

## Un'introduzione ai Web service

Valeria Cardellini  
Università di Roma Tor Vergata

## Definizione di Web service

- Definizione fornita del W3C  
<http://www.w3.org/TR/ws-arch/>

A Web service is a software system designed to support **interoperable machine-to-machine interaction** over a network.

It has an **interface** described in a **machine-processable format** (specifically WSDL).

Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

## Definizione di Web service (2)

- Definizione tratta dal tutorial introduttivo dell'IBM "Web Services - The Web's next revolution"  
<http://www.ibm.com/developerworks/edu/ws-dw-wsbasics-i.html>

Web services are a new breed of Web application. They are self-contained, self-describing, modular applications that can be **published, located, and invoked across the Web**.

Web services perform functions that can be anything from simple requests to complicated business processes.

A sample Web service might provide stock quotes or process credit card transactions.

Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service.

## Caratteristiche dei Web service

- Rappresentano una soluzione per permettere l'interazione e l'interoperabilità **tra applicazioni** in ambito Web
- Si basano sull'idea di fornire un linguaggio ed una piattaforma interoperabile, comune a sistemi differenti
- Sono una combinazione di diversi standard tecnologici, di tipo **aperto** (XML, HTTP, SOAP, ...) che permettono a chiunque di utilizzarli
- Sono accessibili mediante un'interfaccia standard
- Permettono a sistemi eterogenei di lavorare insieme per realizzare il **service oriented computing (SOC)**
  - Programmazione con componenti distribuite sul Web

## Web service

- Componenti software indipendenti dalla piattaforma e dall'implementazione che possono essere:
  - **Descritti** usando un linguaggio di descrizione del servizio
  - **Pubblicati** in un registro di servizi
  - **Scoperti** mediante un meccanismo standard (a runtime o a tempo di progetto)
  - **Invocati** mediante un'API, solitamente tramite la rete
  - **Composti** con altri servizi

## Web service (2)

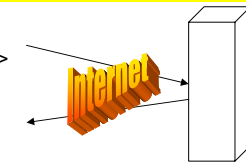
- Componenti software **indipendenti dalla piattaforma e dall'implementazione** che possono essere:
  - **Descritti** usando un linguaggio di descrizione del servizio
  - **Pubblicati** in un registro di servizi
  - **Scoperti** mediante un meccanismo standard (a runtime o a tempo di progetto)
  - **Invocati** mediante un'API, solitamente tramite la rete
  - **Composti** con altri servizi

Un client non può dire quale linguaggio, sistema operativo o tipo di computer è stato usato

Non possono essere inviati o ricevuti dati binari (ma ci sono eccezioni)

```
<name>Character data
</name><cost>123.45</cost>
```

```
<response>Character data
</response>
```



## Web service (3)

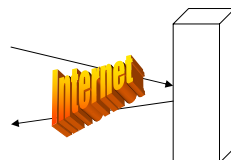
- Componenti software indipendenti dalla piattaforma e dall'implementazione che possono essere:
  - **Descritti** usando un linguaggio di descrizione del servizio
  - **Pubblicati** in un registro di servizi
  - **Scoperti** mediante un meccanismo standard (a runtime o a tempo di progetto)
  - **Invocati** mediante un'API, solitamente tramite la rete
  - **Composti** con altri servizi

Un Web service deve descrivere se stesso: quali tipi di richieste può soddisfare, quali sono gli argomenti, quale è il trasporto

What information do you need?

2 arguments:

(1) Item name (2) Quantity

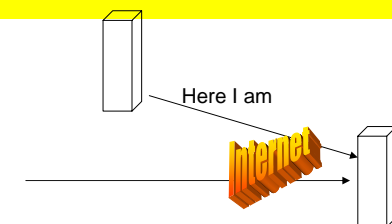


## Web service (4)

- Componenti software indipendenti dalla piattaforma e dall'implementazione che possono essere:
  - **Descritti** usando un linguaggio di descrizione del servizio
  - **Pubblicati** in un registro di servizi
  - **Scoperti** mediante un meccanismo standard (a runtime o a tempo di progetto)
  - **Invocati** mediante un'API, solitamente tramite la rete
  - **Composti** con altri servizi

Un Web service deve indicare ad un registro di servizi dove è localizzato

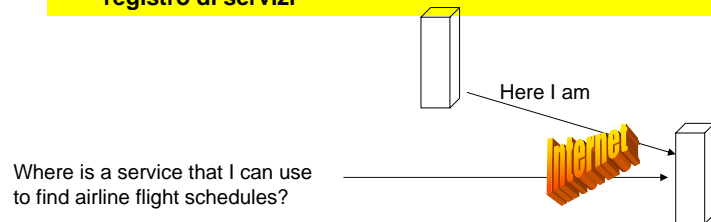
Where is a service that I can use to find airline flight schedules?



## Web service (5)

- Componenti software indipendenti dalla piattaforma e dall'implementazione che possono essere:
  - **Descritti** usando un linguaggio di descrizione del servizio
  - **Pubblicati** in un registro di servizi
  - **Scoperti** mediante un meccanismo standard (a runtime o a tempo di progetto)
  - **Invocati** mediante un'API, solitamente tramite la rete
  - **Composti** con altri servizi

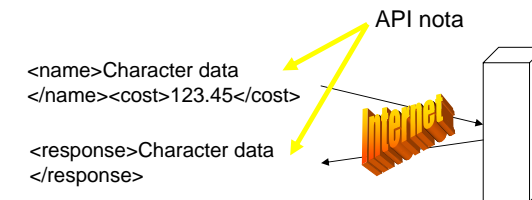
Un potenziale client deve trovare il Web service in un registro di servizi



## Web service (6)

- Componenti software indipendenti dalla piattaforma e dall'implementazione che possono essere:
  - **Descritti** usando un linguaggio di descrizione del servizio
  - **Pubblicati** in un registro di servizi
  - **Scoperti** mediante un meccanismo standard (a runtime o a tempo di progetto)
  - **Invocati** mediante un'API, solitamente tramite la rete
  - **Composti** con altri servizi

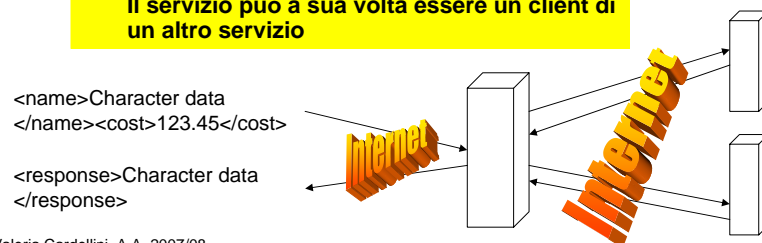
Gli argomenti ed i tipi di dato restituiti devono essere noti



## Web service (7)

- Componenti software indipendenti dalla piattaforma e dall'implementazione che possono essere:
  - **Descritti** usando un linguaggio di descrizione del servizio
  - **Pubblicati** in un registro di servizi
  - **Scoperti** mediante un meccanismo standard (a runtime o a tempo di progetto)
  - **Invocati** mediante un'API, solitamente tramite la rete
  - **Composti con altri servizi**

Il servizio può a sua volta essere un client di un altro servizio



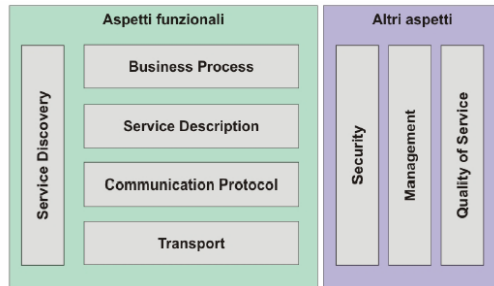
## Service Oriented Architecture

- Architettura di riferimento dei Web service: Service Oriented Architecture (SOA)
  - **Service requestor**: richiede l'esecuzione di un Web service
  - **Service provider**: implementa il servizio e lo rende disponibile sul Web
  - **Service registry**: offre un servizio di pubblicazione e ricerca dei servizi disponibili



## Pila protocollare dei Web service

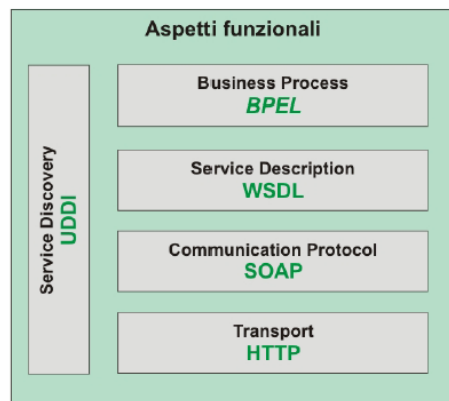
- Aspetti funzionali
  - **Protocollo di trasporto**: invio e ricezione di richieste e risposte tra requestor e provider
  - **Protocollo di comunicazione**: scambio di messaggi basato su XML
  - **Descrizione del servizio**: interfaccia funzionale del servizio
  - **Business process**: composizione dei servizi
  - **Scoperta dei servizi**: definizione dei service registry



## Standard per i Web service

- Il service provider costruisce e definisce il servizio usando WSDL
  - **Web Services Description Language (WSDL)**  
<http://www.w3.org/TR/wsdl.html>  
<http://www.w3.org/TR/wsdl20-primer/>
- Il service provider registra il servizio mediante UDDI
  - **Universal Description Discovery and Integration (UDDI)**  
<http://www.uddi.org/>
- Il service requestor trova il servizio cercando in un registro UDDI
- Il service requestor si collega al Web service fornito dal service requestor ed invoca le sue operazioni mediante SOAP
  - **Simple Object Access Protocol (SOAP)**  
<http://www.w3.org/TR/soap12-part0/>

## Stack tecnologico dei Web service



## XML e Web service

- I Web service si basano sul linguaggio XML perché indipendente da linguaggi, applicazioni e piattaforme specifiche
- XML garantisce
  - Ricchezza espressiva
  - Estendibilità
  - Portabilità
  - Facilità di comprensione
- Gli schemi XML possono essere validati da entrambe le parti che comunicano

## Simple Object Access Protocol

---

- **Simple Object Access Protocol** (SOAP)
- Specifico **protocollo di comunicazione** tra Web service basato su XML per scambiare dati ed invocare metodi su oggetti remoti usando un protocollo applicativo sottostante (HTTP, SMTP, FTP, ...)
  - XML permette di scambiare strutture dati anche complesse nel payload del messaggio SOAP
  - Serializzazione dei dati in XML
- Protocollo leggero, robusto e flessibile
- Indipendente dal sistema operativo e dal linguaggio di programmazione

## Motivazioni per SOAP

---

- Molte applicazioni distribuite comunicano usando chiamate a procedura remota (RPC) tra oggetti distribuiti (ad esempio, DCOM e CORBA)
- Ma HTTP non è progettato per questi oggetti
  - Le chiamate RPC non possono essere facilmente adattate ad Internet
- Inoltre esistono problemi di sicurezza per RPC
  - La maggior parte dei firewall e dei proxy server sono impostati per bloccare questo tipo di traffico
- HTTP è l'unico vero protocollo firewall-friendly
- HTTP è supportato da tutti i Web browser e server

## Obiettivi di SOAP

---

- Aumentare l'interoperabilità rispetto a soluzioni proprietarie
  - Ottenibile grazie all'uso di XML e HTTP
- Permettere una facile manutenibilità ed aggiornamento
  - Il formato del payload in XML può essere esteso facilmente
- Eliminare le limitazioni dovute alle politiche di sicurezza
  - L'uso di HTTP e messaggi testuali permette di utilizzare proxy Web
  - Controllo degli header HTTP da parte di firewall

## Limitazioni di SOAP

---

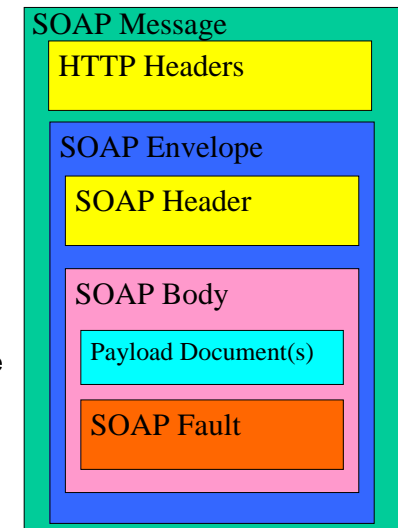
- Non considera le problematiche proprie di un sistema di oggetti distribuiti
- Non è ottimale a livello di prestazioni
  - I dati sono serializzati in XML
  - La deserializzazione richiede di usare un parser XML per estrarre i dati dal payload
- E' un protocollo senza stato
  - Come HTTP, non è in grado di mantenere nativamente informazioni di stato fra una connessione e l'altra
- Non gestisce la sicurezza
  - Per avere sicurezza occorre basare SOAP su HTTPS

## Cosa definisce SOAP

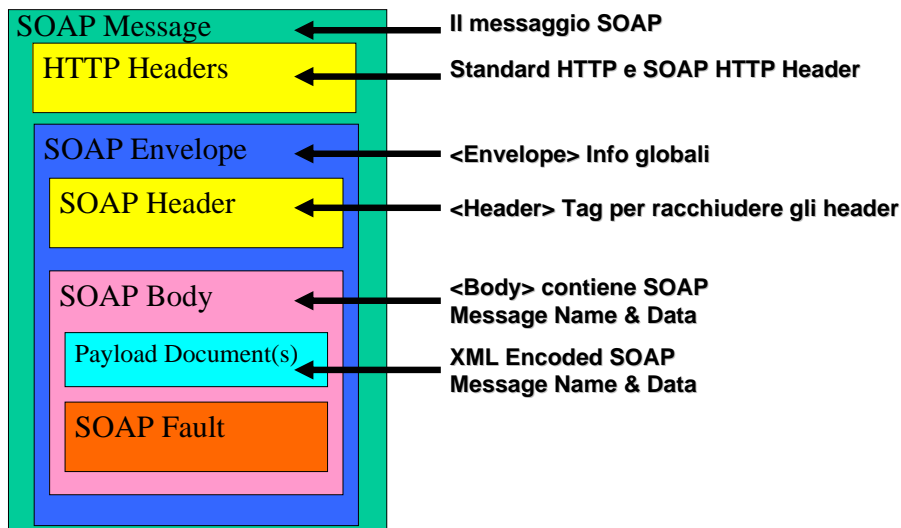
- Specifica della **SOAP envelope** (busta)
  - Definisce il modo di incapsulare i dati da scambiare fra host
  - In caso di errore definisce il formato del messaggio di fault
- Regole di **codifica dei dati**
  - Definisce la codifica con cui sono scambiati i dati (es. numeri float)
  - Vengono utilizzate le definizioni di XML schema
- Convenzioni per definire una **Remote Procedure Call**
  - Definisce come specificare il nome della procedura da chiamare, passare i parametri e ricevere la risposta (valore di ritorno)

## Struttura generale del messaggio SOAP

- SOAP envelope
  - Identifica il documento XML come messaggio SOAP
- SOAP header (opzionale)
  - Contiene informazioni aggiuntive per il processamento del messaggio
- SOAP body
  - Contenuto vero e proprio del messaggio
  - Contiene chiamate o risposte
- SOAP fault
  - Contiene informazioni su eventuali errori occorsi durante il processamento



## Struttura generale del messaggio SOAP (2)



## Struttura generale del messaggio SOAP (3)

```

<SOAP:Envelope xmlns:SOAP=
  "http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Header>
    <!-- Content of header goes here -->
    ...
  </SOAP:Header>
  <SOAP:Body>
    <!-- Content of body goes here -->
    ...
  </SOAP:Body>
</SOAP:Envelope>
    
```

## Richiesta SOAP su HTTP

```
POST /travelservice
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "http://www.acme-travel.com/flightinfo"

<SOAP:Envelope
  xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <m:GetFlightInfo
      xmlns:m="http://www.acme-travel.com/flightinfo"
      SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <airlineName xsi:type="xsd:string">UL
      </airlineName>
      <flightNumber xsi:type="xsd:int">506
      </flightNumber>
    </m:GetFlightInfo>
  </SOAP:Body>
</SOAP:Envelope>
```

Header della richiesta HTTP  
(utilizza il metodo POST)

Envelope della richiesta SOAP

## Risposta SOAP su HTTP

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: xxx

<SOAP:Envelope
  xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP:Body>
    <m:GetFlightInfoResponse
      xmlns:m="http://www.acme-travel.com/flightinfo"
      SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <flightInfo>
        <gate xsi:type="xsd:int">10</gate>
        <status xsi:type="xsd:string">ON TIME</status>
      </flightInfo>
    </m:GetFlightInfoResponse>
  </SOAP:Body>
</SOAP:Envelope>
```

Header della risposta HTTP

Envelope della risposta SOAP  
(contiene il valore di ritorno)

## Descrizione dei servizi

- Una volta che il Web service è attivo, come fanno i service requestor a sapere quale tipo di messaggio inviare per la richiesta?
  - SOAP non basta, definisce soltanto il formato di una envelope
- Per garantire l'interoperabilità fra sistemi eterogenei è necessario un meccanismo che permetta a requestor e provider di capire l'esatta struttura ed il tipo di dati dei messaggi
  - Occorre dire al requestor quale tipo di messaggio XML può inserire nel body del messaggio SOAP
- **Web Services Description Language (WSDL)** è un dialetto XML che consente di descrivere un servizio in modo strutturato
- Un documento WSDL fornisce la descrizione funzionale del servizio, specificando il formato dei messaggi di richiesta e di risposta

## Web Service Description Language

- Un file WSDL è un tipo di documento XML contenente informazioni sul servizio riguardanti
  - La semantica delle interfaccia
  - Dettagli amministrativi per la chiamata ad un Web service
- Quando qualcuno vuole usare un Web service
  - Individua il servizio (ad es. tramite UDDI)
  - Richiede il file WSDL
  - Analizza il file WSDL per determinare
    - La locazione del servizio
    - Le chiamate dei metodi ed i parametri
    - Come accedere ai metodi
  - Crea una richiesta SOAP
  - Invia la richiesta SOAP al servizio
  - Tutto ciò può essere automatizzato

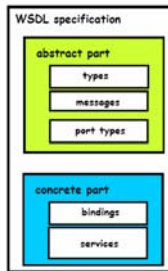
## Cosa descrive WSDL?

- Le operazioni (o metodi) forniti dal servizio
- Dettagli sui formati dei dati e sui protocolli necessari per accedere al servizio
  - XML Schema
- Dettagli sulla locazione del servizio
  - Variano a secondo del protocollo di trasmissione usato
  - Ad es.: URL, indirizzo di e-mail, ...

## Architettura di WSDL

- WSDL descrive i Web service, iniziando con i messaggi che possono essere scambiati tra requestor e provider
- I messaggi sono descritti prima in modo astratto; in seguito vengono aggiunte informazioni pratiche sui protocolli di rete ed i formati dei messaggi
- Un messaggio consiste in una collezione di elementi tipati
- Uno scambio di messaggi è definito una **operation**
- Una collezione di operation è definita un **portType**
- Un service contiene una collezione di **port**
- Ogni port è l'implementazione di un portType e include tutti i dettagli concreti necessari al verificarsi della comunicazione

## Descrizione di un servizio

- Un documento WSDL è formato da 7 elementi, corrispondenti a parti dell'applicazione
    - message
    - operation
    - portType
    - type
    - binding
    - port
    - service
- Descrizione astratta
- Descrizione concreta
- 
- ```
graph TD
    subgraph WSDL_specification [WSDL specification]
        subgraph abstract_part [abstract part]
            types
            messages
            port_types
        end
        subgraph concrete_part [concrete part]
            bindings
            services
        end
    end
```

## WSDL: descrizione astratta

- **type**
  - Per definire i tipi di dato usati all'interno del documento
  - Definito usando XML Schema come type system
- **message**
  - Definizione astratta e tipata dei dati scambiati tra requestor e provider, contenente i parametri di richiesta e di risposta
  - Può essere un messaggio di input, output o fault
- **portType**
  - Combinano i messaggi con lo scopo di definire l'interazione
  - Di solito uno per documento WSDL
  - Corrisponde al servizio stesso
  - Composto da un insieme di elementi operation
- **operation**
  - Specifica i nomi delle operazioni, gli input e output
  - Vengono specificati i messaggi scambiati durante l'operazione



## Esempio: descrizione astratta

```
<message name="GetFlightInfoInput">
  <part name="airlineName" type="xsd:string"/>
  <part name="flightNumber" type="xsd:int"/>
</message>
<message name="GetFlightInfoOutput">
  <part name="flightInfo" type="fixsd:FlightInfoType"/>
</message>
<message name="CheckInInput">
  <part name="body" element="eticketxsd:Ticket"/>
</message>
<portType name="AirportServicePortType">
  <operation name="GetFlightInfo">
    <input message="tns:GetFlightInfoInput"/>
    <output message="tns:GetFlightInfoOutput"/>
  </operation>
  <operation name="CheckIn">
    <input message="tns:CheckInInput"/>
  </operation>
</portType>
```

## WSDL: descrizione concreta

- binding
  - Fornisce i dettagli per l'implementazione delle operazioni contenute in un portType
  - Specifica il protocollo di trasporto e la codifica dei dati (HTTP, SOAP, ...)
- port
  - Specifica l'indirizzo di rete del servizio con cui effettuare la connessione
- service
  - Una collezione di port correlati
  - Permette di raggruppare tutti i portType, in modo che sia immediatamente leggibile e comprensibile per un utente quali sono i port supportati da un determinato servizio
    - Ad esempio, tutti i port associati ad una transazione che richiede più passi

## Esempio: descrizione concreta

```
<binding name="AirportServiceSoapBinding"
  type="tns:AirportServicePortType">
  <soap:binding transport=
    "http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetFlightInfo">
    <soap:operation style="rpc"
      soapAction="http://acme-travel/flightinfo"/>
    <input>
      <soap:body use="encoded"
        namespace="http://acme-travel.com/flightinfo"
        encodingStyle=
          "http://schemas.xmlsoap.org/soap/encoding"/>
    </input>
    <output>
      <soap:body use="encoded"
        namespace="http://acme-travel.com/flightinfo"
        encodingStyle=
          "http://schemas.xmlsoap.org/soap/encoding"/>
    </output>
  </operation>
  <operation name="CheckIn">
    <soap:operation style="document"
      soapAction="http://acme-travel.com/checkin"/>
    <input>
      <soap:body use="literal"/>
    </input>
  </operation>
</binding>
<service name="travelservice">
  <port name="travelservicePort"
    binding="tns:AirportServiceSoapBinding">
    <soap:address location=
      "http://acmetravel.com/travelservice"/>
  </port>
</service>
```

## Universal Description, Discovery, Integration

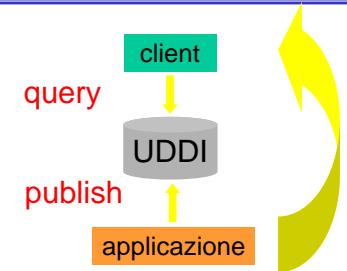
- **Universal Description, Discovery, Integration (UDDI)** è un servizio di directory basato su XML che permette agli utenti dei Web service di localizzarli
- Senza UDDI, due applicazioni possono comunicare solo se già si conoscono, conoscono i servizi offerti e la loro localizzazione
- E' necessario un archivio per permettere ai Web service di rendere pubblica la loro presenza e per renderli raggiungibili dagli utenti

## Panoramica di UDDI

- UDDI è utilizzato da due classi di utenti
  - Provider: chi offre un Web service
  - Requestor: chi ricerca un Web service
- UDDI è simile al Domain Name System
  - Ma il DNS lavora ad un livello più basso, perché risolve indirizzi IP, mentre UDDI lavora a livello più alto perché risolve servizi
- UDDI è un servizio globale condiviso tra server differenti sparsi in tutto il mondo, anche se non organizzati secondo una struttura gerarchica
- I diversi server possono condividere i dati mediante un protocollo di replicazione
- UDDI si basa su SOAP per la trasmissione dei messaggi

## Interazione con UDDI

- Due azioni principali
  - Registrazione
  - Scoperta



- La sicurezza è un aspetto fondamentale
  - Problema: un concorrente potrebbe cancellare il servizio di un altro publisher
  - Soluzione: autenticazione dei publisher
  - Ogni server mantiene traccia dei publisher e di cosa hanno pubblicato
  - Solo chi ha pubblicato un servizio è autorizzato a modificarlo o cancellarlo

## Informazioni in UDDI

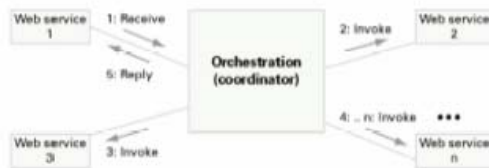
- Divise in tre categorie principali (analogia con elenchi del telefono)
- Pagine bianche
  - Informazioni sui contatti e gli indirizzi dei service provider
- Pagine gialle
  - Informazioni sui diversi servizi disponibili organizzati per categorie di business, per tipo di servizi, ...
- Pagine verdi
  - Informazioni tecniche sul servizio stesso (eventualmente anche il documento WSDL del servizio)

## Composizione di servizi

- Combinazione di più servizi al fine di realizzare attività complesse che coinvolgono diversi partner aziendali
  - Mediante la composizione si crea un nuovo servizio a valore aggiunto
- Due approcci per la composizione dei servizi:  
**coreografia** e **orchestrazione**

## Orchestrazione

- Coordinatore centralizzato (**broker**) che controlla i Web service coinvolti e coordina l'esecuzione delle differenti operazioni
- I singoli servizi non sanno di prendere parte ad un business process a livello di astrazione più elevato
- Solo il coordinatore conosce gli obiettivi della composizione e gestisce l'ordine e la logica delle invocazioni dei servizi, nonché il relativo passaggio dei dati
- Linguaggio BPEL (Business Process Execution Language) come standard de facto

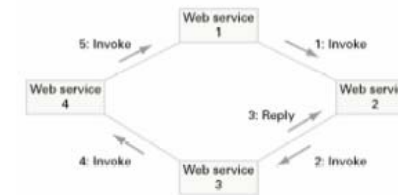


IW - Valeria Cardellini, A.A. 2007/08

41

## Coreografia

- Collaborazione tra entità di pari livello
- Ogni servizio coinvolto nella composizione sa quando eseguire le operazioni e con quali servizi interagire
- Tutti i partecipanti alla coreografia sono consapevoli della logica del business process, delle operazioni da eseguire e dei messaggi da scambiare



IW - Valeria Cardellini, A.A. 2007/08

42

## Esempi di Web service

- Esempi di codice (anche semplici Web service)
  - <http://www.xmethods.com/>
- Amazon Web services
  - <http://aws.amazon.com/>

IW - Valeria Cardellini, A.A. 2007/08

43

## Piattaforme per Web service

- I Web service sono basati su tecnologie "aperte"
  - Garanzia di interoperabilità
- Esistono molti framework per sviluppare applicazioni che utilizzano la tecnologia dei Web service, ad es:
  - IBM ETTK-WS (<http://www.alphaworks.ibm.com/tech/ettkws>)
  - Microsoft .NET
  - Sun Java (<http://java.sun.com/webservices/>)
- Tutti i framework condividono lo stesso insieme di tecnologie
- Apache Axis (<http://ws.apache.org/axis2/>)
  - E' un SOAP engine
  - Implementazione di servizi in Java (Axis2)

IW - Valeria Cardellini, A.A. 2007/08

44