Διάλεξη

Web Services,

REST, JSON

Web Services

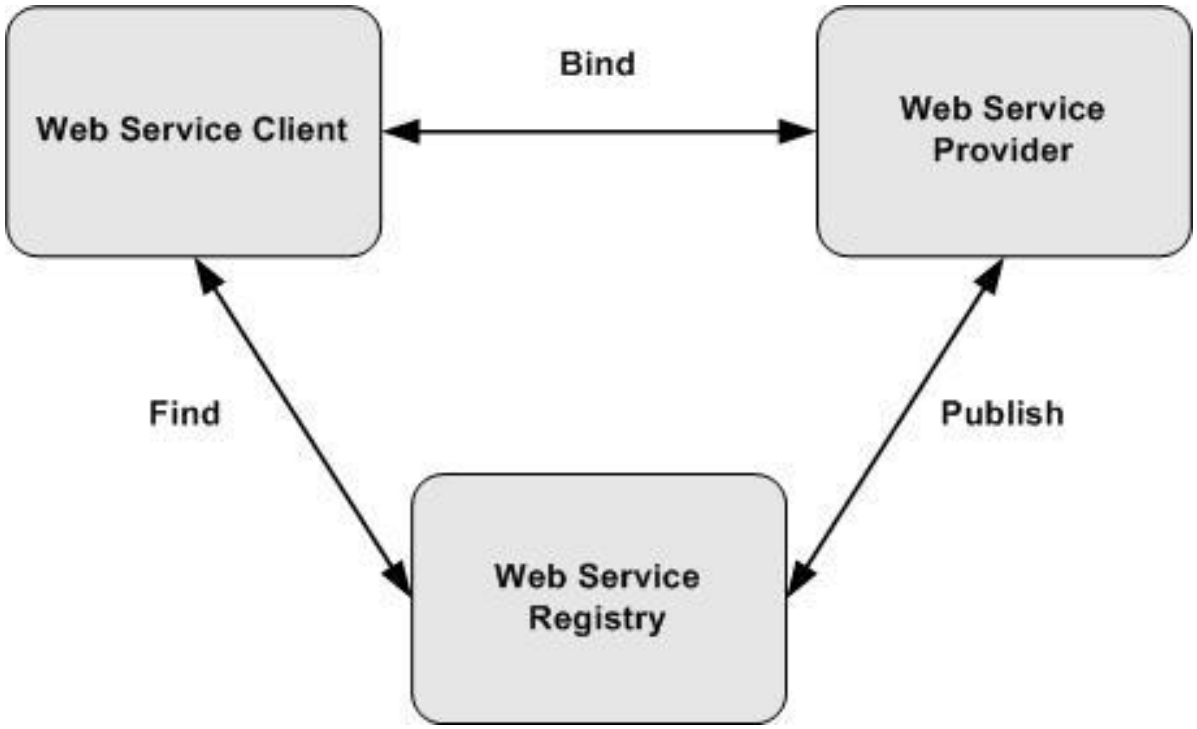
Τι είναι τα Web Services

- Είναι μια τεχνολογία για κατανεμημένα υπολογιστικά συστήματα (όπως οι: CORBA, RMI, EJB, κ.α.) όσον αφορά σε υπηρεσίες request/response.
- Πρόκειται για ευέλικτα software components που παρέχουν ένα πλαίσιο επικοινωνίας μεταξύ συστημάτων και εφαρμογών διαφορετικών τεχνολογιών, καθώς χρησιμοποιούν standard XML - γλώσσες.
- Για τη μετάδοση μηνυμάτων τα Web Services χρησιμοποιούν κυρίως HTTP.
- Πλεονεκτήματα:
  - Διαλειτουργικότητα, λόγω XML
  - Μετάδοση μηνυμάτων και διαμέσω firewalls, λόγω HTTP (χρήση fixed port 80 που τα firewalls επιτρέπουν να περάσει)

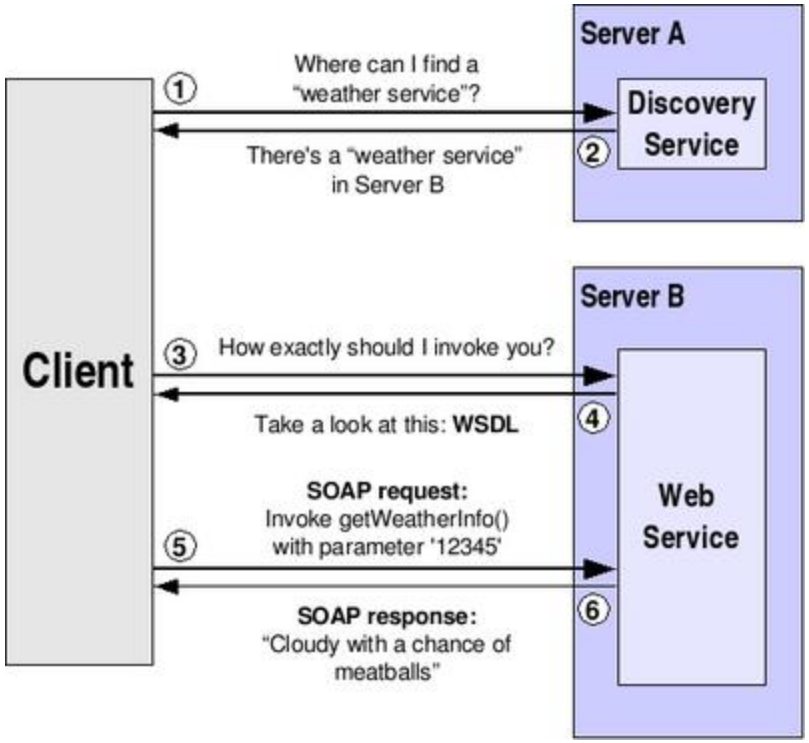
Πλεονεκτήματα σε σχέση με παλαιότερες κατανεμημένες τεχνολογίες

- Ευκολότερος χειρισμός δεδομένων
- Απλότητα πρωτοκόλλου επικοινωνίας
- Απλότητα υποδομής
- Ευκολία στην επικοινωνία
- Διαλειτουργικότητα και ευκολία ανάπτυξης νέων εφαρμογών

Publish, find, bind paradigm



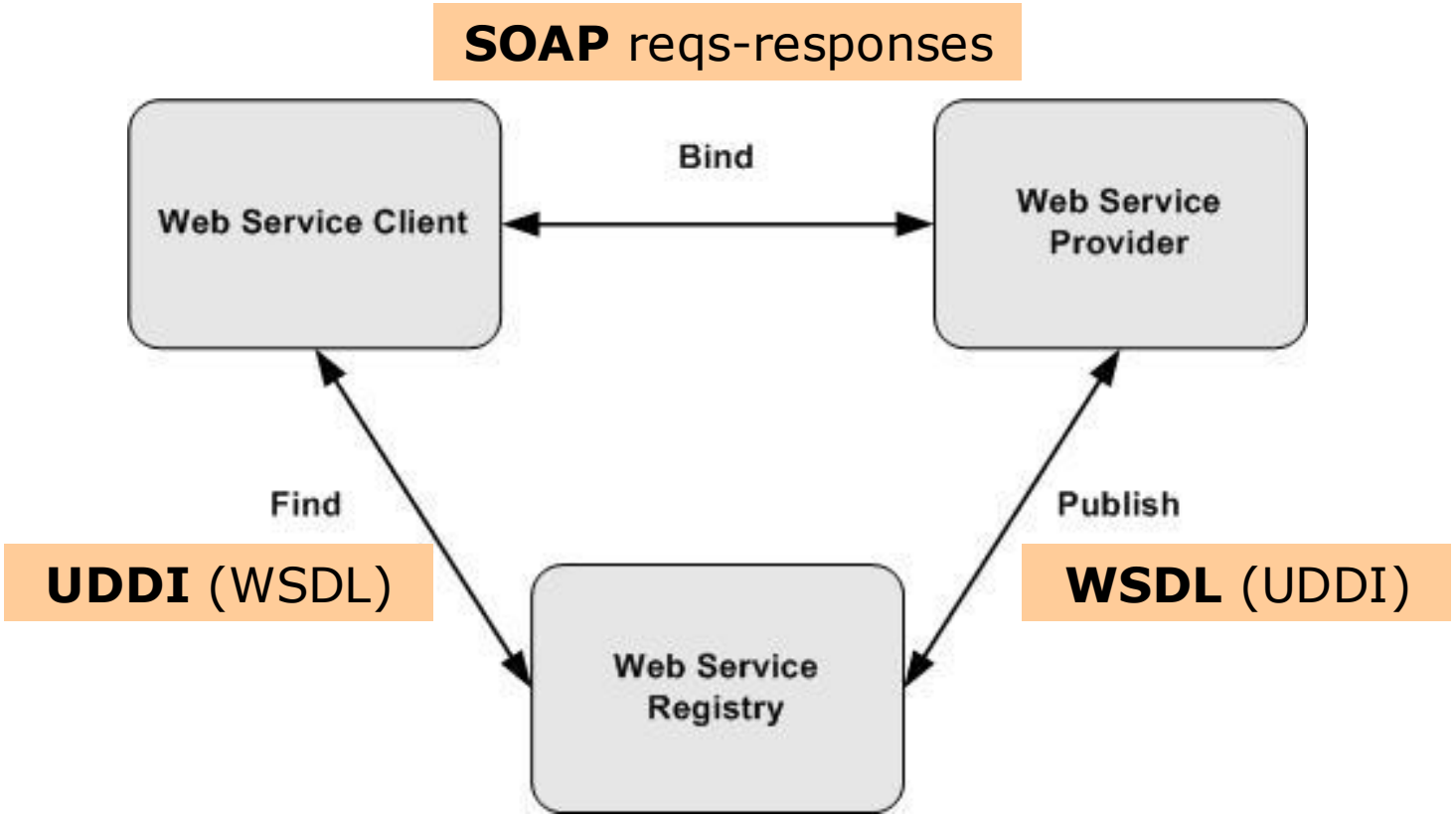
…από μια άλλη οπτική



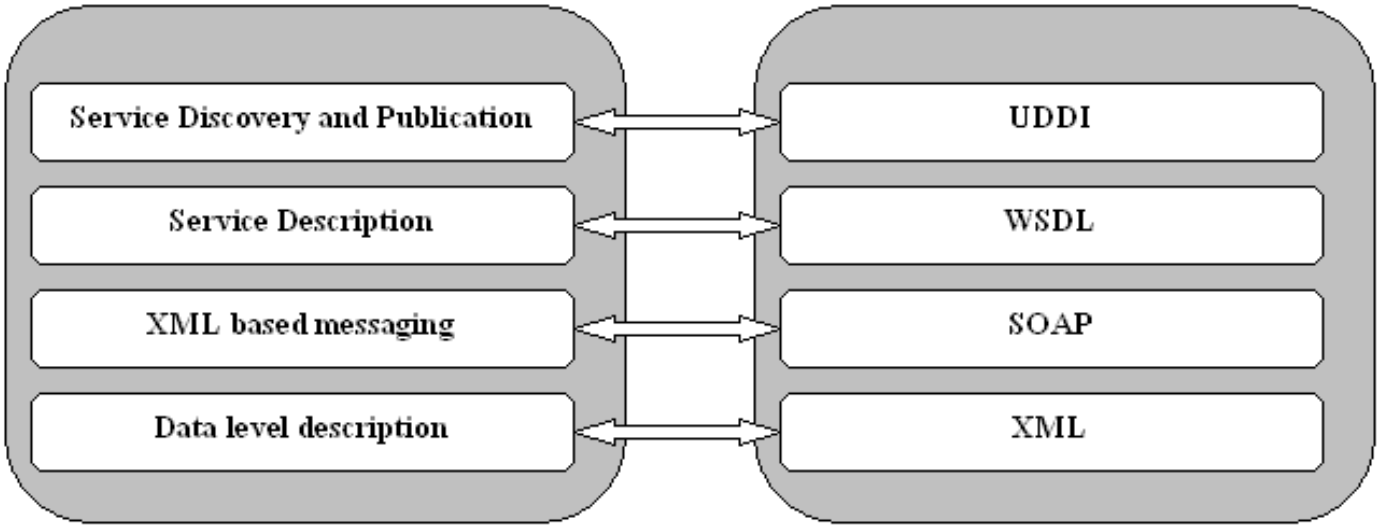
Η διευθυνσιοδότηση των Web services γίνεται μέσω ενός URI: π.χ. http://webservices.mysite.com/weather/us/WeatherService

Οι WS διευθύνσεις προσπελαύνονται μόνο από μηχανές (όχι ανθρώπους)

Τυπική Web Service διάδραση

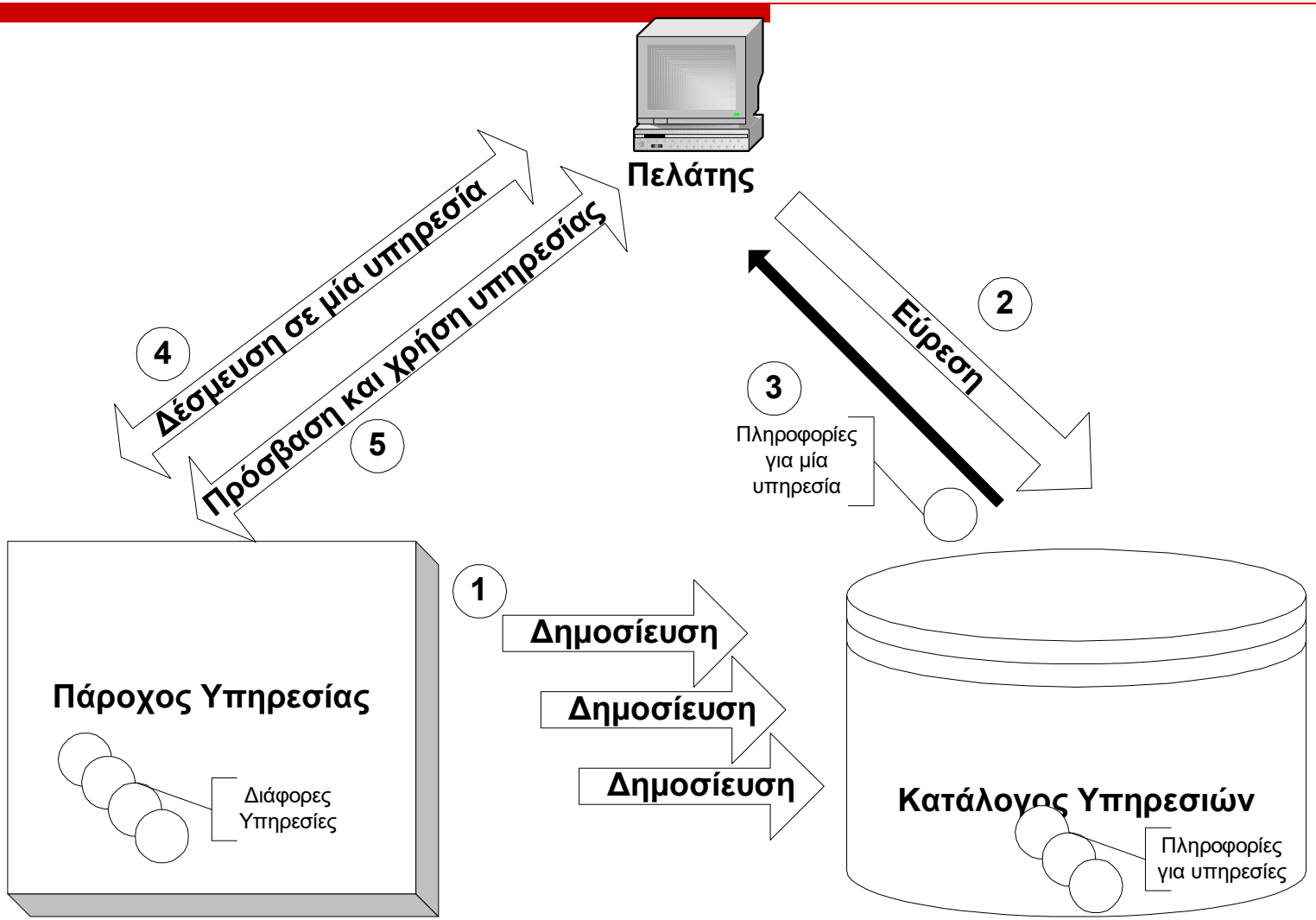


Web Services Network Stack

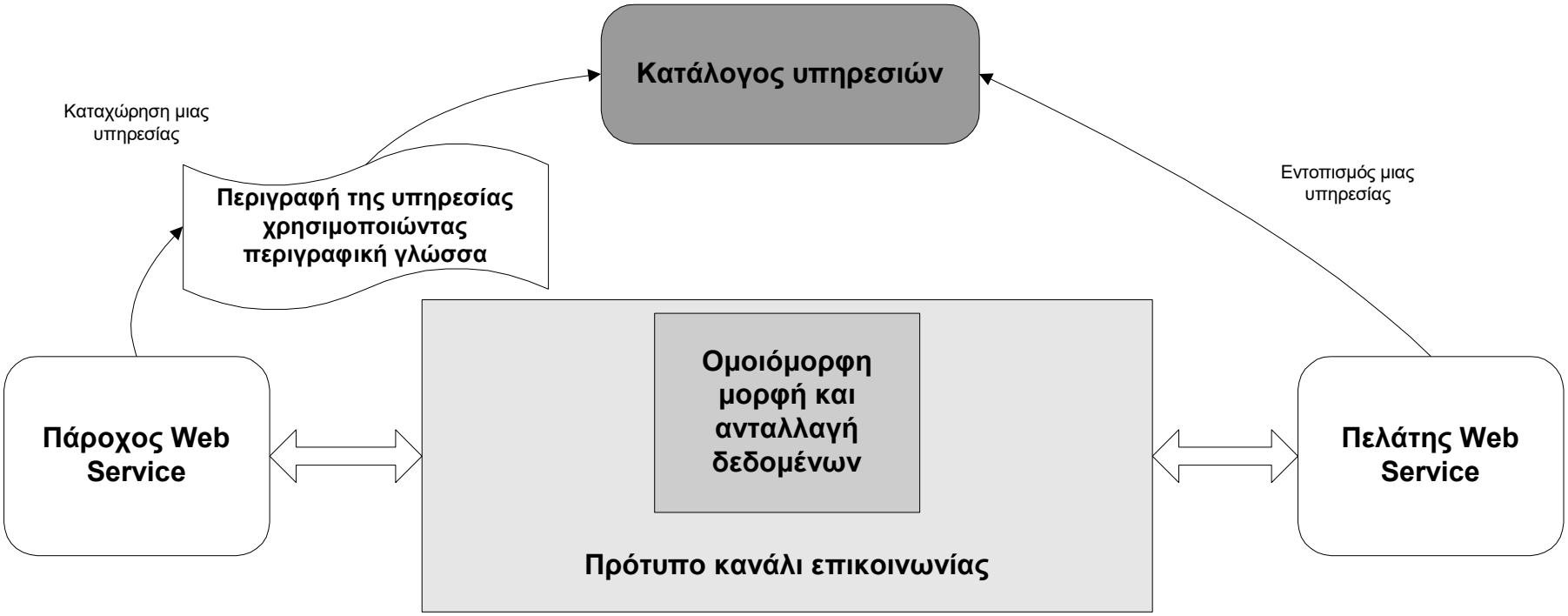


Το UDDI (Universal Description Discovery Integration) παρέχει έναν κατάλογο από Web Services, για να διευκολύνει τους clients να ανακαλύψουν τα WS της επιλογής τους. Ο πάροχος του WS δημοσιεύει την περιγραφή του WS (WSDL) με άλλα στοιχεία αναζήτησης στο μητρώο του UDDI. Ο client χρησιμοποιεί το UDDI για να βρει ένα WS.

Μοντέλο των Web Services



Αρχιτεκτονική των Web Services



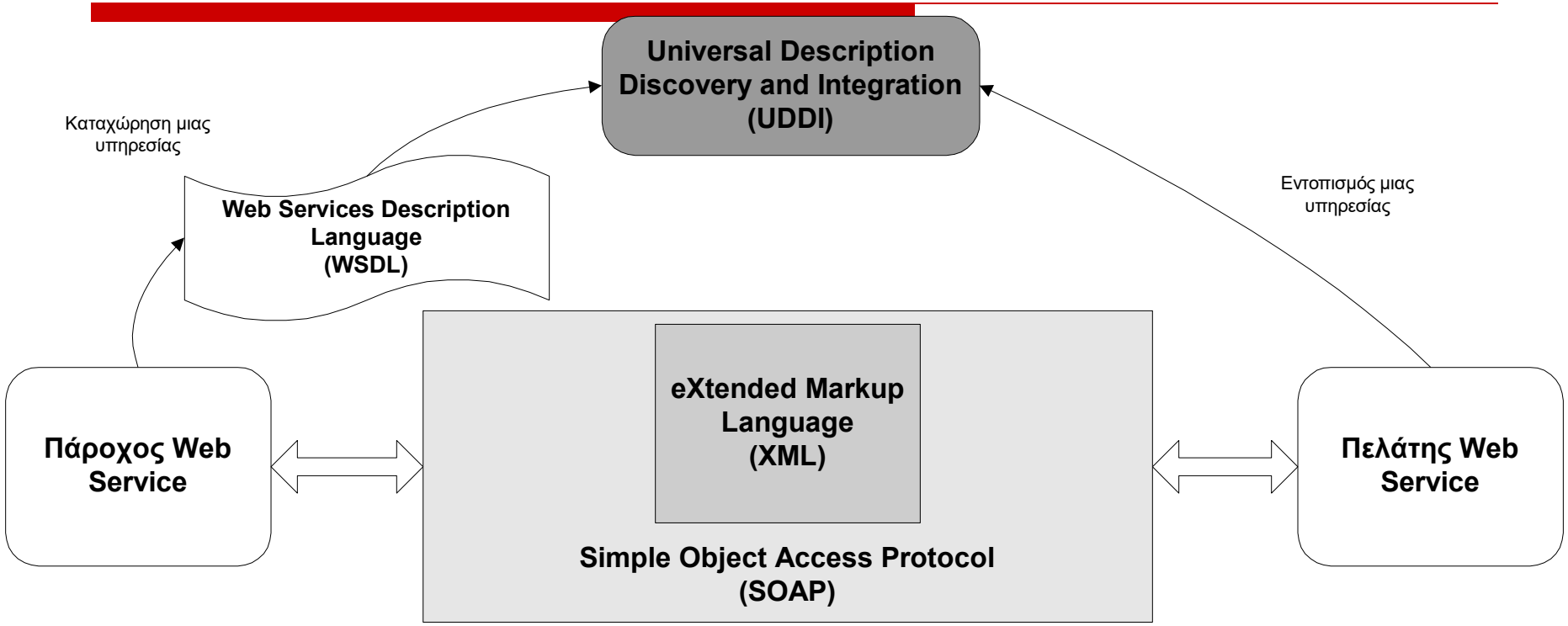
Διαφορές από προηγούμενες τεχνολογίες

- Τα δεδομένα είναι μορφοποιημένα για μεταφορά χρησιμοποιώντας XML, βελτιώνοντας ή εξαλείφοντας το marshalling, το unmarshalling και άλλες απαιτήσεις σχετικά με τη μετάφραση που συνήθως προγραμματίζονταν από τον ίδιο τον προγραμματιστή.
- Τα δεδομένα μεταφέρονται χρησιμοποιώντας προτυποποιημένα πρωτόκολλα όπως το HTTP ή το SMTP, τα οποία έχουν δημοσιευμένα καλά καθορισμένα πρότυπα.
- Η προς έκθεση υπηρεσία είναι καλά καθορισμένη χρησιμοποιώντας ένα γνωστό και αποδεκτό μηχανισμό, την WSDL.
- Οι υπηρεσίες ανευρίσκονται χρησιμοποιώντας

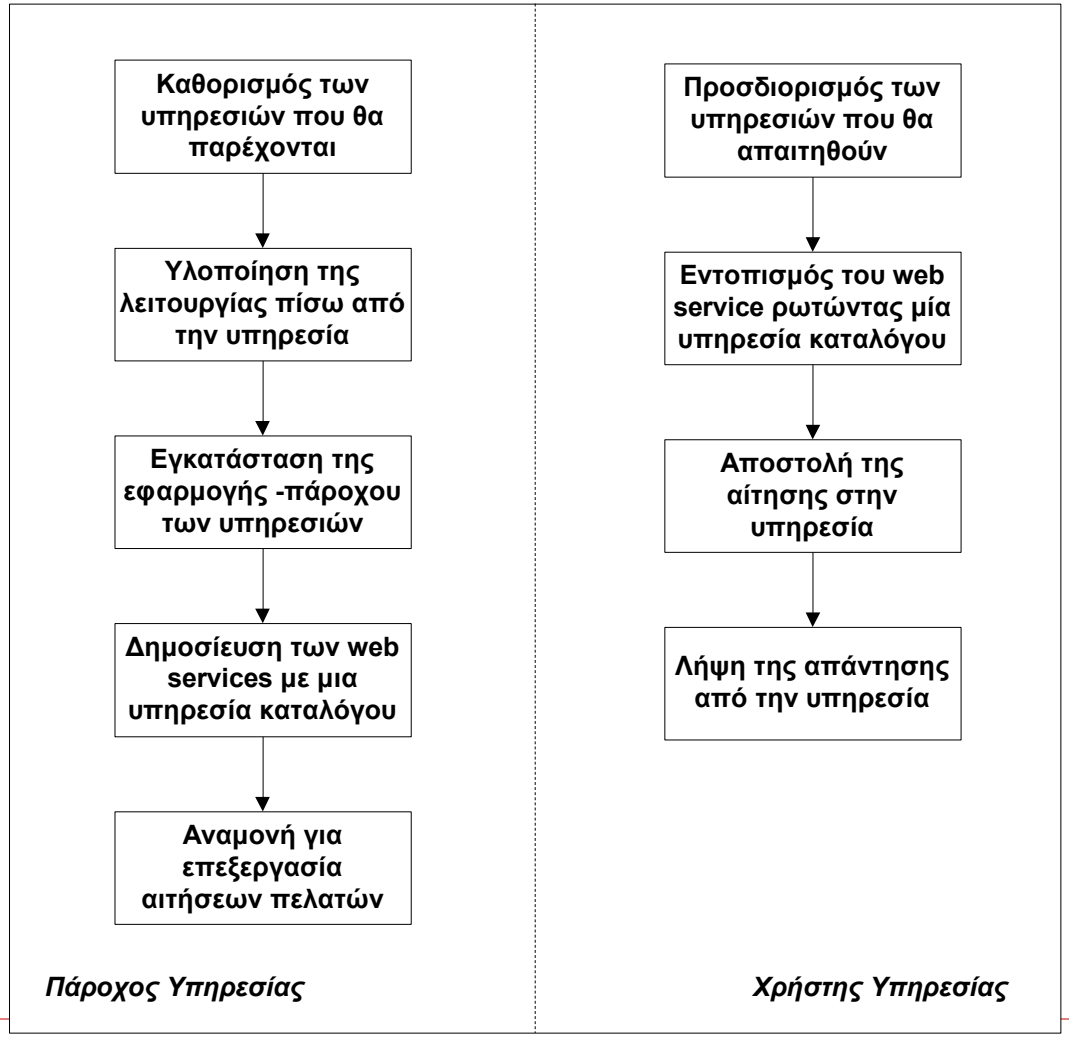
Βασικές Τεχνολογίες των Web Services

Επίπεδο	Τεχνολογία
Ομοιόμορφη μορφή και ανταλλαγή δεδομένων	XML
Πρότυπο κανάλι επικοινωνίας	SOAP
Πρότυπη περιγραφική γλώσσα για την περιγραφή των παρεχόμενων υπηρεσιών	WSDL
Καταχώρηση και εντοπισμός των παρεχόμενων υπηρεσιών	UDDI

Βασικές Τεχνολογίες των Web Services



Βασικά βήματα για την ανάπτυξη εφαρμογών με web services



WSDL

- Η **W**eb **S**ervice **D**escription **L**anguage (WSDL) είναι μια W3C standard XML-based γλώσσα που περιέχει πληροφορίες για το interface, τη σημασιολογία και τους κανόνες κλήσης ενός Web service.
- Η WSDL ορίζει τα Web services σαν μια συλλογή από ports και operations (abstract interface). Περιλαμβάνει επίσης σαφείς οδηγίες για την περιγραφή της διεύθυνσης και πληροφορίες πρωτοκόλλου του Web service.

Αφού δημιουργήσετε ένα Web service, δημοσιεύετε την περιγραφή του και έναν σύνδεσμο προς αυτό στο μητρώο UDDI. Για να χρησιμοποιήσει κάποιος το WS ζητά το WSDL αρχείο του ώστε να βρει την τοποθεσία του WS, τις μεθόδους κλήσης και πώς να τις προσπελάσει. Στη συνέχεια χρησιμοποιεί τις πληροφορίες αυτές για να δημιουργήσει ένα SOAP αίτημα προς εσάς.

<definitions>: Root element του WSDL

<types>: Τί τύπου δεδομένα θα μεταδοθούν;
<message>: Ποια μηνύματα θα μεταδοθούν;
<portType>: Ποιες λειτουργίες (μέθοδοι) υποστηρίζονται;
<binding>: Πώς θα μεταδοθούν τα μηνύματα; Τι SOAP-specific λεπτομέρειες υπάρχουν;
<service>: Σε ποια τοποθεσία βρίσκεται το WS;

WSDL - Δομή

Element	Defines
<types>	The data types used by the web service
<message>	The messages used by the web service
<portType>	The operations performed by the web service
<binding>	The communication protocols used by the web service
<service>	The name and location of the web Service

```
<definitions>

<types>
  definition of types.....
</types>

<message>
  definition of a message....
</message>

<portType>
  definition of a port.......
</portType>

<binding>
  definition of a binding....
</binding>

</definitions>
```

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portType>
```

http://www.w3schools.com/webservices/default.asp

WSDL Ports

One-way

```
<message name="newTermValues">
  <part name="term" type="xs:string"/>
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="setTerm">
    <input name="newTerm"
message="newTermValues"/>
  </operation>
</portType >
```

Request-Response

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portType>
```

Type	Definition
One-way	The operation can receive a message but will not return a response
Request-response	The operation can receive a request and will return a response
Solicit-response	The operation can send a request and will wait for a response
Notification	The operation can send a message but will not wait for a response

WSDL Binding

- **binding element**
  - ***name attribute:*** defines the name of the binding
  - ***type attribute:*** points to the port for the binding
- **soap:binding element**
  - ***style attribute:*** can be "rpc" or "document"
  - ***transport attribute:*** defines the SOAP protocol to use
- **operation element**
  - a SOAP action for each operation
  - definition of the encoding for the input and output

```
<binding type="glossaryTerms" name="b1">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
  <operation>
    <soap:operation soapAction="http://example.com/getTerm"/>
    <input><soap:body use="literal"/></input>
    <output><soap:body use="literal"/></output>
  </operation>
</binding>
```

WSDL – Παράδειγμα (1/3)

```
<?xml version='1.0' encoding='UTF-8'?>
<xs:schema xmlns:tns="http://calculator.me.org/"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           version="1.0"
           targetNamespace="http://calculator.me.org/">

<xs:element name="add" type="tns:add" />

<xs:element name="addResponse" type="tns:addResponse" />

<xs:complexType name="add">
<xs:sequence>
<xs:element name="a" type="xs:int" />
<xs:element name="b" type="xs:int" />
</xs:sequence>
</xs:complexType>

<xs:complexType name="addResponse">
<xs:sequence>
<xs:element name="return" type="xs:int" />
</xs:sequence>
</xs:complexType>
</xs:schema>
```

Το namespace για το συγκεκριμένο web service

Ορισμός δύο complex types που θα χρησιμοποιηθούν από το wsdl

Είσοδος

Έξοδος

WSDL – Παράδειγμα (2/3)

```
<?xml version='1.0' encoding='UTF-8'?>
<definitions xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
             xmlns:wsp="http://www.w3.org/ns/ws-policy"
             xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy"
             xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:tns="http://calculator.me.org/"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns="http://schemas.xmlsoap.org/wsdl/"
             targetNamespace="http://calculator.me.org/"
             name="CalculatorWSService">
    <types>
        <xsd:schema>
            <xsd:import namespace="http://calculator.me.org/"
                        schemaLocation="http://localhost:8080/CalculatorWSApplication/CalculatorWSService?xsd=1" />
        </xsd:schema>
    </types>
    <message name="add">
        <part name="parameters" element="tns:add" />
    </message>
    <message name="addResponse">
        <part name="parameters" element="tns:addResponse" />
    </message>
    <portType name="CalculatorWS">
        <operation name="add">
            <input wsam:Action="http://calculator.me.org/CalculatorWS/addRequest" message="tns:add" />
            <output wsam:Action="http://calculator.me.org/CalculatorWS/addResponse" message="tns:addResponse" />
        </operation>
    </portType>
```

WSDL – Παράδειγμα (3/3)

```
<binding name="CalculatorWSPortBinding" type="tns:CalculatorWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document" />
    <operation name="add">
        <soap:operation soapAction="" />
        <input>
            <soap:body use="literal" />
        </input>
        <output>
            <soap:body use="literal" />
        </output>
    </operation>
</binding>
<service name="CalculatorWSService">
    <port name="CalculatorWSPort" binding="tns:CalculatorWSPortBinding">
        <soap:address location="http://localhost:8080/CalculatorWSApplication/CalculatorWSService" />
    </port>
</service>
</definitions>
```

Πρωτόκολλο επικοινωνίας και τρόπος σύνδεσης SOAP με το WSDL.

Τύπος soap μηνυμάτων

Το όνομα του web service και που βρίσκεται

SOAP

- Το **S**imple **O**bject **A**ccess **P**rotocol (SOAP) είναι ένα W3C standard XML-based πρωτόκολλο για αποστολή μηνυμάτων και υλοποίηση απομακρυσμένων κλήσεων σε ένα κατανεμημένο περιβάλλον. Με χρήση του SOAP τα δεδομένα μπορούν να διαταχθούν χωρίς να ληφθεί υπόψη το πρωτόκολλο μεταφοράς (συνήθως HTTP).
- Το SOAP αποτελεί ένα επεκτάσιμο πλαίσιο επικοινωνίας που επιλύει το πρόβλημα ασυμβατότητας διαφορετικών συστημάτων όσον αφορά στην πρόσβαση στα data.

SOAP Schema namespace

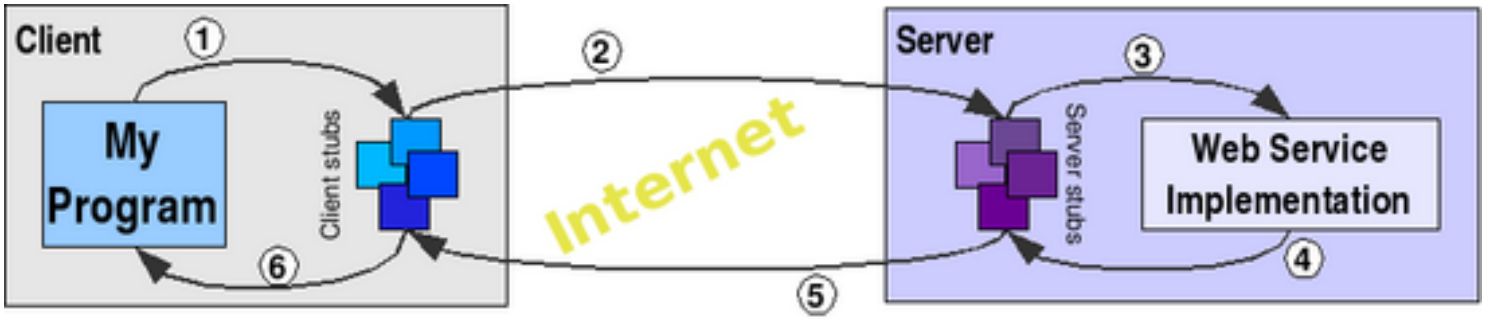
SOAP encodings

```
<SOAP-ENV: Envelope
        xmlns: SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
        SOAP-ENV: encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

<SOAP-ENV:Header>
πληροφορία για ενδιάμεσους processing nodes (π.χ. authentication or authorization server  κ.α.)
– το Header δεν περιέχει πληροφορία σχετική με το body
 </SOAP-ENV:Header>

        <!-- Request  -->
<SOAP-ENV:Body>
   <m:GetLastTradePrice
        xmlns:m="some-URI">
     <symbol>DEF</Symbol>
   </m: GetLastTradePrice>
</SOAP-ENV:Body>

        <!-- Response -->
<SOAP-ENV:Body>
   <m:GetLastTradePriceResponse
        xmlns:m="some-URI">
     <price>22.50</price>
   </m: GetLastTradePriceResponse>
</SOAP-ENV:Body>

</SOAP-Envelope>
```

Get Last Trade Price for symbol DEF

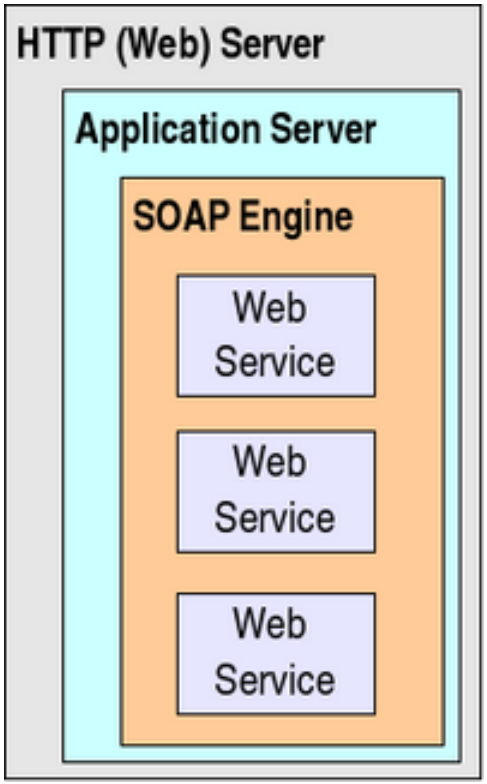
Respond with price 22.50

Stubs



Σε μια WS εφαρμογή, ο κώδικας των SOAP κλήσεων πάντα δημιουργείται και μεταφράζεται αυτόματα. Όταν έρθει η στιγμή για μια client εφαρμογή να προσπελάσει ένα Web Service, ανατίθεται η διαδικασία σε ένα κομμάτι software που λέγεται ***stub***.

Ένας Web Service πάροχος



REST

Τι είναι το REST (1/2)

- Το REST (Representational State Transfer) είναι μια αρχιτεκτονική σχεδιασμού δικτυακών εφαρμογών.
  - Στηρίζεται σε ένα πρωτόκολλο stateless, client-server, cachable. Σχεδόν πάντα HTTP.
  - Βασική φιλοσοφία: Αντί για πολύπλοκους μηχανισμούς, όπως π.χ. CORBA, RPC, SOAP, να χρησιμοποιηθεί απλό HTTP για όλες τις κλήσεις μεταξύ των μηχανών.
  - Πλήρες. Μπορεί να χρησιμοποιηθεί για πολύπλοκες εφαρμογές.

Τι είναι το REST (2/2)

- Το web μπορεί να θεωρηθεί REST εφαρμογή.
- Οι REST εφαρμογές χρησιμοποιούν HTTP μεθόδους για να μεταδόσουν, να ζητήσουν ή να διαγράψουν δεδομένα. HTTP για όλες τις πράξεις CRUD (Create, Read, Update, Delete)

REST ως ελαφρύ web service

- Όπως τα web services, το REST είναι
  - ανεξάρτητο πλατφόρμας
  - ανεξάρτητο γλώσσας
  - βασισμένο σε πρότυπα (HTTP)
  - μπορεί να χρησιμοποιηθεί με firewalls
- Δεν προδιαγράφει
  - ασφάλεια (συνήθως username, password)
  - κρυπτογράφηση (συνήθως HTTPS)

Απλότητα του REST

Web Services, SOAP

```
<?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:body pb="http://www.acme.com/phonebook">
   <pb:GetUserDetails
    <pb:UserID>12345</pb:UserID>
   </pb:GetUserDetails>
  </soap:Body>
</soap:Envelope>
```

REST

http://www.acme.com/phonebook/UserDetails/12345

Πολυπλοκότερο REST

```
http://www.acme.com/phonebook/UserDetails?firstName=John&lastName=Doe
```

- Στις περισσότερες περιπτώσεις μια αίτηση GET είναι αρκετή
- Αν απαιτείται μεγάλο μέγεθος ή δυαδικό είδος παραμέτρων, συνήθως χρησιμοποιείται POST και περνάμε τις παραμέτρους στο σώμα της POST.
- Καλή πρακτική:
  - GET για αιτήσεις που είναι read-only, απλές παράμετροι.
  - POST για όλα τα υπόλοιπα (update, delete, create) και read με πολύπλοκες παραμέτρους.

REST και XML

- Οι REST αιτήσεις (requests) σπάνια χρησιμοποιούν REST.
- Η μορφή XML μπορεί να χρησιμοποιηθεί ώς απόκριση σε μια REST υπηρεσία.
  - Χρήση XML για ευκολότερη πιστοποίηση type safety.

Παράδειγμα XML

```
<employee>
    <shift id= "counter" time="8-12">
        <phone id = "1">
                All phone information
                <number>3444333</number >
        </phone>
    </shift >
    <shift id="help_desk" time="1-5">
        <phone id = "2">
                All phone information
                <number>332333</number >
        </phone>
    </shift >
    <home-address>
        <street>3434 Norwalk street</street>
                <city>New York</city>
        <state>NY</state>
    </home-address>
</employee>
```

Απόκριση REST

- Συχνά ένα αρχείο XML – εύκολο να επεκταθεί, type safety
- Άλλα formats:
  - JSON (JavaScript Object Notation) – εύκολο parsing με Javascript και άλλες γλώσσες
  - CSV (Comma Separated Values) – συμπυκνωμένη μορφή

```
<parts-list>
 <part id="3322">
  <name>ACME Boomerang</name>
  <desc>
   Used by Coyote in <i>Zoom at the Top</i>, 1962
  </desc>
  <price currency="usd"
      quantity="1">17.32</price>
  <uri>http://www.acme.com/parts/3322</uri>
 </part>
 <part id="783">
  <name>ACME Dehydrated Boulders</name>
  <desc>
   Used by Coyote in <i>Scrambled Aches</i>, 1957
  </desc>
  <price currency="usd"
      quantity="pack">19.95</price>
  <uri>http://www.acme.com/parts/783</uri>
 </part>
</parts-list>
```

Αρχιτεκτονική REST (1/4)

- Πόροι (Resources) – αναγνωρίζονται με λογικά URLs. Τόσο η κατάσταση (state) όσο και η λειτουργικότητα (functionality) εμφανίζονται ως πόροι.
  - Τα λογικά URLs σημαίνουν ότι οι πόροι είναι προσβάσιμοι από όλα τα σημεία του συστήματος.
  - Οι πόροι είναι το βασικό συστατικό ενός REST συστήματος. Όλες οι πληροφορίες για ένα αντικείμενο (ή σύνδεσμοι προς αυτές) είναι διαθέσιμες στον πόρο που αντιπροσωπεύει το αντικείμενο.
    - Σε RPC, SOAP services υπάρχουν μέθοδοι ή υπηρεσίες για κάθε τμήμα της πληροφορίας (π.χ. τιμή, ποσότητα, χρώμα, κτλ.)

Αρχιτεκτονική REST (2/4)

- Ιστός από πόρους
  - Ένας πόρος δεν θα πρέπει να είναι υπερβολικά μεγάλος και να περιέχει πολλές λεπτομέρειες.
  - Όπου αρμόζει, ένας πόρος θα πρέπει να περιέχει συνδέσμους προς άλλες πληροφορίες, όπως στο web.
- Client – server.
  - Ο server σε ένα σύστημα μπορεί να είναι ο client σε ένα άλλο.

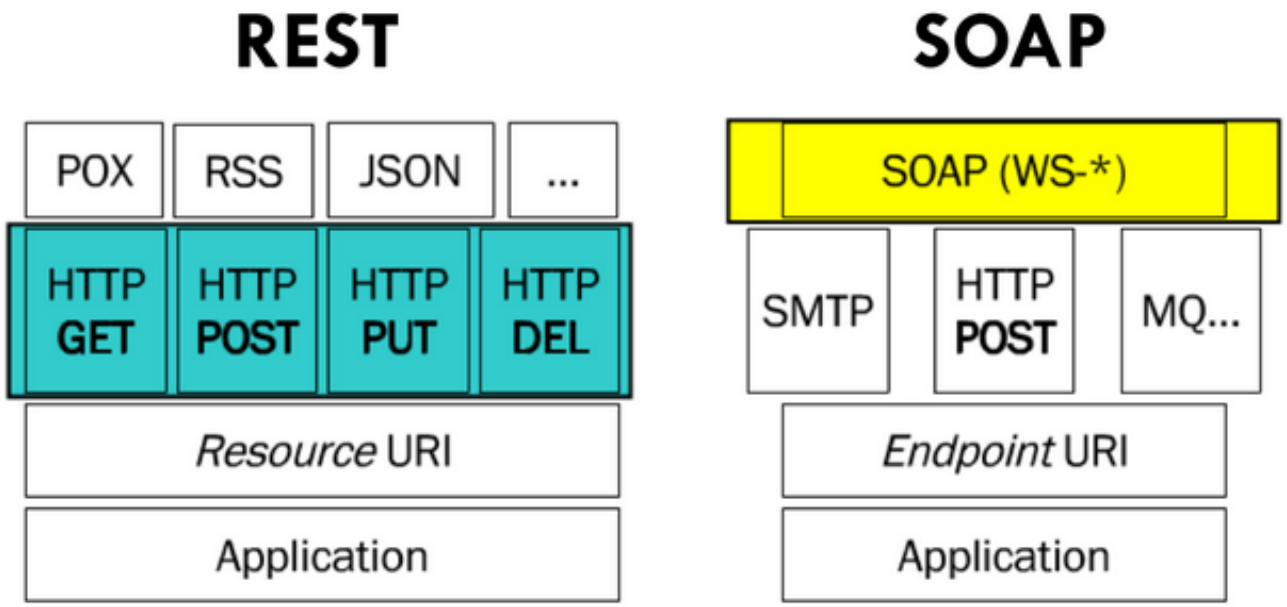
Αρχιτεκτονική REST (3/4)

- Stateless
  - Δεν υπάρχει κατάσταση στη σύνδεση.
  - Οι servers, clients μπορεί να είναι stateful.
  - Κάθε νέα αίτηση πρέπει να φέρει όλη την απαραίτητη πληροφορία για την επεξεργασία της, ανεξάρτητα από προηγούμενη αλληλεπίδραση του ίδιου client με τον server.

Αρχιτεκτονική REST (4/4)

- Cacheable
  - Οι πόροι πρέπει να μπορούν να αποθηκευτούν προσωρινά όπου είναι εφικτό (με χρόνο λήξης).
  - Το πρωτόκολλο πρέπει ρητά να επιτρέπει τη δήλωση ικανότητας προσωρινής αποθήκευσης ή μη καθώς και το χρόνο λήξης.
  - Χρησιμοποιείται HTTP → χρησιμοποιούνται HTTP cache-control headers
  - Οι clients θα πρέπει να ακολουθούν την προδιαγραφή των servers για την προσωρινή αποθήκευση
- Proxy Servers
  - Μπορεί να χρησιμοποιηθούν ως τμήμα της αρχιτεκτονικής για τη βελτίωση της απόδοσης και της κλιμακωσιμότητας.

SOAP vs REST


REST
POX
RSS
JSON
...
HTTP GET
HTTP POST
HTTP PUT
HTTP DEL
Resource URI
Application
SOAP
SOAP (WS-*)
SMTP
HTTP POST
MQ...
Endpoint URI
Application

SOAP vs REST

SOAP	REST
Soap stands for Simple Object Access Protocol.	Rest stands for Representational State Transfer.
SOAP uses a service interface	while REST uses URIs.
SOAP is a protocol	REST is an architectural style
SOAP only uses XML	Rest uses json, XML, PLAIN TEXT, JSON ,HTML
SOAP IS LIKE an envelope,	REST IS LIKE AN POSTCARD.
SOAP requires more server and bandwidth	REST requires fewer server and bandwidth
SOAP supports WS-Security.	REST can use the secure version of the HTTP protocol
SOAP have a Web Service Description Language file	REST does not have
SOAP API exposes the operation.	REST API exposes the data.

Οδηγίες για σχεδιασμό REST 1/3

- Όχι χρήση «πραγματικών» URL, όπως π.χ. ένα XML αρχείο http://www.acme.com/inventory/product003.xml
  - Χρήση «λογικών» URLs: http://www.acme.com/inventory/product/003
- Οι ερωτήσεις δεν θα πρέπει να επιστρέφουν υπερβολικά πολλά δεδομένα.
  - Αν απαιτείται, μπορεί να χρησιμοποιηθεί μηχανισμός σελιδοποίησης, επιστρέφοντας π.χ. τα πρώτα 10 αποτελέσματα με συνδέσμους για τα υπόλοιπα.

Οδηγίες για σχεδιασμό REST 2/3

- Καλή τεκμηρίωση απόκρισης REST, όχι εύκολη αλλαγή
  - Η απόκριση REST μπορεί να είναι οτιδήποτε.
  - Η κύρια κατανάλωση αποκρίσεων REST είναι μηχανές που αν αλλάξει η μορφή και το περιεχόμενο της, δεν θα μπορέσουν να την αναγνωρίσουν
  - Αν η απόκριση είναι XML, να συνοδεύεται από DTD ή schema.

Οδηγίες για σχεδιασμό REST 2/3

- Συμπερίληψη REST URLs σε αποκρίσεις
  - Δεν χρειάζεται να αφεθούν οι clients να κατασκευάζουν μόνοι τους τα URLS.
  - Σε κάθε απόκριση να συμπεριλαμβάνεται το πραγματικό URL για περισσότερες πληροφορίες
    - Μεγαλύτερο μέγεθος απόκρισης
    - Καλύτερη συμπεριφορά clients σε μελλοντικές αλλαγές
- Οι αιτήσεις GET δεν θα πρέπει ποτέ να προκαλούν αλλαγή κατάστασης
  - Χρήση POST για αλλαγή κατάστασης (ή PUT, DELETE, κτλ.)

Διάλεξη 6

JSON

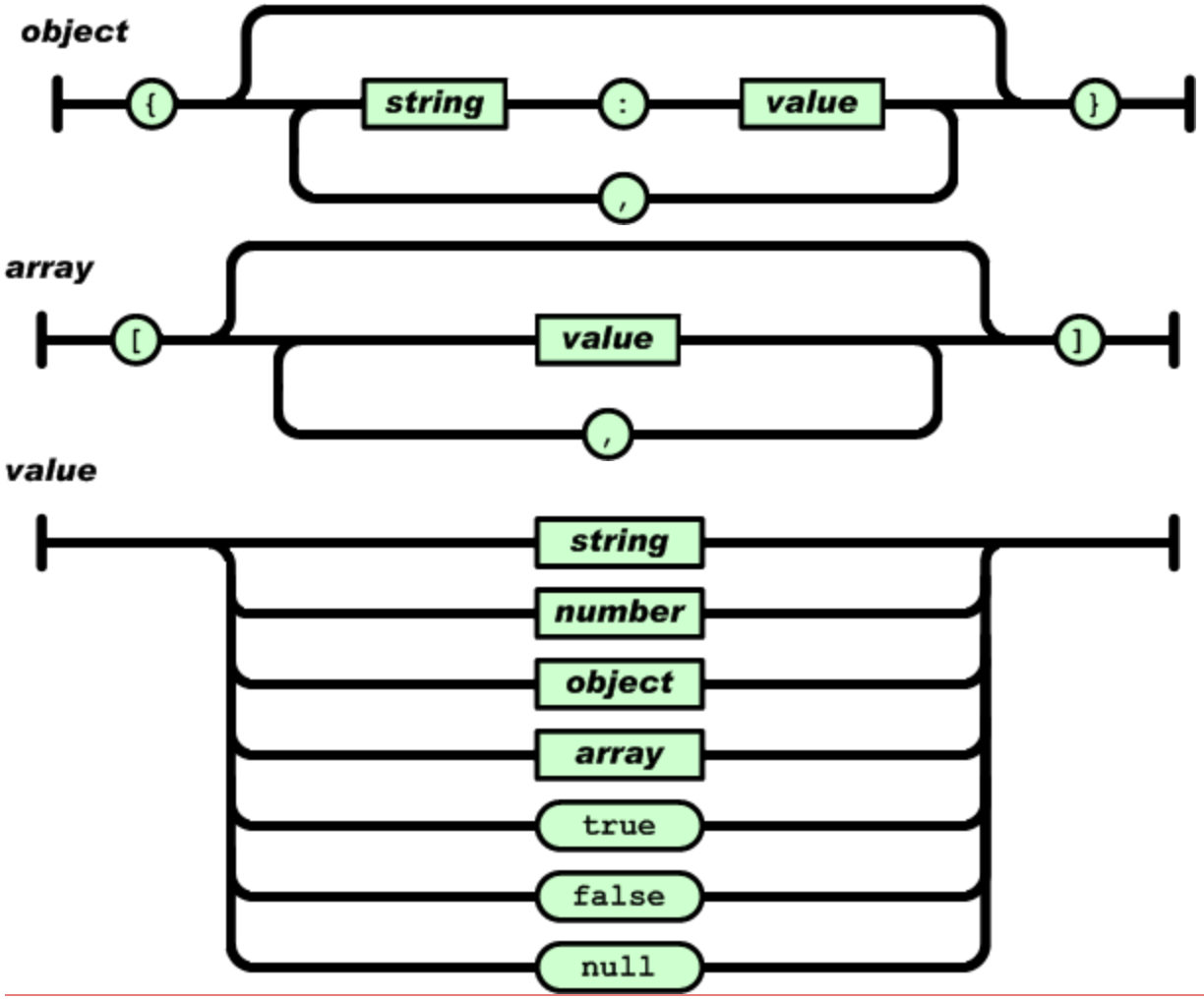
JSON

- JavaScript Object Notation
- Format ανταλλαγής δεδομένων
- Εύκολο για ανθρώπους να γράψουν και να διαβάσουν
- Εύκολο για μηχανές να δημιουργήσουν και να αναλύσουν
- Προέρχεται από τη γλώσσα JavaScript
- Είναι ανεξάρτητο γλώσσας προγραμματισμού
- Από τα δημοφιλέστερα formats ανταλλαγής δεδομένων

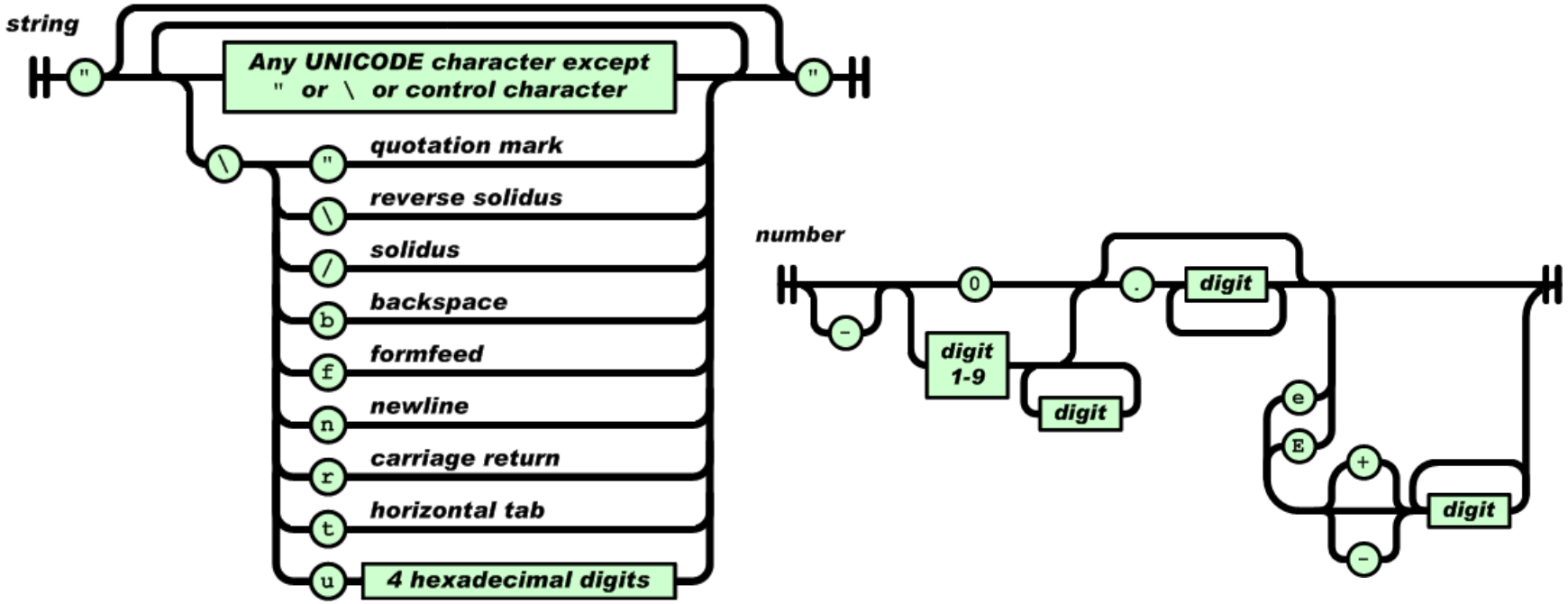
Δομή JSON

- Βασίζεται στις δομές δεδομένων:
  - Συλλογή ζευγών ονόματος-τιμής. Γνωστή και ως object, record, struct, dictionary, hash table, keyed list, associative array.
  - Ταξινομημένη λίστα τιμών. Γνωστή ως array, vector, list, sequence.

Μορφή JSON 1/2



Μορφή JSON 2/2



Παράδειγμα JSON

```
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
],
  "children": [],
  "spouse": null
}
```

RESTful services σε JAVA

- JAX-RS: Java API for RESTful Web Services (JAX-RS), σύνολο από APIs για την ανάπτυξη REST services. Το JAX-RS είναι τμήμα της Java EE6 (Java Specification Request (JSR) 311).
- Jersey: Η πρότυπη υλοποίηση του JAX-RS
  - Παρέχει βιβλιοθήκη για την υλοποίηση RESTful web services σε Java servlet container

Jersey

- Στο μέρος του server, το Jersey παρέχει μια υλοποίηση servlet που σαρώνει προκαθορισμένες κλάσεις για να ταυτοποιήσει πόρους REST. Το servlet αυτό πρέπει να καταχωριστεί στο web.xml της web εφαρμογής.
- Στο μέρος του client παρέχεται και client βιβλιοθήκη για την επικοινωνία με υπηρεσίες REST.
- Το servlet αναλύει την εισερχόμενη HTTP αίτηση και επιλέγει την κατάλληλη κλάση και μέθοδο για να απαντήσει. Η επιλογή βασίζεται σε annotations.
- Μία REST web εφαρμογή αποτελείται από κλάσεις δεδομένων (πόρους) και υπηρεσίες. Οι δύο τύποι συνήθως διατηρούνται σε διαφορετικά packages.
- Το JAX-RS υποστηρίζει τη δημιουργία XML and JSON αρχείων μέσω τηςJava Architecture for XML Binding (JAXB).

JAX-RS Annotations 1/2

Annotation	Description
@PATH(your_path)	Sets the path to base URL + /your_path. The base URL is based on your application name, the servlet and the URL pattern from the web.xml configuration file.
@POST	Indicates that the following method will answer to an HTTP POST request.
@GET	Indicates that the following method will answer to an HTTP GET request.
@PUT	Indicates that the following method will answer to an HTTP PUT request.
@DELETE	Indicates that the following method will answer to an HTTP DELETE request.

JAX-RS Annotations 2/2

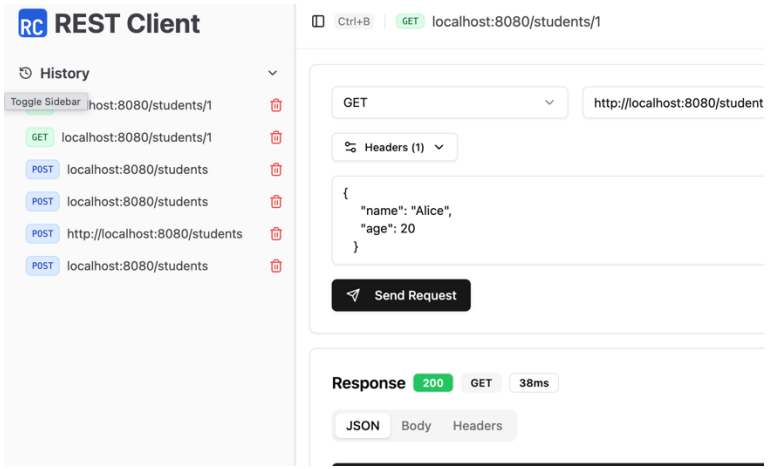
Annotation	Description
@Produces(MediaType.TEXT_PLAIN[, more-types])	@Produces defines which MIME type is delivered by a method annotated with @GET. In the example text ("text/plain") is produced. Other examples would be "application/xml" or "application/json".
@Consumes(type[, more-types])	@Consumes defines which MIME type is consumed by this method.
@PathParam	Used to inject values from the URL into a method parameter. This way you inject, for example, the ID of a resource into the method to get the correct object.

Useful links

https://start.spring.io/

https://www.geeksforgeeks.org/springboot/spring-boot-with-h2-database/

Rest Client – Chrome Web Store



Classes vs Interfaces