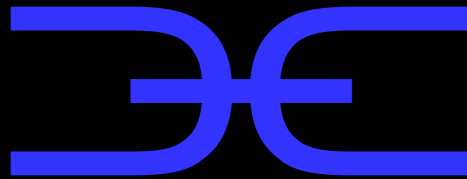


# Tehnologii Web

## servicii Web (I)



de la SOA la SOAP, WSDL și UDDI

“Prețuim ceea ce înțelegem.”

**Kevin Budelmann**

# Care sunt scopurile Web-ului?

# Constituirea și interacțiunea cu un spațiu de comunicare inter-umană

- ▶ partajarea cunoștințelor

# Constituirea și interacțiunea cu un spațiu de comunicare inter-umană

- ▶ partajarea cunoștințelor

Web social (“Web 2.0”), Web al datelor (semantic),...

# Exploatarea puterii computaționale

accesul la Web se poate realiza  
via dispozitive având resurse reduse

# Exploatarea puterii computaționale

accesul la Web se poate realiza  
via dispozitive având resurse reduse

**Web ubicuu** (omniprezent): **Web mobil, Web 3D, ...**  
performanță ► asigurarea scalabilității

# remarcă

Interacțiunea dintre om și Web se rezolvă  
prin intermediul formularelor Web și  
explorarea legăturilor via adrese Web – URI-uri



Cum pot fi accesate și procesate resursele  
– date, informații, cunoștințe –  
disponibile pe Web?

# nevoi ale dezvoltatorilor Web

Soluții multi-platformă, slab-conectate

integrare (în timp-real) la nivel de Internet/Web  
a aplicațiilor, serviciilor și sistemelor

# nevoi ale dezvoltatorilor Web

Soluții multi-platformă, slab-conectate

integrare (în timp-real) la nivel de Internet/Web  
a aplicațiilor, serviciilor și sistemelor

exemplificare: găsirea de resurse Web, pe baza localizării  
geografice a utilizatorului, privind ofertele de servicii  
multiple disponibile în contextul dispozitivelor mobile

# nevoi ale dezvoltatorilor Web

## Soluții multi-platformă, slab-conectate

datele să poată fi descrise pentru a fi „înțelese” de calculatoare și pentru a fi interconectate facil

# nevoi ale dezvoltatorilor Web

## Soluții multi-platformă, slab-conectate

datele să poată fi descrise pentru a fi „înțelese” de calculatoare și pentru a fi interconectate facil

*Web “puzzles”* ► inter-conectarea mai multor servicii informative (*e.g.*, situri de știri, *blog*-uri) conform preferințelor utilizatorului, pe baza intereselor sale

# nevoi ale dezvoltatorilor Web

Servicii ataşabile (*pluggable*) & versatile

*Software as a Service* – SaaS

*Application Service Provider* – ASP

# soluție

Divizarea aplicațiilor în **servicii** – **independente** –  
care se pot **compune**,  
menite a se **conecta** și **orchestra** în mod **spontan**  
în cadrul proceselor de afaceri/tehnice

# soluție

Divizarea aplicațiilor în **servicii** – **independente** –  
care se pot **compune**,  
menite a se **conecta** și **orchestra** în mod **spontan**  
în cadrul proceselor de afaceri/tehnice

*Web component-based software*



# soluție

*“The Web is the computer”*

# soluție

*“The Web is the computer”*

disponibilitatea unei/unor arhitecturi care...

oferă suport pentru paradigme de comunicare  
– bazată pe actualele tehnologii Web –  
între aplicații eterogene

# soluție

*“The Web is the computer”*

disponibilitatea unei/unor arhitecturi care...

permit(e) localizarea transparentă a serviciilor

# soluție

*“The Web is the computer”*

disponibilitatea unei/unor arhitecturi care...

facilitează adăugarea, înlocuirea, eliminarea  
serviciilor în mod dinamic

# soluție

*“The Web is the computer”*

disponibilitatea unei/unor arhitecturi care...

ascund(e) dezvoltatorului detaliile de sistem

# soluție

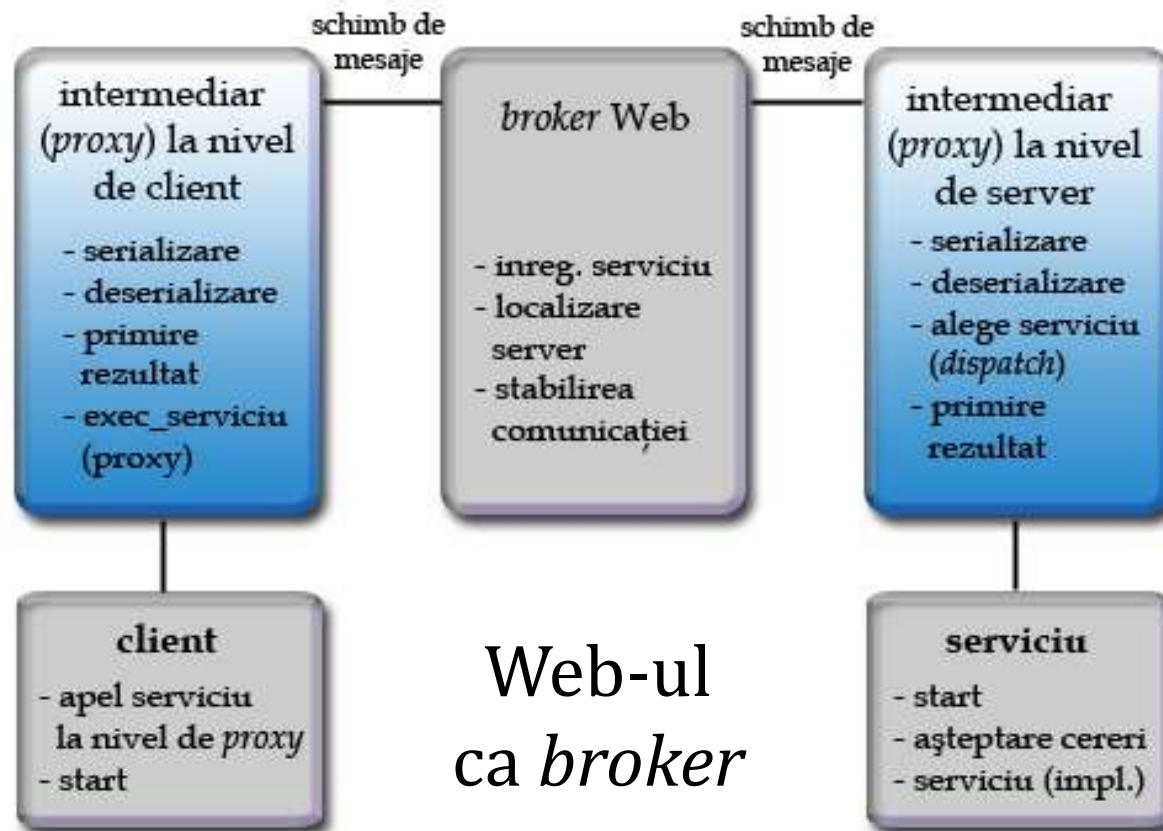
*“The Web is the computer”*

disponibilitatea unei/unor arhitecturi care...

asigură calitatea dezvoltării și exploataării  
aplicațiilor distribuite și/sau paralele:  
**standardizare, securitate, disponibilitate,  
reutilizare, mentenanță facilă etc.**

# soluție

## Arhitectura – Web-ul ca tehnologie *middleware*



# Ce sunt serviciile Web?



# servicii web

Software oferind o funcționalitate specifică

acces la resurse – Delicious, Pinterest, Slideshare, Vimeo

agregare de știri – Digg, Reddit

cartografiere – Bing Maps, Google Maps, Nokia HERE

mesagerie instantanee – Jabber, Twitter, Twilio

procesări – Anger Detection, Ping.it, Skyttle, Truthy,...

realizare de statistici Web – Google Analytics

rețele sociale – e.g., Facebook Open Graph Protocol

*spelling checking* – Spellr.us

stocare de date – Amazon S3, Dropbox, OneDrive etc.

...

pocket ▾

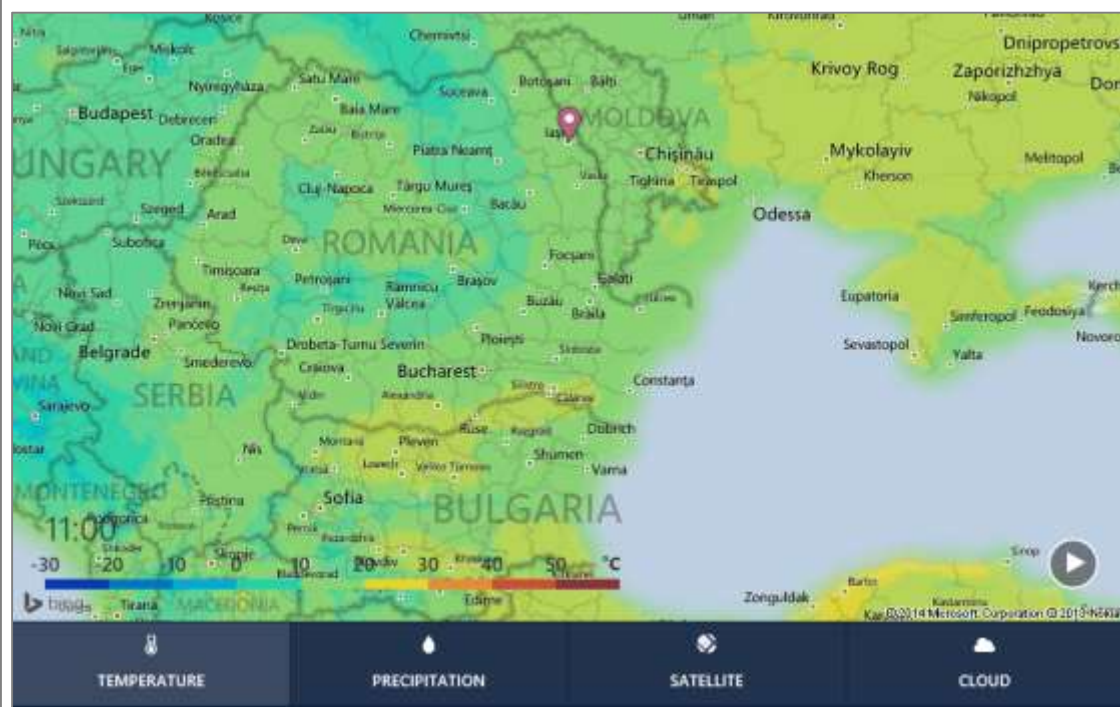
service

New Sync ▾

- Postman - A powerful HTTP client to test web services  
getpostman.com
- Migrating To A Service Oriented Architecture (SOA) - Belly Card Engineering  
tech.bellycard.com
- Composing Services into Layers | Stephen Haunts { Coding in the Trenches }  
stephenhaunts.com
- BBC - Research and Development: Developing the World Service archive protot  
bbc.co.uk
- Music Discovery Service sevl.fm Launches  
semanticweb.com

previous next

page: 1 ▾



utilizate – la distanță – de alte aplicații/servicii

# servicii web

Accesate standardizat via Web

adresare de resurse cu URI

transfer de date via HTTP

mesaje adoptând formate de date: CSV, JSON, XML,...

# servicii web: exemplu

Serviciul unei agenții de turism  
oferirea – și vânzarea, eventual –  
a unor formule de petrecere a vacanței

# servicii web: exemplu

Serviciul unei agenții de turism  
utilizează alte servicii (software)  
disponibile la nivel de Web

servicii cartografice + meteo  
servicii hoteliere

tranzacții financiare – *e.g., e-banking*

servicii de transport

servicii de recomandare socială

Cum am putea implementa un serviciu?

# servicii web

## Implementare standard

recurgerea la servere/*framework*-uri de aplicații Web



ASP.NET, Django, JSP, Node.js,  
PHP (CodeIgnater, Symfony,...),  
Ruby on Rails,...

# servicii web

Tradițional, aplicația oferă o interfață-utilizator disponibilă pe Web

limbaj de marcare – *e.g.*, HTML

stiluri de prezentare a conținutului – CSS

interactivitate via JavaScript (+biblioteci/*framework*-uri)



# servicii web

Tradițional, aplicația oferă o interfață-utilizator disponibilă pe Web

cererile sunt capt(ur)ate via formulare  
+ legături hipertext

# servicii web

Tradițional, aplicația oferă o interfață-utilizator disponibilă pe Web

utilizatorii umani trebuie să interpreteze etichetele și câmpurile de dialog

valoare	421.38
TVA (19%)	80.0622
total (cu TVA)	501.4422

# servicii web

Tradițional, aplicația oferă o interfață-utilizator disponibilă pe Web

serviciul implementat oferă un răspuns  
(o reprezentare a unei resurse Web)

uzual, un document HTML al cărui conținut e transferat la client via un protocol: HTTP(S)

# servicii web

Cum obținem răspunsul pentru a fi (re)folosit  
în programele noastre?

procesarea datelor din codul HTML ► **Web scrapping**

```
<tr><td>valoare</td><td><input name="val" value="0" readonly type="text"></td></tr>  
<tr><td>total (cu TVA)</td><td><input name="cutva" value="0" readonly type="text"></td></tr>
```

# servicii web

Cum obținem răspunsul pentru a fi (re)folosit  
în programele noastre?



orice modificare în marcaje ► rescrierea programului  
de preluare a datelor din documentul HTML

# servicii web: caracterizare

Serviciile Web fac explicite specificațiile implicite

datele de intrare și răspunsul pot fi specificate (riguros)  
via diverse scheme de validare

# servicii web: caracterizare

Utilizate la interacțiunea dintre aplicații

dinamice

lipsa unei cunoașteri *a-priori* a interacțiunii  
cu alte aplicații/servicii Web

# servicii web: caracterizare

Puncte finale utilizate pentru procesarea datelor, în manieră publică – eventual, via API-uri deschise



# servicii web: caracterizare

Dezvoltate pe baza platformelor, arhitecturilor,  
tehnologiilor și limbajelor curente

Există un model arhitectural de dezvoltare  
a serviciilor la nivel de Web?

**soa**

Arhitectura orientată spre servicii

**SOA – *Service Oriented Architecture***

# soa

Arhitectura orientată spre servicii

*SOA – Service Oriented Architecture*

stil arhitectural de proiectare și dezvoltare de aplicații  
considerate drept servicii  
care pot fi invocate de alte aplicații

# soa

Paradigmă de dezvoltare a software-ului  
care adoptă folosirea de servicii,  
oferind funcționalități solicitate de utilizatori

# soa

Paradigmă de dezvoltare a software-ului  
care adoptă folosirea de servicii,  
oferind funcționalități solicitate de utilizatori

resursele sunt disponibile via o **suită de servicii  
independente** ale căror **implementări**  
nu trebuie să fie cunoscute (*black box*)

# soa

Componentele sistemului în ansamblu  
au un grad mare de **independență** (*de-coupling*)

# soa

Componentele sistemului în ansamblu  
au un grad mare de **independență** (*de-coupling*)

serviciile pot fi recompuse/orchestrate  
conform cerințelor sau contextului de exploatare



# soa

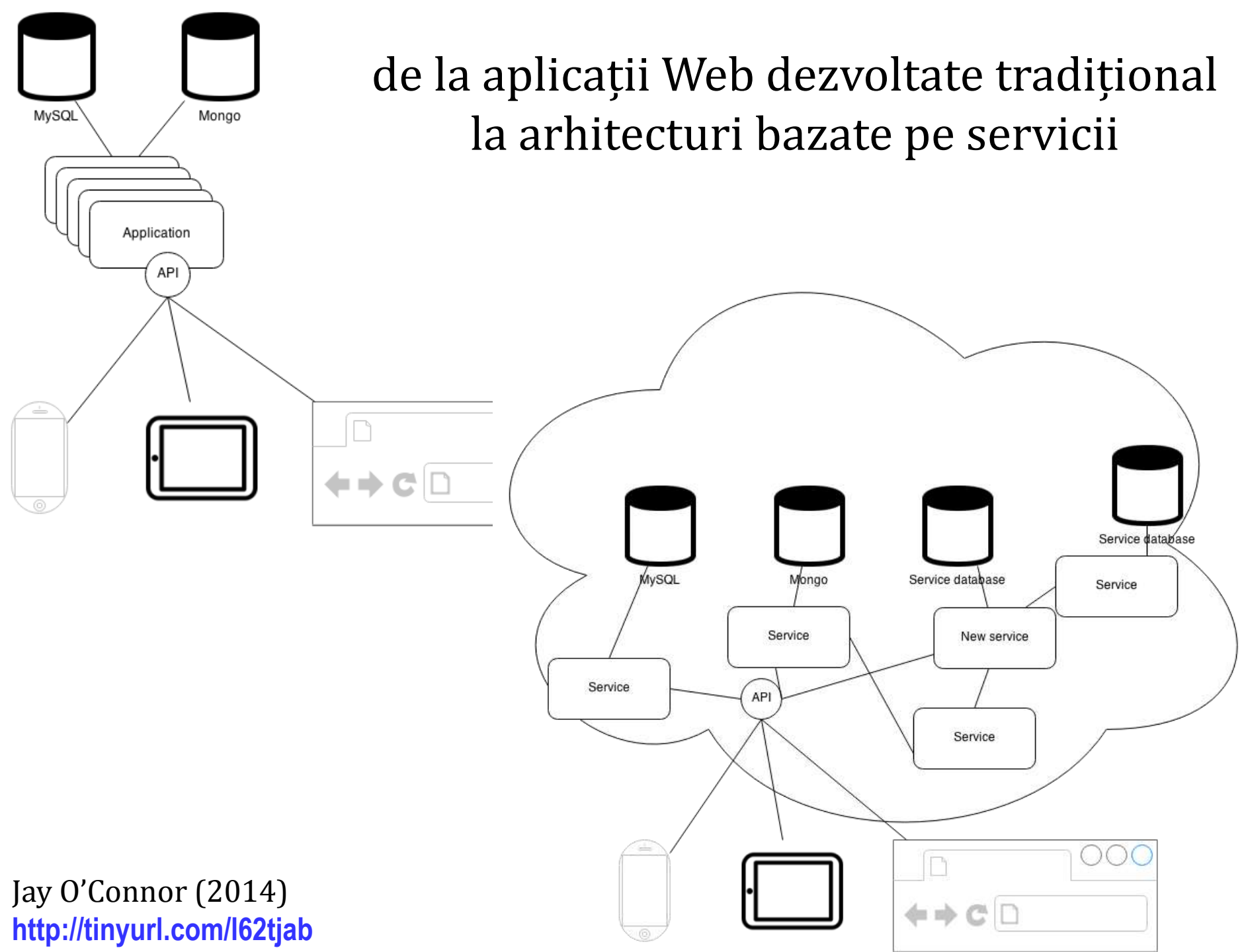
Serviciile nu vor depinde de starea comunicării  
(*statelessness*)

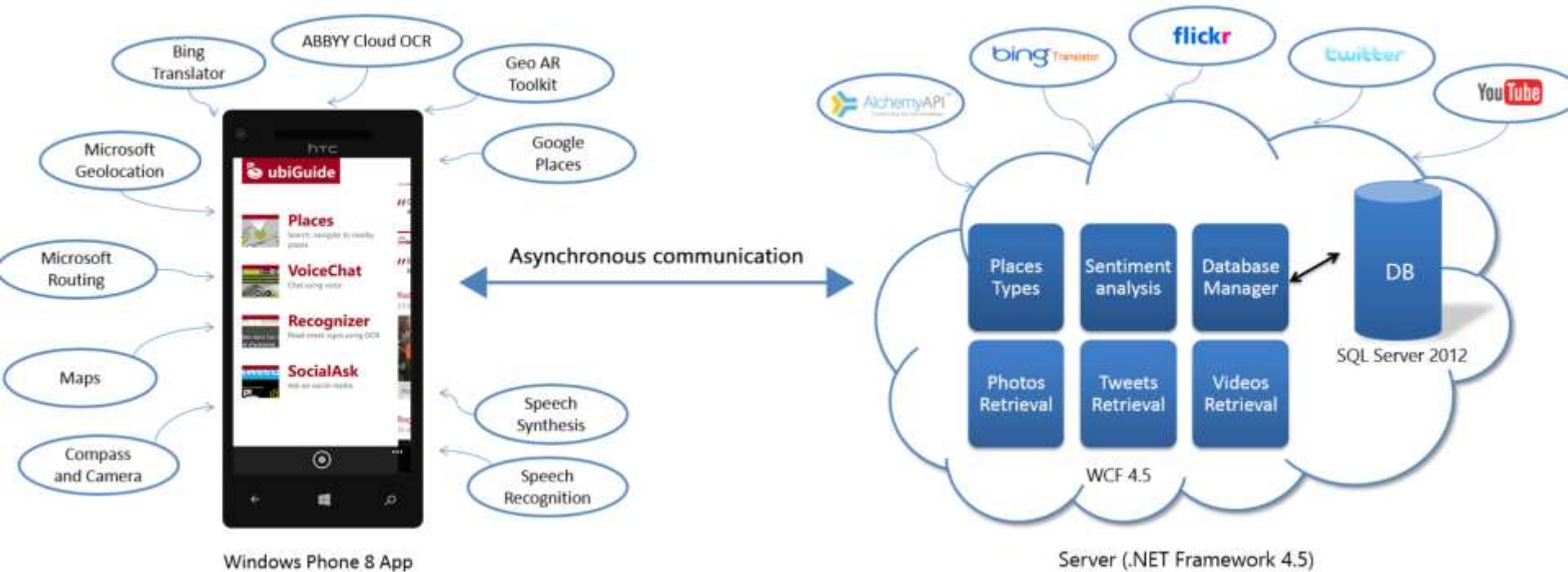
# soa

Serviciile nu vor depinde de starea comunicării  
(*statelessness*)

pentru a efectua o procesare, cantitatea de informație  
ce trebuie reținută trebuie să fie minimală

# de la aplicații Web dezvoltate tradițional la arhitecturi bazate pe servicii





## proiectul **ubiGuide**

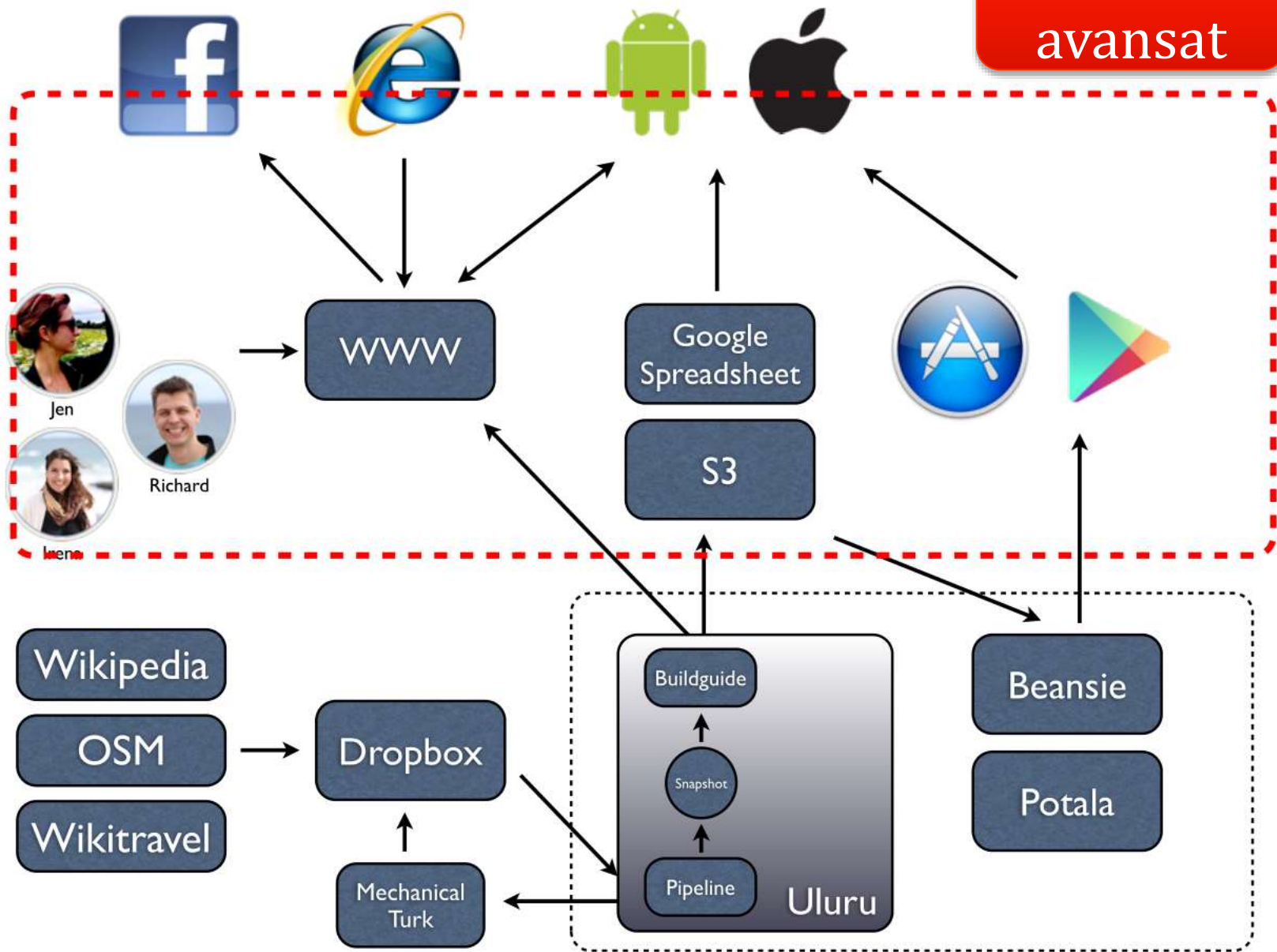
(Ionuț Dănilă & Mihaela Ghimiciu, 2013—2014)

servicii publice folosite: ABBYY Cloud OCR, AlchemyAPI, Google Places API, YouTube API,...

biblioteci: Flickr.NET, GART (*Geo Augmented Reality Toolkit*), Hammock, TweetSharp etc.

<https://www.youtube.com/watch?v=wygXE6hQ07c>

<http://www.slideshare.net/ionutdanila/ubi-guide>



Ce-ar fi să indicăm datele de intrare și  
răspunsul oferit de serviciul Web  
într-un format standardizat?

# invocare

Necesitatea unui **protocol de comunicare**  
(**transport**) între platforme/aplicații eterogene

# invocare

Protocolul va trebui să ofere un mecanism de **invocare** și de **transmitere** structurată a datelor



# invocare

Protocolul va trebui să ofere un mecanism de **invocare** și de **transmitere** structurată a datelor

facilitarea de interacțiuni complexe între aplicații

asigurarea extensibilității + securitate, fiabilitate, *caching*

# invocare: soluție

## XML-RPC (1999)

simplic de utilizat, nepretențios

bazat pe RPC (*Remote Procedure Call*)

mesajele sunt modelate în XML

<http://xmlrpc.scripting.com/spec>

# invocare: soluție

## SOAP

sofisticat, mai flexibil

suită de standarde W3C (2007)

utilizat cu precădere în aplicații de tip *enterprise*

<http://www.w3.org/TR/soap12/>

# invocare: soluție

Recurgerea la alte reprezentări

CSV (*Comma Separated Values*)

POX (*Plain Old XML*)

JSON (*JavaScript Object Notation*)

dezvoltatorul realizează metode proprii de serializare

# invocare: soap

Scop:

protocol de comunicație între două mașini  
(client și server) pentru interschimb de date XML,  
indiferent de platformă/limbaj de programare

# invocare: soap

Standard al Consorțiului Web (2003, 2007)

[www.w3.org/TR/soap12-part0/](http://www.w3.org/TR/soap12-part0/)

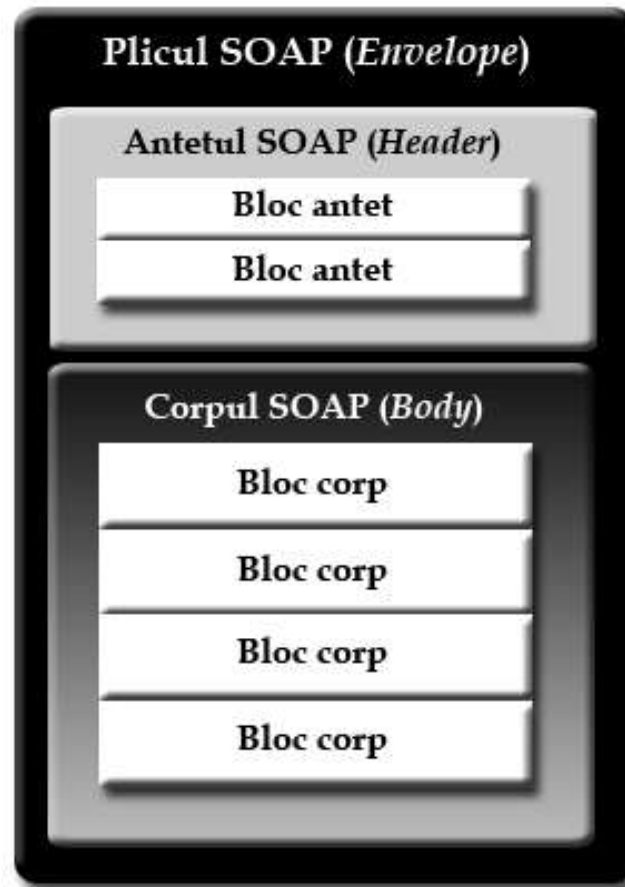
# invocare: soap

„Dialogul” dintre aplicații are loc  
via **mesaje SOAP**  $\equiv$  documente XML

**plic + antet și corp**

*XML messaging*

# invocare: soap





# invocare: soap

Se poate descrie un model de procesare  
a conținutului

*SOAP encoding rules*

# invocare: soap

Poate specifica o cale de la expeditor la destinatar,  
via un intermediar (*proxy*) opțional

*SOAP routing*

# invocare: soap

Anteturile pot fi procesate de intermediari diferiți

# invocare: soap

Datele XML din corp pot fi transportate indiferent de protocolul folosit

uzual, HTTP – pot fi adoptate și alte protoacoale  
(*e.g.*, SMTP, XMPP)

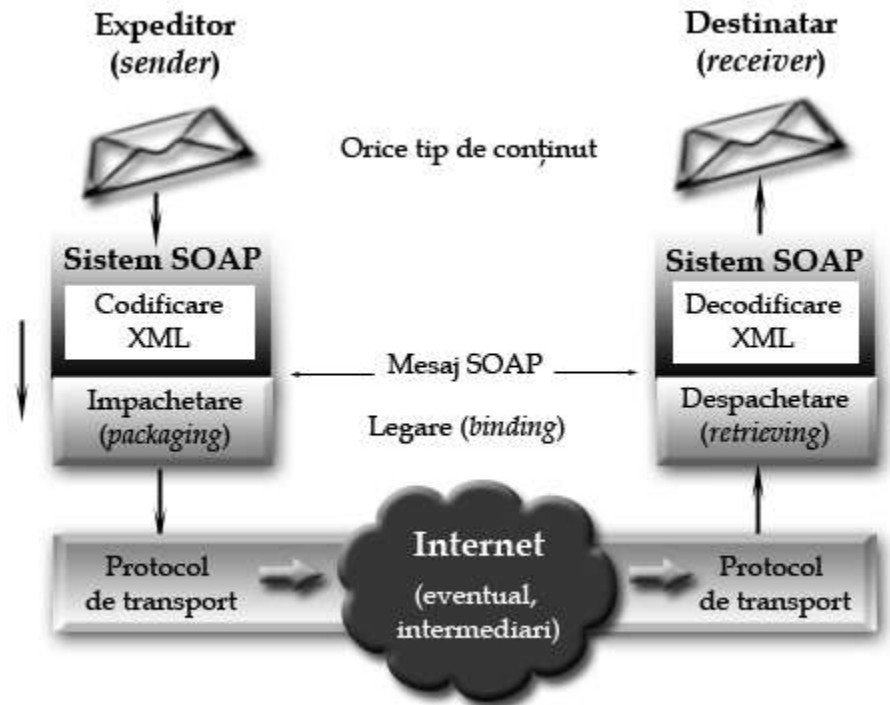
# invocare: soap

Via SOAP poate fi specificat un apel de metodă disponibilă pe alt calculator

*SOAP RPC representation*

# invocare: soap

## Vehicularea mesajelor SOAP



# invocare: soap – exemplu

## Scenariu:

un client trimite serviciului  
un nume de sortiment de portocale (argument de intrare)  
pentru a afla cantitatea disponibilă – *i.e.* răspunsul

# invocare: soap – exemplu

Abordare clasică TCP/IP ☹️

*socket*-uri ► se utilizează un port stabilit de utilizator  
(*e.g.*, 7777) + o convenție de transmitere a datelor



# invocare: soap – exemplu

## Abordare RPC ☹️

apel la o procedură la distanță, executată de server  
(la un port al dispecerului RPC),  
folosind XDR – *External Data Representation* –  
ca protocol binar de (de)serializare a datelor  
implementări tipice în C, C++ sau Java (cu RMI)

# invocare: soap – exemplu

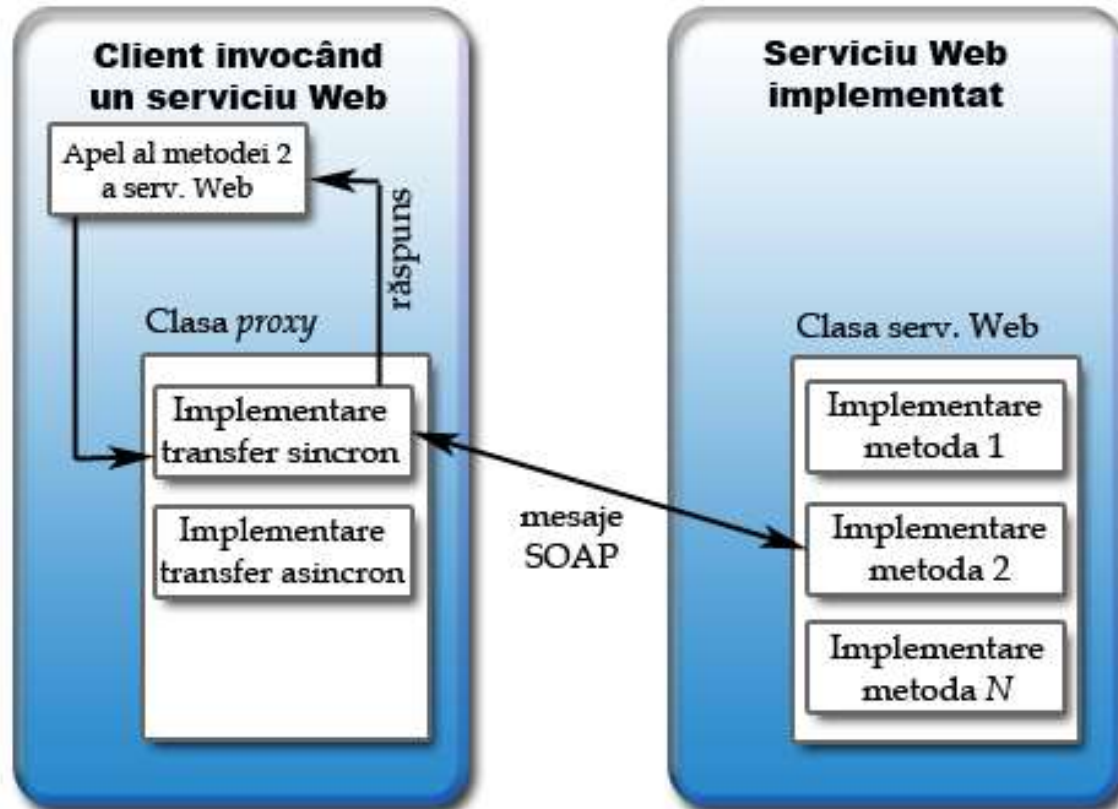
## Abordare SOAP – *XML messaging*

datele vehiculate vor fi împachetate  
de „plicuri” (mesaje) SOAP, transportate via HTTP

independentă de platformă și de limbaj

# invocare: soap

Invocarea unui serviciu Web – în stilul RPC




# invocare: un mesaj SOAP – cerere HTTP

POST <http://undeva.info/portocale/>

Accept: text/xml

Content-Type: text/xml



spațiu de nume  
XML specific SOAP

```
<S-ENV:Envelope
  xmlns:S-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <S-ENV:Body>
    <!-- se apelează metoda de furnizare a stocului -->
    <v:furnizeazaStoc xmlns:v="http://portocale.infoiasi.ro/">
      <!-- date de intrare: numele sortimentului de portocale -->
      <v:arg0>albastre</v:arg0>
    </v:furnizeazaStoc>
  </S-ENV:Body>
</S-ENV:Envelope>
```




parametru  
de intrare

# invocare: un mesaj SOAP – răspuns

HTTP/1.1 200 OK

Content-Type: text/xml; charset="utf-8"

```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <tns:furnizeazaStocResponse
      xmlns:tns="http://portocale.infoiasi.ro/">
      <!-- răspunsul propriu-zis recepționat -->
      <return>33</return>
    </tns:furnizeazaStocResponse>
  </soap:Body>
</soap:Envelope>
```



spațiu de nume XML  
definit de serviciul  
nostru

# semnalarea erorilor (*SOAP fault*)

```
<s:Envelope
  xmlns:s="http://www.w3.org/2003/05/soap-envelope">
  <s:Body>
    <s:Fault>
      <faultcode>flickr.error.0</faultcode>
      <faultstring>Invalid SOAP envelope.</faultstring>
      <faultactor>
        http://www.flickr.com/services/soap/
      </faultactor>
      <details>
        Please see http://www.flickr.com/services/api/
        for more details
      </details>
    </s:Fault>
  </s:Body>
</s:Envelope>
```

aici, invocare eronată  
a serviciului Web  
SOAP oferit de Flickr

# invocare: soap

SOAP  $\equiv$  RPC la nivel de Web

cerere+răspuns incluzând parametri de  
intrare/ieșire (+tipurile lor specificate în XML)

# invocare: soap

SOAP  $\equiv$  protocol de mesagerie (serializare)

cererea conține un obiect-cerere serializat

răspunsul include un obiect-răspuns serializat



# specificare

Necesitatea unui limbaj de descriere  
a serviciilor Web

# specificare

Necesitatea unui limbaj de descriere  
a serviciilor Web

Cum găsim un serviciu Web?

Care este sintaxa mesajelor vehiculate?

Cum se desfășoară transferul de date?

specificare: **wSDL**

WSDL – *Web Service Description Language*

recomandare a Consorțiului Web (2007)

<http://www.w3.org/TR/wsd120/>

# specificare: **wSDL**

Un serviciu Web e descris în format XML  
de un document **.wSDL**

tipurile de date (argumente de intrare + răspuns oferit)  
se definesc via scheme XML

specifică sintaxa, nu semantica unui serviciu

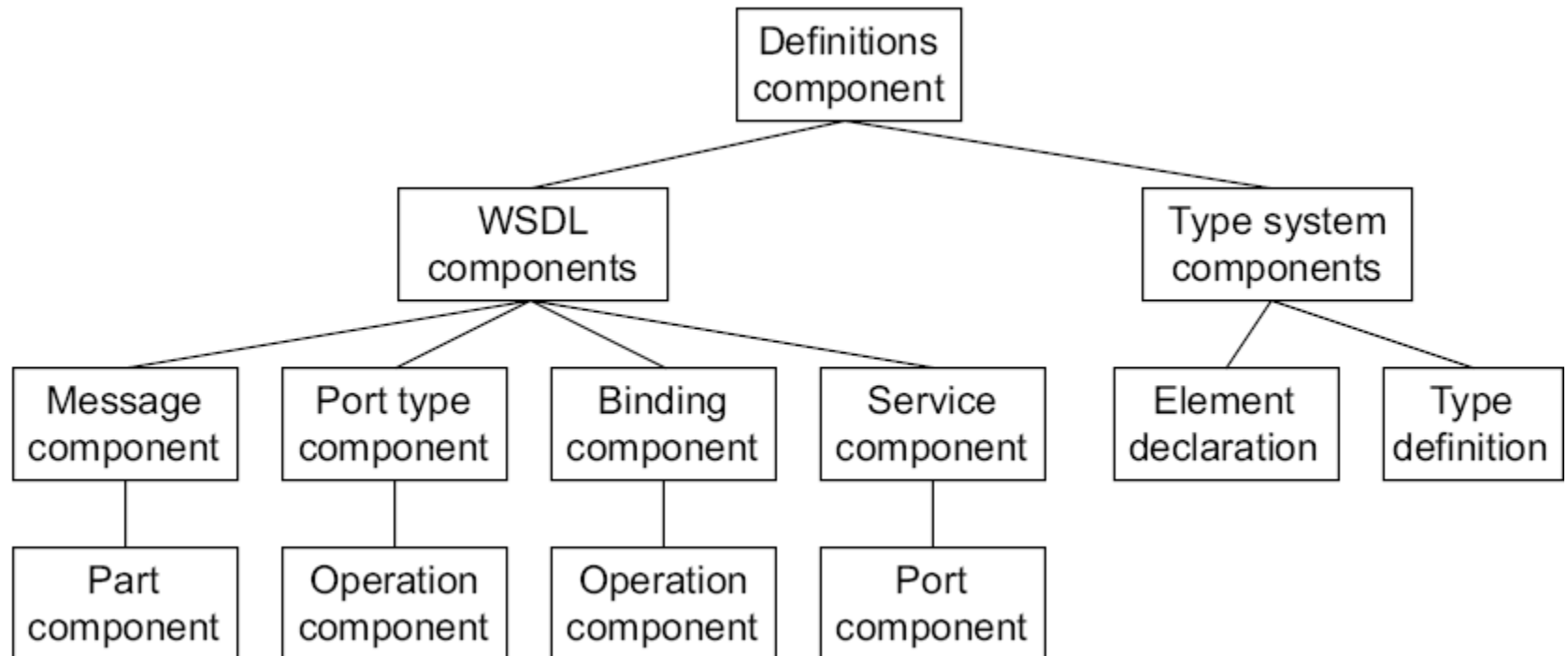
# specificare: **wSDL**

Interfața serviciului (definiție abstractă)	Mesaje ( <i>messages</i> )
	Operatii ( <i>operation</i> )
	Interfață ( <i>interface</i> )
Implementarea serviciului Web (specificație concretă)	Legare ( <i>binding</i> )
	Serviciu ( <i>service</i> )
	Punct terminal ( <i>endpoint</i> )

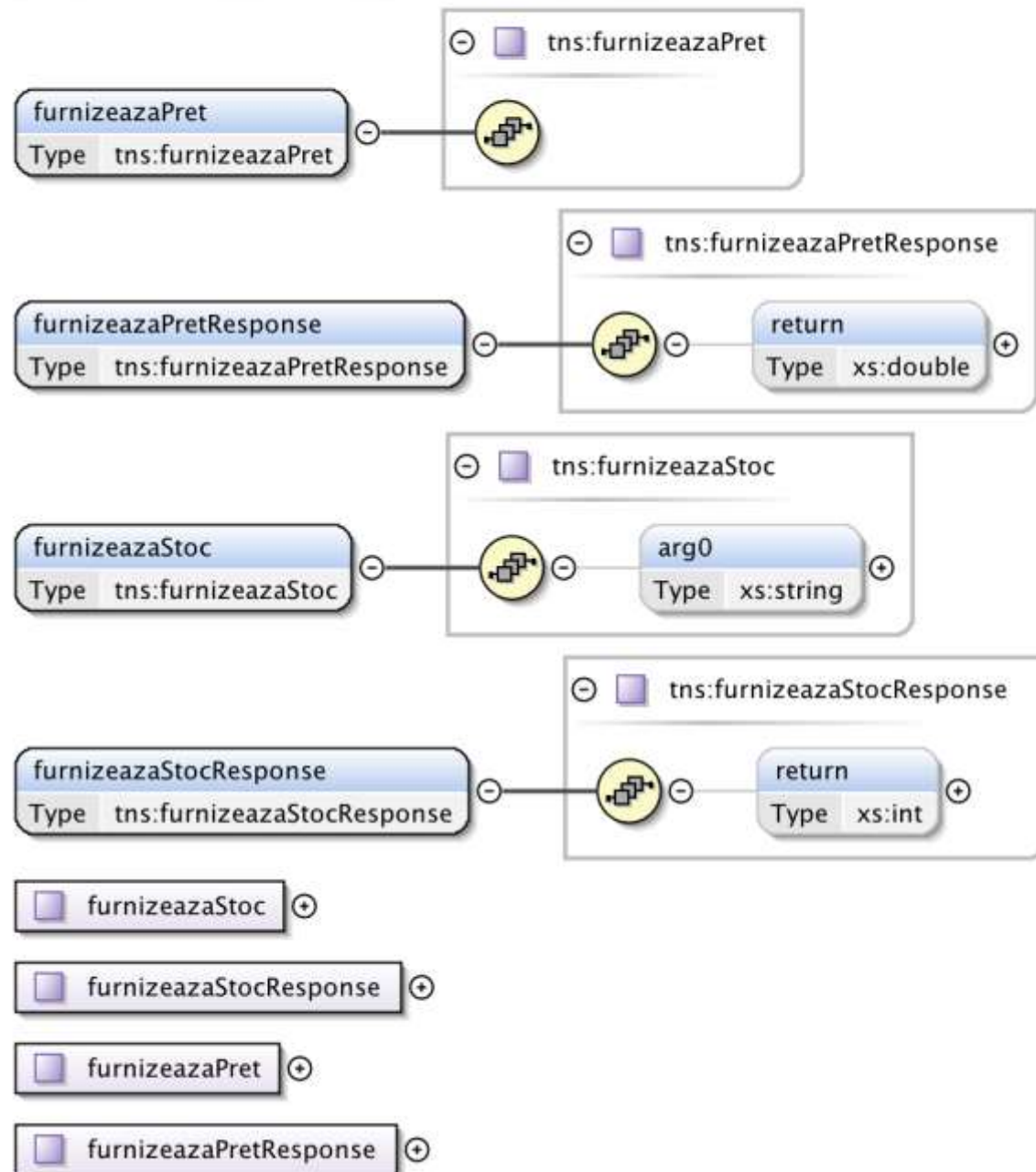
# specificare: **wSDL**

Conceptual, se folosește un model de date reprezentat printr-un set de componente având atașate proprietăți

# specificare: wsdli



schema  
Target Namespace <http://portocale.infoiasi.ro/>



schema XML folosită  
de fișierul WSDL  
al serviciului Web  
vizând stocuri  
de portocale



În ce manieră  
pot fi (re)găsite serviciile Web existente?

# regăsire: uddi

*Universal Description, Discovery, and Integration*

catalog distribuit, universal, al listei de servicii Web  
disponibile (înregistrate)

versiunea curentă: UDDI 3.0.2 – standard OASIS (2004)

[www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm](http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm)

# regăsire: uddi

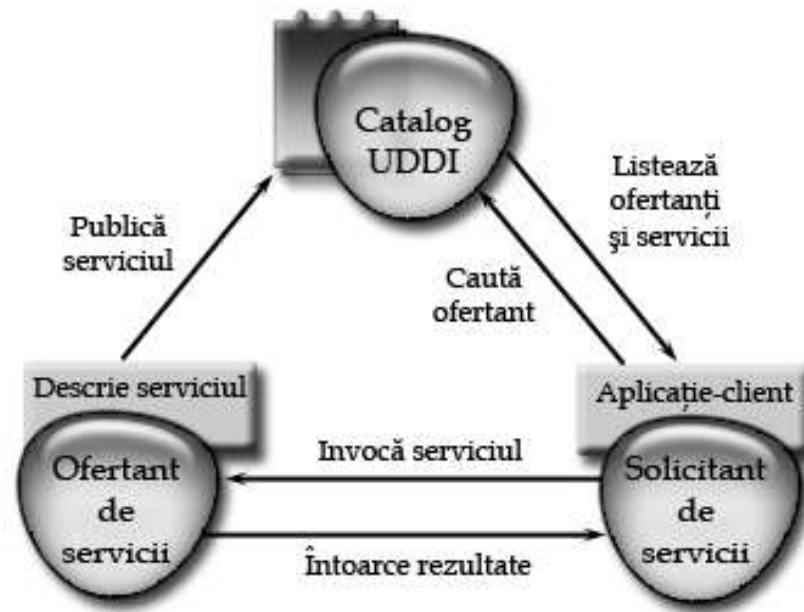
*Universal Description, Discovery, and Integration*

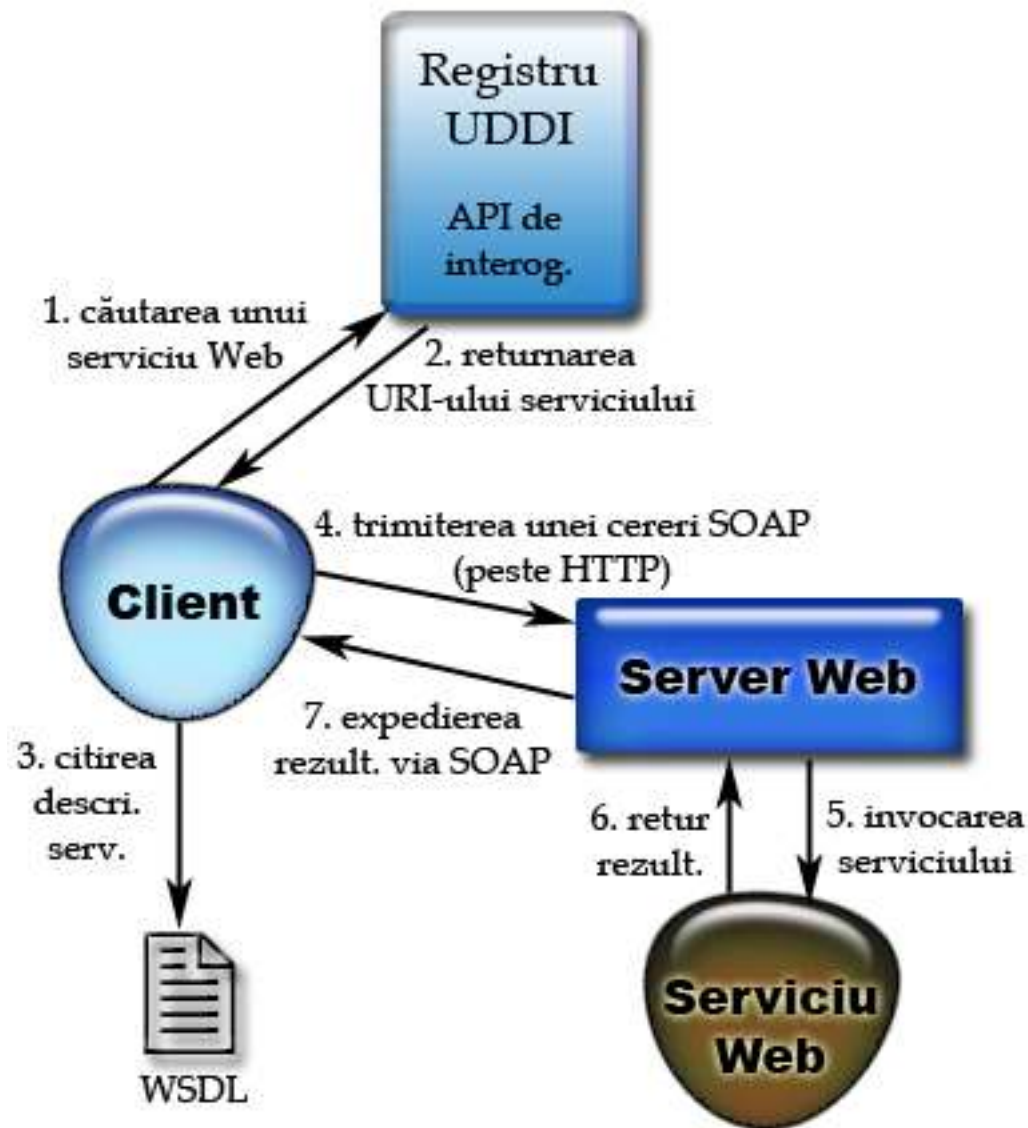
UDDI este în fapt serviciu Web, invocabil prin SOAP

înregistrările sunt replicate

actualmente, disponibil la nivel privat (*enterprise*)

# regasire: uddi





# regasire: uddi – implementare

Instrumente software – exemple:

**jUDDI** (Apache): <http://juddi.apache.org/>  
permite gestionarea unui catalog UDDI privat

## SOA Client

extensie pentru Firefox

realizează interogări asupra unui catalog UDDI  
<https://addons.mozilla.org/firefox/addon/soa-client/>

Access Web Services   Access UDDI Registries   Advanced Access   SOA Client   About

### Access UDDI Registries

UDDI Inquiry API Address:

Search for:  in

Communication: ☐ sync ☒ async UDDI API Version: ☐ v1 ☒ v2 ☐ v3

Search Result:

- 3. Service Name: Eurimage Quickbird Products**
  - Service Key: 9591B0F0-DB88-11DB-B14D-B4612BBE8D3F
  - Business Key: 956D8720-DB88-11DB-B14D-8E96DC4DEB15
- 4. Service Name: HM Products**
  - Service Key: 90B8B1A0-DB88-11DB-B14D-839F52733CFF
  - Business Key: 8DCE3BE0-DB88-11DB-B14D-C096BF2FAE5C
- 5. Service Name: MERIS RR Cloud-free Product for Cat-1 Users Order**

Raw Request Header   Raw Request Body   Raw Response Header   Raw Response Body

```
<?xml version="1.0" encoding="UTF-8"?><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Body>
<serviceList generic="2.0" operator="jUDDI.org" xmlns="urn:uddi-org:api_v2">
<serviceInfos>
<serviceInfo businessKey="B060CA10-DB88-11DB-B14D-C773432D3686" serviceKey="B233B550-DB88-11DB-
B14D-CE241DB0545F">
<name>Dry Matter Productivity Basic</name>
```

Status: 200 OK

interogări asupra unui registru **jUDDI** via extensia **SOA Client**

# specificații & inițiative adiționale (WS-\*)

Adresare: WS-Addressing

Descoperire: WS-Inspection, WS-Discovery

Mesagerie: Reliable HTTP (HTTPR),  
WS Attachments, WS-Routing,...

Securitate și autorizare:  
WS-Security, WS-Trust, WS-Policy,...

Procesarea tranzacțiilor:  
WS-Coordination, WS-Transaction



# specificații & inițiative adiționale (WS-\*)

Interacțiunea dintre servicii Web și utilizatori:

WS for Remote Portlets (WSRP),  
WS for Interactive Applications (WSIA)

*Workflow*-uri: Business Process Execution Language (BPEL), WS-Choreography, WS Flow Language (WSFL),...

Interoperabilitate – inițiativa WS-I: [www.oasis-ws-i.org](http://www.oasis-ws-i.org)

...și multe altele

Existența serviciilor Web este suficientă?

# dezvoltare

Datele și serviciile trebuie să fie accesibile  
de pe fiecare dispozitiv și de oriunde

a se considera ubicuitatea Web-ului

# dezvoltare

Necesitatea unei infrastructuri  
orientate către servicii

o „magistrală” de comunicare între servicii/componente

# dezvoltare

Noile servicii pot fi compuse din serviciile Web deja existente și accesate în mod transparent

# dezvoltare

Noile servicii pot fi compuse din serviciile Web  
deja existente și accesate în mod transparent

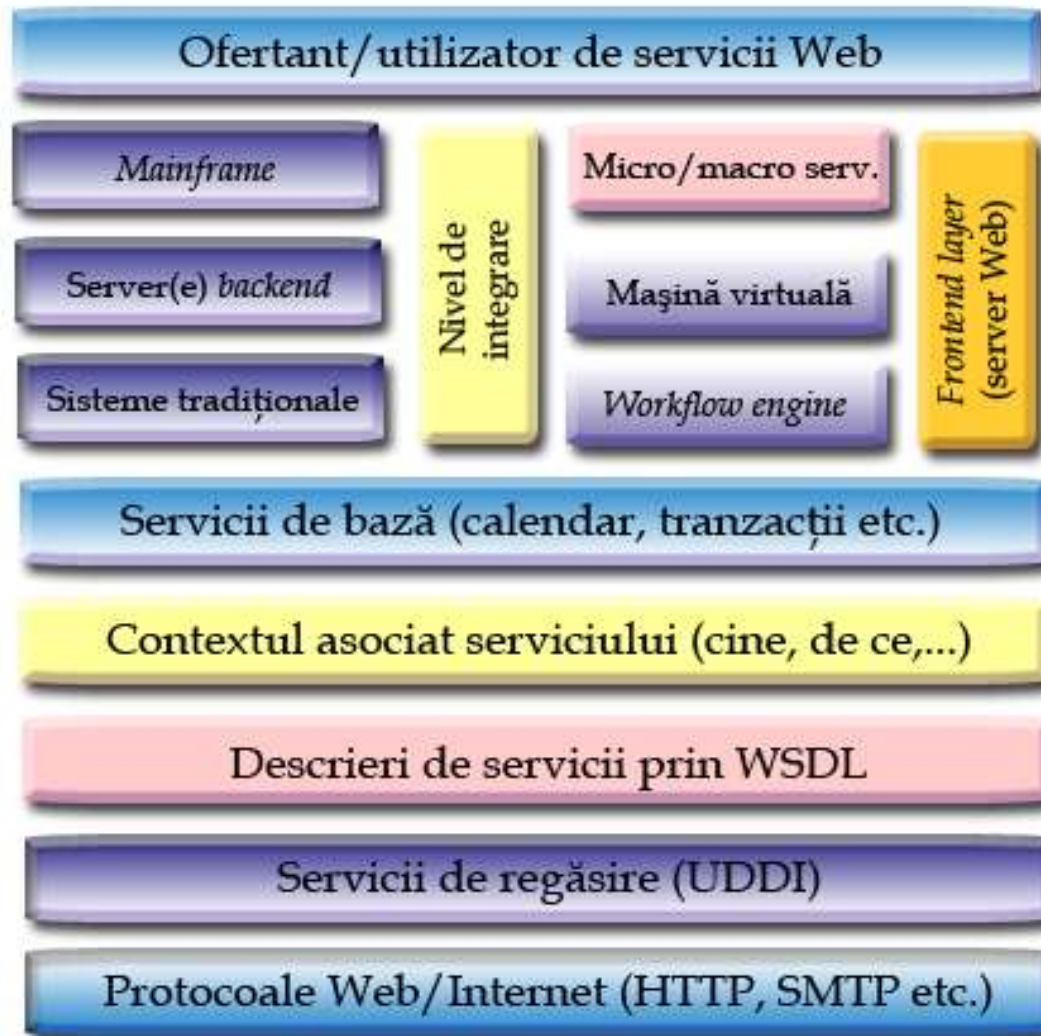
necesitatea unei platforme independente de dispozitiv,  
permițând rularea serviciilor

*middleware* oferind funcționalități + inter-operabilitate

# dezvoltare

Suport pentru conținut Web în stil „vechi”  
(*e.g.*, CGI, servere de aplicații Web) + servicii Web

servere Web  $\equiv$  „porți” spre pagini și/sau servicii Web



*framework-uri Web (structura conceptuală stratificată)*



# dezvoltare

Tehnologii, produse, aplicații – exemplificări:

Apache Axis 2 (Java), Synapse, Tuscany (C++, Java)

Eclipse SOA Tools – <http://www.eclipse.org/soa/>

gSOAP (C, C++) – <http://gsoap2.sourceforge.net/>

implementări Python: osa, soaplib, ZSI etc.

JAX-WS – *Java Architecture for XML: Web Services*

suport nativ în .NET Framework (C#, alte limbaje)

facilități oferite de Play! (Java, Scala)

# dezvoltare

Tehnologii, produse, aplicații – exemplificări:

NuSOAP, PEAR::SOAP (PHP4) + suport nativ în PHP5

Red Hat JBoss Enterprise SOA Platform (Java)

soap, soap-server (module Node.js)

SOAP::Lite (modul Perl)

SOAPEngine (client SOAP pentru aplicații iOS)

WSDL2ObjC (Objective-C)

...și altele

# dezvoltare

Servicii publice prin SOAP – exemple:

Adobe Marketing Cloud, Alexa, Amazon, BankCheck, eBay,  
Google, Faces, Interfax, Microsoft, Monster, NeonCRM,  
PayPal, Shopsync, ScrumWorks, UPS, WalletBit, Yahoo!



**2155 API-uri** (mai 2014) – 1943 (aprilie 2013), 3986 (mai 2012)  
vezi <http://tinyurl.com/34dchvx>

# dezvoltare: studiu de caz

Invocarea unui serviciu Web public  
pe baza descrierii WSDL a acestuia

# dezvoltare: studiu de caz

Invocarea unui serviciu Web public  
pe baza descrierii WSDL a acestuia

utilizăm situl **Programmable Web**

– <http://www.programmableweb.com/> –

pentru a obține lista serviciilor invocabile prin SOAP

## SPECS

API Provider <http://aonaware.com>API Homepage <http://services.aonaware.com/DictService/>

Primary Category Reference

Secondary Categories Dictionary

Protocol / Formats XML, SOAP

APIhub URL

API Kits C#

Contact Email [website@aonaware.com](mailto:website@aonaware.com)

## API MASHUPS (6)

**Molu - The Search dog**

Molu is a super fast vertical meta search engine searching across various categories in a single go. What more, it does not store any of the user history unlike other search engines which track the...

**Scrabble Helper and Cheat**

A Scrabble helper, Scrabble cheat, and Scrabble word finder. When you find a word, you can look it up in the dictionary.

**Molu - Dictionary Search Bot**

This GTalk/Jabber bot searches the dictionary meaning of any given word and gives users the power of information at their fingertips.

## Aonaware Dictionary API

oferă descrierea  
funcționalităților  
via WSDL

acces fără  
autentificare

nu necesită  
cheie de utilizare  
(*developer key*)  
obținută  
în prealabil

**Functions**

DefineInDict  
Define  
DictionaryInfo  
DictionaryListExtended  
DictionaryList  
MatchInDict  
Match  
ServerInfo  
StrategyList

http://services.aonaware.com/DictService/DictService.asmx?WSDL

Browse WSDL

URL-ul descrierii WSDL  
a serviciului Web

**Define**

Request XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:s="http://services.aonaware.com/webservices/">
  <SOAP-ENV:Body>
    <s:Define>
      <s:word>programming</s:word>
    </s:Define>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Call function

Result:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
```

**WSDLBrowser**  
[wsdlbrowser.com](http://wsdlbrowser.com)

oferă lista operațiilor implementate de serviciul Web:  
**Define DictionaryInfo DictionaryList Match** etc.

## Aonaware Dictionary API

folosind specificația WSDL, putem determina structura parametrilor de intrare pentru operația **Define**

```
<s:element name="Define">  
  <s:complexType>  
    <s:sequence>  
      <s:element minOccurs="0" maxOccurs="1"  
        name="word" type="s:string"/>  
    </s:sequence>  
  </s:complexType>  
</s:element>
```



definiție  
XML schema



```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:s="http://services.aonaware.com/webservices/">
  <SOAP-ENV:Body>
    <!-- precizează operația (metoda) ce va fi invocată -->
    <s:Define>
      <!-- parametrul de intrare; aici, un termen (șir de caractere) -->
      <s:word>
        programming
      </s:word>
    </s:Define>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

**Dictionary API – invocarea operației (cerere SOAP)**

## Aonaware Dictionary API

pe baza WSDL, putem cunoaște tipul răspunsului  
furnizat de operația **Define**

```
<s:element name="DefineResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1"
        name="DefineResult"
        type="WordDefinition"/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="WordDefinition">
  <s:sequence>...</s:sequence>
</s:complexType>
```

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <DefineResponse xmlns="http://services.aonaware.com/webservices/">
      <DefineResult>
        <Word>programming</Word>
        <Definitions>
          <Definition>
            <Word>programming</Word>
            <Dictionary><Id>wn</Id><Name>WordNet</Name></Dictionary>
            <WordDefinition>program: a system...</WordDefinition>
          </Definition>
          <Definition> <!-- alte definiții --> </Definition>
        </Definitions>
      </DefineResult>
    </DefineResponse>
  </soap:Body>
</soap:Envelope>
```

**Dictionary API – datele obținute (răspuns SOAP)**

# dezvoltare: php

Clasa **SoapServer** deservește cereri SOAP  
metode utile: **addFunction ()**, **setClass ()**, **handle ()**, **fault ()**

```
try {  
    $server = new SoapServer (null,          // nu oferim nicio descriere WSDL  
        array ('uri' => 'urn:portocale.info')); // spațiul de nume al serviciului Web  
    // adăugăm metodele (funcționalitățile) implementate  
    $server->addFunction ('furnizeazaCantit');  
    $server->handle ();                      // așteptăm cereri SOAP  
} catch (SOAPFault $exception) { die ('Ah, o problemă... ' . $exception); }
```

# dezvoltare: php

Clasa **SoapClient** realizează cereri SOAP către un serviciu  
**\_\_call ()**, **\_\_soapCall ()**, **\_\_doRequest ()**, **\_\_getLastResponse ()**

```
$client = new SoapClient (null, // nu recurgem la WSDL
    array ('location' => 'http://undeva.info/srv.php', // adresa serviciului Web
        'uri' => 'urn:portocale.info')); // spațiul de nume
// realizăm o suită de invocări ale metodei dorite
foreach (array ('albastre', 'japoneze', 'celeste') as $sortiment) {
    $rez = $client->__soapCall ('furnizeazaCantit', array ($sortiment));
    echo "<p>Stocul de portocale $sortiment e $rez.</p>";
}
```

# dezvoltare: java

Utilizarea adnotărilor pentru specificarea serviciului

```
package ro.infoiasi.portocale;
```

```
import javax.jws.WebService;  
import javax.jws.WebMethod;
```

**@WebService**

```
public class Portocale { // clasa ce implementează serviciul Web
```

```
    @WebMethod           // o metoda publică oferind stocul de portocale
```

```
    public Integer furnizeazaStoc (String sortiment) { ... }
```

```
    @WebMethod           // o altă metodă publică furnizând prețul
```

```
    public Double furnizeazaPret () {... }
```

```
}
```

# dezvoltare: java

## Publicarea serviciului Web

```
package ro.infoiasi.portocale.serviciu;  
  
import javax.xml.ws.Endpoint;  
import ro.infoiasi.portocale.*;  
  
public class ServiciuExpus {  
    public static void main (String[] args) {  
        try {  
            // publicăm la URL-ul specificat serviciul Web  
            Endpoint.publish ("http://localhost:8888/porto", new Portocale ());  
        } catch (Exception e) { /* a survenit o excepție... */ }  
    }  
}
```

# dezvoltare: java

Accesarea (consumarea) serviciului Web de către un client

```
public class ClientDorindPortocale {  
    public static void main (String[] args) {  
        try { // instanțiem serviciul pe baza clasei ciot  
            // generate în prealabil cu utilitarul 'wsimport'  
            PortocaleService serviciu = new PortocaleService ();  
            Portocale porto = serviciu.getPortocalePort ();  
            // apelăm metodele expuse de serviciu  
            System.out.println ("Stocul de portocale albastre are valoarea " +  
                porto.furnizeazaStoc ("albastre") * porto.furnizeazaPret ());  
        } catch (Exception e) { /* a survenit o excepție... */ }  
    }  
}
```



# dezvoltare: direcții

Servicii Web bazate pe Java  
conform modelului **ESB** (*Enterprise Service Bus*)  
vezi și proiectul GlassFish – [glassfish.java.net](http://glassfish.java.net)

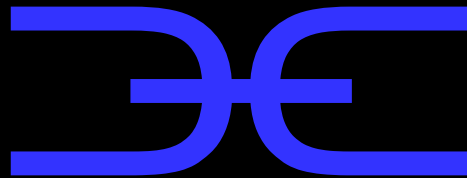
Servicii Web bazate pe .NET  
**WCF** (*Windows Communication Foundation*)  
<http://msdn.microsoft.com/en-us/library/dd456779.aspx>

# servicii web: demo



# rezumat

## servicii Web



„definiții”, caracterizare, arhitectură,  
SOAP, tehnologii și aplicații



# episodul viitor: **servicii Web** prin REST

## Users

[Show/Hide](#) | [List Operations](#) | [Expand Operations](#) | [Raw](#)

POST	/Users/login	Login a user with username/email and password
POST	/Users/logout	Logout a user with access token
GET	/Users/confirm	Confirm a user registration with email verification token
POST	/Users/reset	Reset password for a user with email
GET	/Users/{id}/accessTokens/{fk}	Find a related item by id for accessTokens
DELETE	/Users/{id}/accessTokens/{fk}	Delete a related item by id for accessTokens
PUT	/Users/{id}/accessTokens/{fk}	Update a related item by id for accessTokens
GET	/Users/{id}/accessTokens	Queries accessTokens of User.
POST	/Users/{id}/accessTokens	Creates a new instance in accessTokens of this model.
DELETE	/Users/{id}/accessTokens	Deletes all accessTokens of this model.
GET	/Users/{id}/accessTokens/count	Counts accessTokens of User.
POST	/Users	Create a new instance of the model and persist it into the data source
PUT	/Users	Update an existing model instance or insert a new one into the data source
GET	/Users	Find all instances of the model matched by filter from the data source
GET	/Users/{id}/exists	Check whether a model instance exists in the data source
HEAD	/Users/{id}	Check whether a model instance exists in the data source
GET	/Users/{id}	Find a model instance by id from the data source
DELETE	/Users/{id}	Delete a model instance by id from the data source
PUT	/Users/{id}	Update attributes for a model instance and persist it into the data source