DOM модела (3-5 пъти повече памет от размера на файла)



## Оптимизация на SOAP преноса

- XML-binary Optimized Packaging (XOP) е механизъм за сериализиране на XML Information Set (XML Infoset), които съдържат двоични данни (кодирани в Base64 encoding), както и за тяхното десериализиране обратно в XML Information Set
- Използва се в комбинация с Multipurpose Internet Mail Extensions (MIME) за дефиниране на механизъм за пакетизиране на двоичните данни в отделни пакети
- W3C Message Transmission Optimization Mechanism (MTOM) дефинира как може да се оптимизира SOAP протокола, така че ефикасно да пренася двоични данни от и към веб услуги с използване на XOP (XML-binary Optimized Packaging)



## Пример за XOP кодиране (Wikipedia)

```
MIME-Version: 1.0
Content-Type: Multipart/Related;boundary=MIME_boundary;
--MIME_boundary
Content-Type: application/xop+xml;
<soap:Envelope ...
  <soap:Body>...
    <m:photo xmlns:mime:contentType='image/png'>
      <xop:Include xmlns:xop='http://www.w3.org/2004/08/xop/include'
        href='cid:http://example.org/me.png'/></m:photo> ...
--MIME_boundary
Content-Type: image/png
Content-Transfer-Encoding: binary
Content-ID: <http://example.org/me.png>
// binary octets for png
```

## Binary XML

Терминът Binary XML се отнася до множество спецификации, които дефинират по-ефикасно представяне на **XML Information Set (XML Infoset)** в двоичен формат, като например:

- **Fastinfoset** – стандарт на **ITU-T Rec. X.891** и **ISO/IEC 24824-1 (Fast Infoset)** реализиран в **GlassFish** проекта на **SUN**, който сериализира XML в двоична форма пригодена за бързо парсване, с индексирание на низовете – около 10 пъти по-бърз от Apache Xerces, пригоден за мобилни устройства и големи обеми от данни през Интернет
- **W3C Efficient XML Interchange (EXI)** – използва XML Schema
- **Virtual Token Descriptor for XML (VTD-XML)** – реализира **non-extractive**, document-centric parsing чрез **Virtual Token Descriptors (VTD)** и **Location Caches (LC)** за бързо откриване

## Референции (1)

- J2EE 1.4 Tutorial –  
<http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html>
- Java EE 5 Tutorial –  
<http://java.sun.com/javaee/5/docs/tutorial/doc/>
- W3Schools – уроци за XML технологии:  
<http://w3schools.com/>
- SOAP Version 1.2 Part 0: Primer (Second Edition) W3C  
Recommendation - <http://www.w3.org/TR/soap12-part0/>
- Web Services Description Language (WSDL) Version 2.0 Part  
0: Primer -  
<http://www.w3.org/TR/2007/REC-wsdl20-primer-20070626/>

## Референции (2)

- XML-binary Optimized Packaging W3C Recommendation - <http://www.w3.org/TR/xop10/>
- SOAP Message Transmission Optimization Mechanism - <http://www.w3.org/TR/soap12-mtom/>
- Binary XML page във Wikipedia - [http://en.wikipedia.org/wiki/Binary\\_XML](http://en.wikipedia.org/wiki/Binary_XML)
- Техническо описание на Fast Infoset от Sun - <http://java.sun.com/developer/technicalArticles/xml/fastinfoset/>
- Efficient XML Interchange (EXI) Format 1.0 W3C Recommendation - <http://www.w3.org/TR/exi/>
- Virtual Token Descriptor for XML във Wikipedia - <http://en.wikipedia.org/wiki/VTD-XML>

Благодаря Ви за Вниманието!  
Въпроси?