



# **Inter-operabilitatea serviciilor Web**

## **Cu exemple de programe Perl si PHP**

**Dr. Sabin Buraga**

# Cuprins

- SOA & servicii Web
- Dezvoltarea & utilizarea serviciilor Web via instrumente *open-source*
- Inter-operabilitatea
- Concluzii

# SOA & servicii Web



- Originile si scopurile Web-ului
  - Spatiu de comunicare inter-umana prin intermediul partajarii cunostintelor
  - Exploatarea puterii computationale
- Remarci
  - Interactiunea om-Web se rezolva prin intermediul formularelor Web
  - Interactiunea intre masini se desfasoara foarte limitat pe Web

# SOA & servicii Web



- Nevoi ale dezvoltatorilor Web
  - Solutii multi-platforma, slab-conectate
  - Integrare Internet/Web a aplicatiilor, serviciilor si sistemelor
  - Performanta prin asigurarea scalabilitatii
  - Servicii atasabile (*pluggable*) & inteligente
  - “Software as Service” – App. Service Provider*
  - Standardizare
  - Securitate, disponibilitate, mentinere

# SOA & servicii Web



- Nevoia unei arhitecturi pentru dezvoltarea de aplicatii distribuite orientate spre Web
- Software-ul trebuie divizat in servicii care se pot compune, menite a se conecta si orchestra in mod spontan in cadrul proceselor de afaceri/tehnice (*component-based software*)
- Aplicatiile standard (“vechi”) trebuie integrate in noua arhitectura  
⇒ protectia investitiilor

# SOA & servicii Web

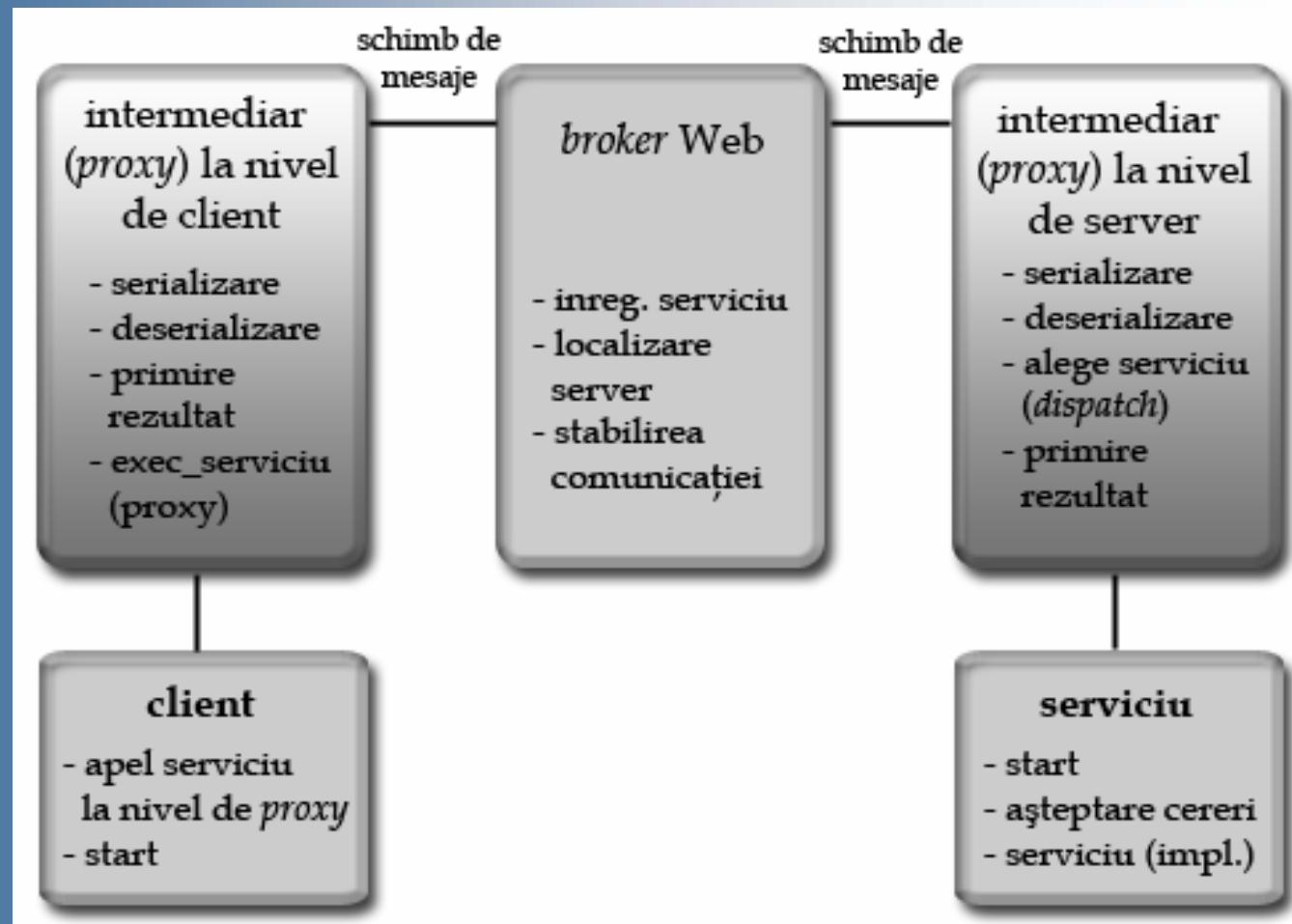


- Solutia: *“The Web is the computer”*
- Crearea unei arhitecturi care...
  - Sa suporte paradigme de comunicare bazata pe Web intre aplicatii
  - Sa ofere localizare transparenta a serviciilor
  - Sa permita adaugarea, inlocuirea, eliminarea serviciilor in mod dinamic
  - Sa ascunda dezvoltatorului detaliile de sistem

# SOA & servicii Web



- Arhitectura – Web-ul ca tehnologie *middleware*



# SOA & servicii Web



- Ce sunt serviciile Web?
  - Aplicatii
  - Utilizate de alte aplicatii (la distanta)
  - Accesate standardizat via Web
    - URI, HTTP, XML
  - “Internet-based modular applications that perform specific business tasks and conform to a **specific technical format**” (Mark Colan – IBM)

# SOA & servicii Web



- Traditional, o aplicatie Web expune o interfata-utilizator
  - Cererile erau capt(ur)ate via formulare
  - Utilizatorii umani trebuie sa interpreteze etichetele si cimpurile de dialog
  - Utilizatorii umani trebuie sa interpreteze raspunsul oferit de serviciu
  - Serviciile Web fac explicite specificatiile implicite!

# SOA & servicii Web



- Caracteristici:
  - Utilizate la interactiunea intre masini
  - Dinamicitatea
  - Lipsa unei cunoasteri *a-priori* a interactiunii cu alte aplicatii si/sau servicii Web
  - Sunt independente de sistem, platforma si limbaj de programare

# SOA & servicii Web

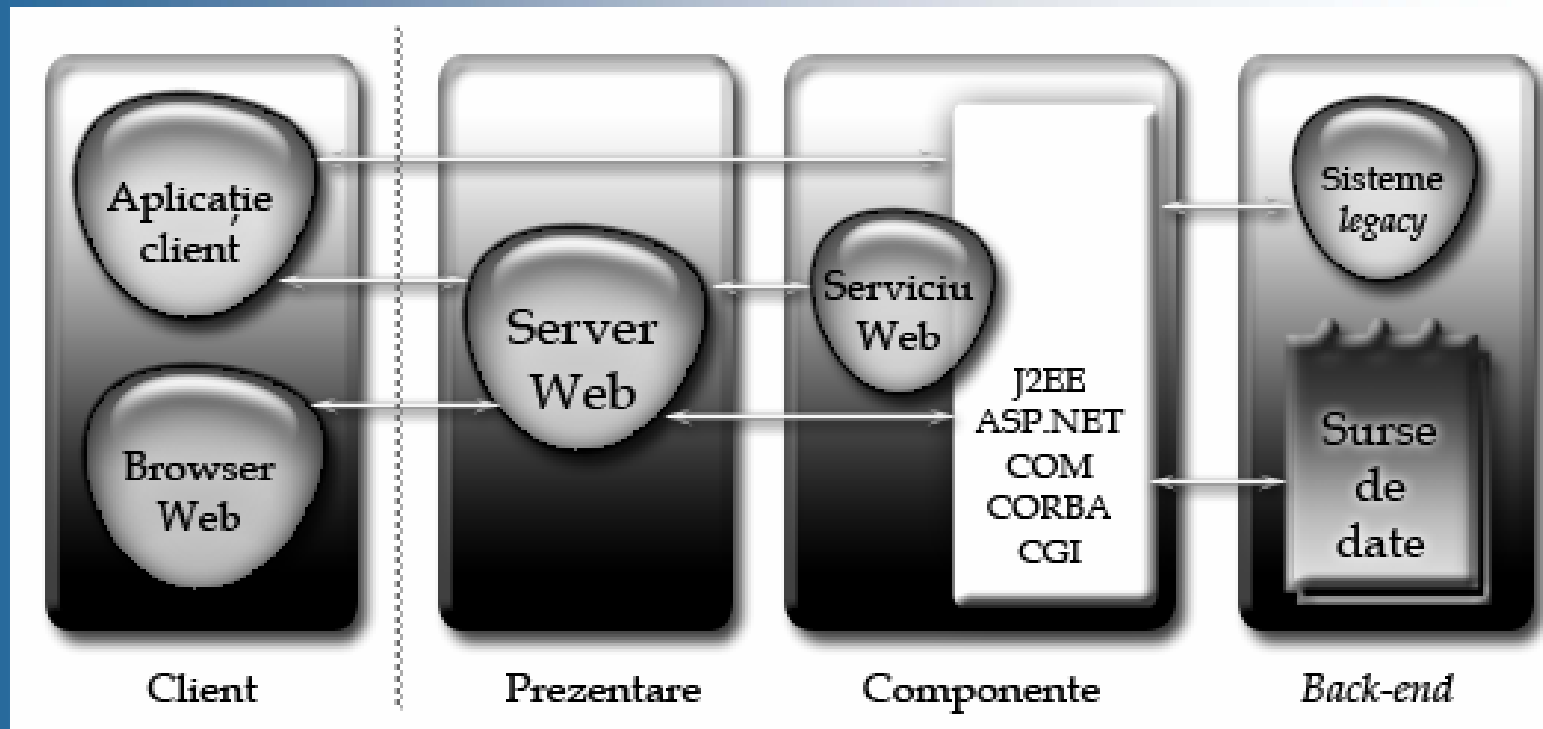


- Caracteristici:
  - Puncte terminale utilizate pentru procesarea datelor, in maniera publica
  - Abilitatea de a prelucra orice tip de date
  - Dezvoltate pe baza platformelor, arhitecturilor si limbajelor curente
  - Deschise si extensibile

# SOA & servicii Web



- Arhitectura orientata spre servicii (SOA - *Service Oriented Architecture*)
  - Stil arhitectural de dezvoltare de aplicatii vazute ca servicii ce pot fi invocate de alte aplicatii



# SOA & servicii Web

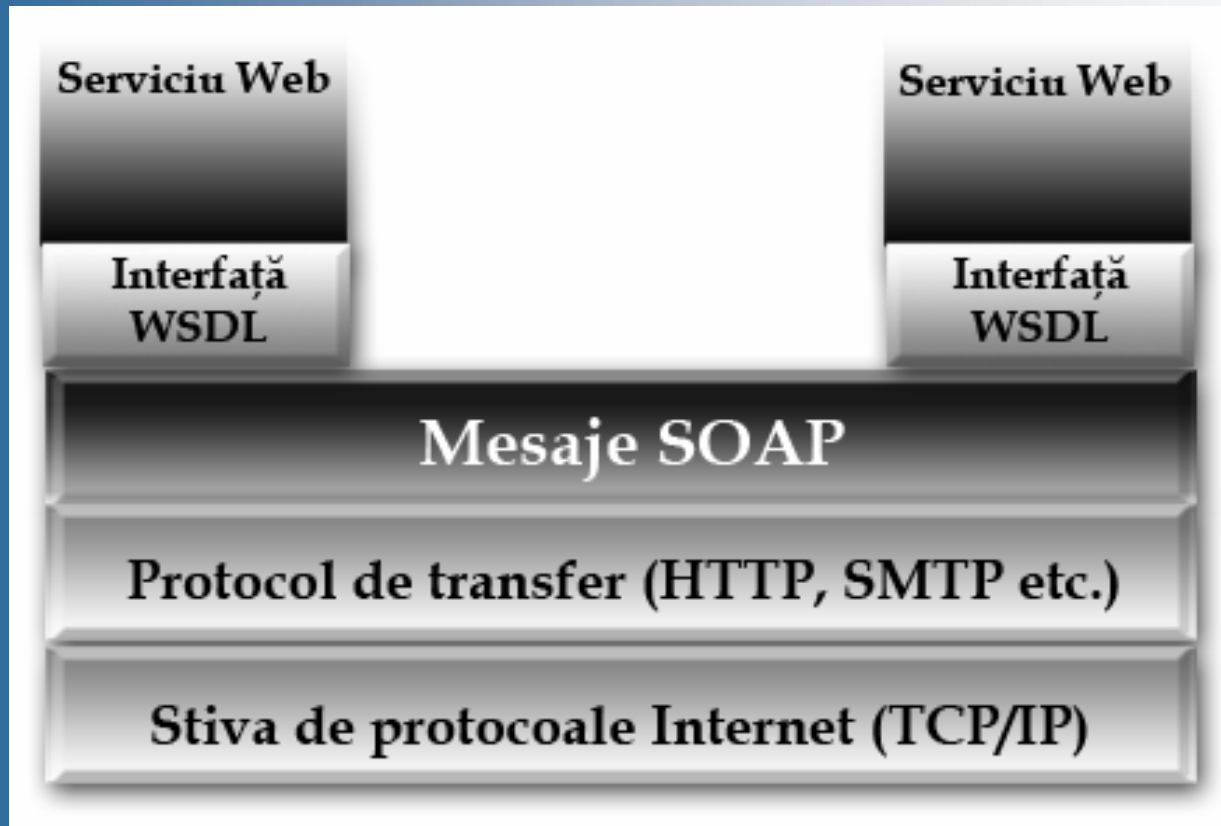


- Tehnologii (bazate pe XML):
  - Descrierea unui serviciu (interfata)  
*WSDL (Web Service Description Language)*
  - Protocol de comunicare via mesaje  
*SOAP*
  - Descoperirea serviciilor Web  
*UDDI (Universal Description, Discovery, and Integration)*

# SOA & servicii Web



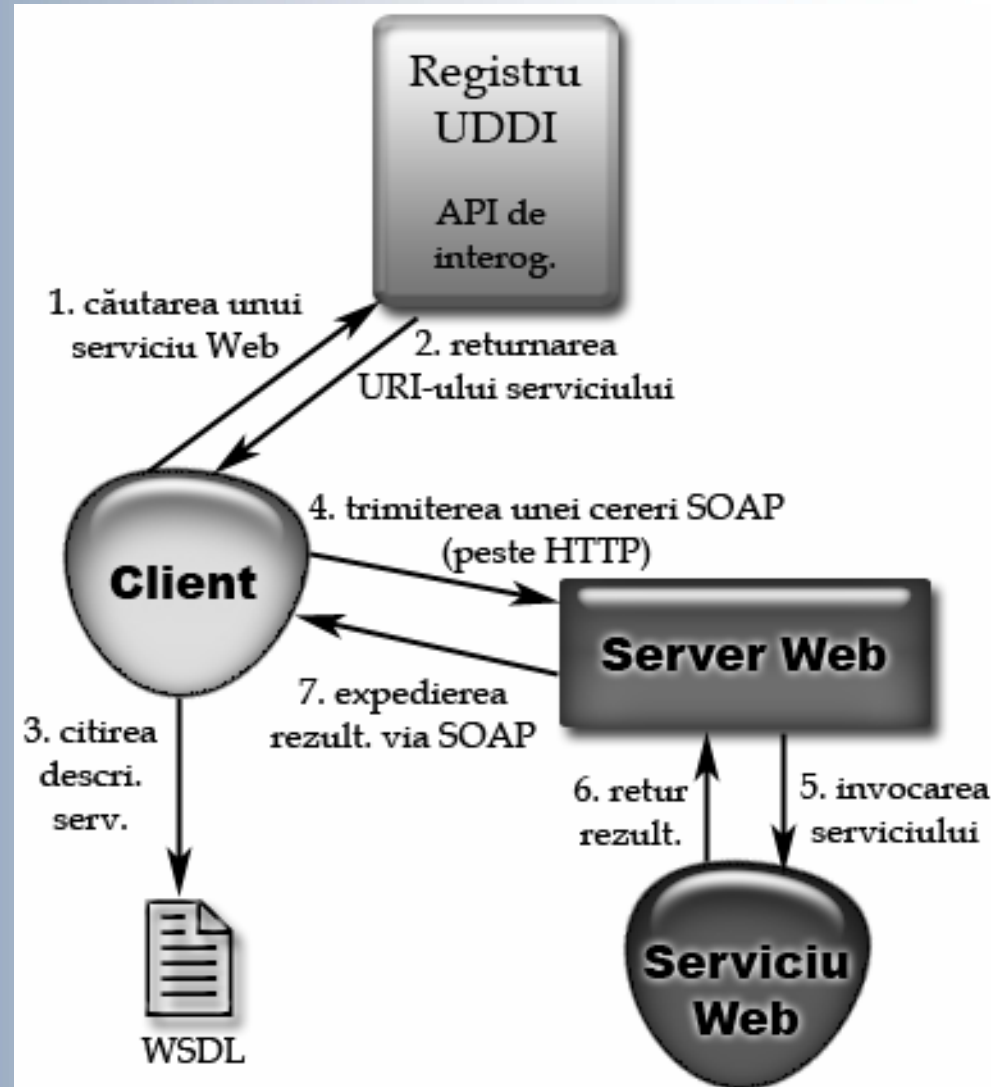
- Stiva de protocoale a serviciilor Web



# SOA & servicii Web



- Maniera de invocare a unui serviciu Web



# SOA & servicii Web



- Instrumente de dezvoltare
  - *Apache Axis2* (C++ / Java)
  - *gSOAP* (C/C++)
  - *.NET Framework* (C#, C++, J#,...)
  - *NuSOAP, PEAR::SOAP* (PHP4)
  - *Soap* (PHP5)
  - *SOAP::Lite* (Perl)
  - *Web Services Developer Pack* (Java)
  - ...

# Instrumente *open source*



- Modulul SOAP::Lite (Perl)
- Implementăm un server SOAP disponibil via HTTP la portul 3333

# inițializăm serverul, folosind un daemon HTTP

```
my $server = SOAP::Transport::HTTP::Daemon->new  
    (LocalAddr => 'localhost', LocalPort => 3333, Reuse => 1);
```

# funcționalitatea serverului e preluată de clasa 'XMLFiles'

```
$server->dispatch_to ('XMLFiles');
```

# ne blocăm, așteptând cereri din partea clienților

```
$server->handle;
```

# Instrumente *open source*



```
sub List { # metoda de generare a listei fişierelor existente
    my $self = shift;
    my $envelope = pop;
    my $files = "";
    # returnam răspunsul ca fiind un fragm. de document XML
    foreach my $file (glob("*")) {
        $files .= "<file name=\"$file\" />";
    }
    return SOAP::Data->type('xml' => $files);
}
```

# Instrumente *open source*



```
sub CheckIfExists { # metoda de verific. a existenței unui fișier
  my $self = shift;
  my $envelope = pop;
  my $filename = $envelope->dataof('//filename');
  # trimitem un 'fault' dacă nu există parametrul de intrare
  die SOAP::Fault->faultcode('Server.CheckIfExists')
    ->faultstring('Input parameter is missing')
    unless $filename;
  # furnizăm dacă există fișierul în cauză
  return (-e $filename->value) ? 1 : 0;
}
```

# Instrumente *open source*



- Un client Perl invocind serviciul Web anterior
- ```
# ne conectăm la server
my $soap = SOAP::Lite->uri('http://127.0.0.1:3333/XMLFiles/')
->proxy('http://127.0.0.1:3333/XMLFiles/xml-files.pl');
# împachetăm explicit parametrii
my @params = (SOAP::Data->name('filename', $ARGV[0]));
# verificăm dacă fișierul exista pe server
$response = $soap->call('CheckIfExists' => @params);
# verificăm dacă a apărut o eroare (SOAP fault)
die 'Eroare: ' . $response->faultstring if $response->fault;
die 'Fișier inaccessibil' if !$response->result;
```

# Instrumente *open source*



POST <http://127.0.0.1:3333/XMLFiles/xml-files.pl> HTTP/1.1

Accept: **text/xml**

Accept: application/soap

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

SOAPAction: "<http://127.0.0.1:3333/XMLFiles/#CheckIfExists>"

```
<soap:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:CheckIfExists xmlns:ns1="http://127.0.0.1:3333/XMLFiles/">
      <filename xsi:type="xsd:string">document.txt</filename>
    </ns1:CheckIfExists>
  </soap:Body>
</soap:Envelope>
```

# Instrumente *open source*



HTTP/1.1 200 OK

Server: libwww-perl-daemon/1.36

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

SOAPServer: SOAP::Lite/Perl/0.65\_5

```
<soap:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns9:CheckIfExistsResponse
      xmlns:ns9="http://127.0.0.1:3333/XMLFiles/">
      <s-gensym27 xsi:type="xsd:int">1</s-gensym27>
    </ns9:CheckIfExistsResponse>
  </soap:Body>
</soap:Envelope>
```

# Instrumente *open source*



- Un client PHP invocind serviciul Web anterior

```
$client = new soapclient('http://endirra.ro:3333/xml-files.pl');  
// stabilim parametrii de intrare ai metodei invocate  
$param = array('filename' => 'document.txt');  
$namespace = 'http://127.0.0.1:3333/XMLFiles/';  
// realizăm apelul  
$rez = $client->call('CheckIfExists',  
    array('parameters' => $param), $namespace, "", true);  
if ($client->fault) { ... } // verificam dacă există vreun fault  
$err = $client->getError();  
if (!$err) { // nu-s erori  
    echo '<p>Fișierul ' . ($rez ? 'exista' : 'e inaccesibil') . '</p>';  
}
```

# Instrumente *open source*



**Rezultat**

Fisierul exista.

**Cerere**

```
POST /xml-files.pl HTTP/1.1
Host: localhost:3333
User-Agent: NuSOAP/0.7.2
Content-Type: text/xml; charset=ISO-8859-1
SOAPAction: ""
Content-Length: 605

<?xml version="1.0" encoding="ISO-8859-1"?><SOAP-ENV:Envelope SOAP-ENV:encoding="utf-8" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"><SOAP-ENV:Header/><SOAP-ENV:Body/></SOAP-ENV:Envelope>
```

**Raspuns**

```
HTTP/1.1 200 OK
Date: Fri, 04 Aug 2006 18:53:20 GMT
Server: libwww-perl-daemon/1.36
Content-Length: 526
Content-Type: text/xml; charset=utf-8
SOAPServer: SOAP::Lite/Perl/0.65_5

<?xml version="1.0" encoding="UTF-8"?><soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Header/><soap:Body/></soap:Envelope>
```

**Depanare**

```
2006-08-04 21:53:15.671876 soapclient: instantiate SOAP with endpoint
2006-08-04 21:53:15.671979 soapclient: call: operation=CheckIfExists,
```

# Instrumente *open source*



- Un client Perl invocind un serviciu Web extern
  - Folosim documentul WSDL descriind interfata (functionalitatile) unui serviciu oferit de XMethods

```
$response = SOAP::Lite
->proxy("http://www.xmethods.net/interfaces/query")
->uri("http://www.xmethods.net/interfaces/query")
->call('getAllServiceNames' => ());
die "Fault: " . $response->faultdetail . " " .
    $response->faultstring if $response->faultcode;
# afișăm o parte dintre serviciile Web întoarse
foreach $r ($response->dataof ('//name')) {
    if ($r->value =~ /^Co/) { # numele începe cu 'Co'
        print $r->value . "\n";
    }
}
```

# Instrumente *open source*



- Raspunsul obtinut in urma rularii clientului Perl

(infoiasi)\$ perl servicii-xmethods.pl

Country Information WebService

Conversions

CountCheat Countdown Service

Congress Member Directory

Code39 Bar Code

ColdFusion Tip-of-the-Day

CountryWebservice

# Inter-operabilitatea



- Serviciul Web poate fi implementat in alte limbaje, pe alte platforme
- Invocarea se poate realiza conform celor prezentate anterior (direct sau via WSDL)
- Apelul unei functionalitati in mod implicit este unul sincron
- Pentru comunicatii asincrone, se poate recurge la suita de tehnologii **AJAX** (*Asynchronous JavaScript And XML*)

# Resurse



- Sabin Buraga, *Tehnologii XML*, Polirom, 2006
- Lenuta Alboaie, Sabin Buraga, *Servicii Web*, Polirom, 2006 (in curs de aparitie)
- [www.webservices.org](http://www.webservices.org)
- [www.soaplite.com](http://www.soaplite.com)
- [dietrich.ganx4.com/nusoap](http://dietrich.ganx4.com/nusoap)
- [www.onlamp.com](http://www.onlamp.com)

# Rezumat



- SOA & servicii Web
- Dezvoltarea & utilizarea serviciilor Web via instrumente *open-source*
- Inter-operabilitatea
- Concluzii



# **Inter-operabilitatea serviciilor Web**

## **Cu exemple de programe Perl si PHP**

Mulumiri pentru atentie!... Intrebari?