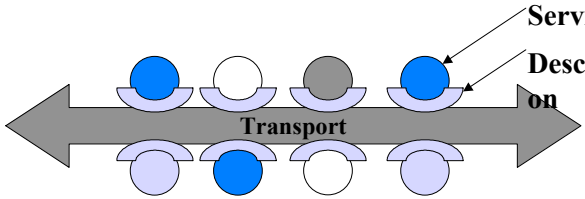
Service Oriented Architectures

bocchi@cs.unibo.it
www.cs.unibo.it/~bocchi

Service Oriented Architecture

- **Una Service Oriented Architecture (SOA) è**
  "...a set of components which can be invoked, and whose interface descriptions can be published and discovered..." [W3C]



- **SOA** è un tipo di **sistema distribuito**
  - agenti = servizi network addressable
  - ciò che importa agli utenti è l'interfaccia
  - formato dei dati e protocolli standard
  - connessione stateless

Service Oriented Architecture (cont.)

- ***Quando utilizzare una SOA***
  - ***Lo sviluppo delle componnti è loosely coupled***
  - ***Le componenti del sistema vengono eseguite su diverse piattaforme***
  - ***Si vuole rendere accessibile un'applicazione attraverso una rete***
  - ***Si vuole rendere accessibile un'applicazione a utenti sconosciuti***
  - ***Si opera su una Internet dove l'affidabilità non può essere garantita***
- ***La SOA può essere istanziata ottenendo diverse architetture***
- ***Le principali applicazioni di SOA riguardano***
  - ***e-business (Web Service Architecture)***

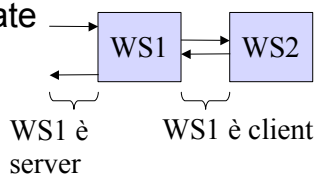
E-business e outsourcing

- Outsourcing: *contract workers from outside of a company to perform specific tasks instead of using company employees.*
- Lo scopo è risparmiare denaro
- L'infrastruttura IT di un'azienda può coinvolgere
  - Reti esterne
  - Risorse esterne
  - Servizi esterni
- Sfruttare tecnologie già esistenti e diffuse

http://ist-socrates.berkeley.edu/~fmb/articles/outsourcingtrends.ht

Scenario…

- Decentramento IT ← outsourcing
- Diffusione Internet
- Permettere interoperabilità tra diverse piattaforme: la piattaforma è Internet
- Internet per le macchine (principalmente b2b)
- → Modello client-server stateless tra applicazioni con interazioni sincrone o asincrone non correlate
- Client-Server → P2P



Sistemi distribuiti

- Agenti software discreti (entità computazionali) che collaborano per implementare qualche funzionalità
  - operano in ambienti differenti
  - comunicano attraverso stack di protocolli intrinsecamente meno affidabili rispetto p.es. alla memoria condivisa
    - latenza non predicibile nell'accesso remoto 
    - problemi di concorrenza
    - partial failure (p.es.)
      - perdita di messaggi: $x(y) \rightarrow 0$
      - crash di un nodo: $[P]^S \rightarrow [*]^S$ e $[*]^S \rightarrow [S]^S$

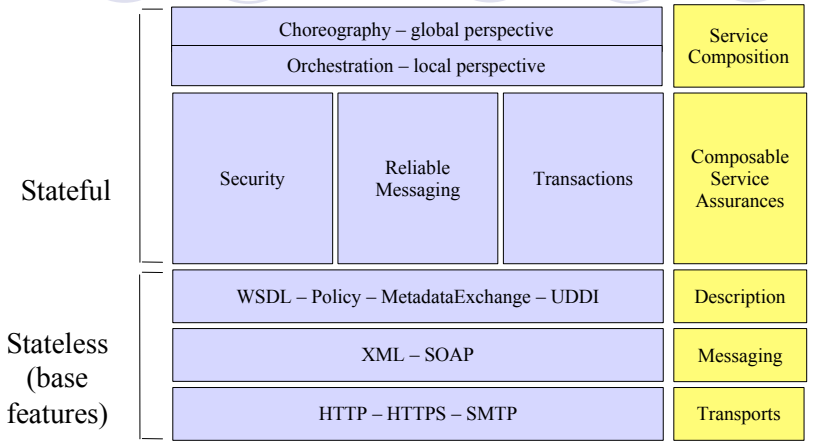
Web Service

- La Web Service Architecture è un'istanza di SOA basata su un particolare stack di protocolli
- Realizzazione del concetto di *dynamic e-business*

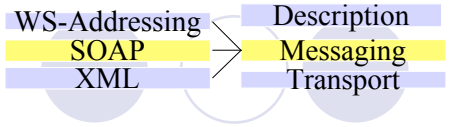
D. Ferguson, "IBM Web Services: Technical and Product Architecture Roadmap,"
IBM Corporation (2001)
http://www-4.ibm.com/software/solutions/webservices/pdf/roadmap.pdf

- Una o più operazioni accessibili tramite URL
- Descritti con linguaggio standardizzato
- Componibili in Web Service più complessi

Web Service: Protocol Architecture

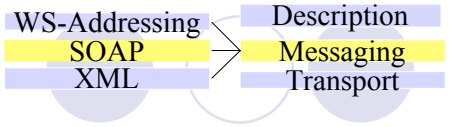


SOAP

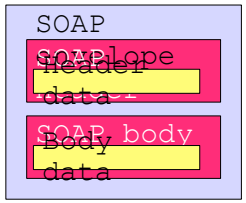


- Problema: comunicazione tra componenti remote
- SOAP
  - comunicazione tra programmi attraverso Internet
    - ~ Remote Procedure Calls (RPC) attraverso HTTP (o SMTP…)
    - HTTP è il protocollo più utilizzato per scambiare informazioni su Internet
  - Permette la comunicazione tra differenti OS, linguaggi, tecnologie

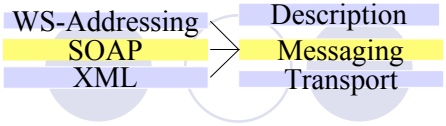
SOAP – cont.



- Un messaggio è un documento XML che contiene:
  - Envelope: identifica il documento XML come un messaggio SOAP
  - Header (opzionale): informazioni su come elaborare il documento
  - Body (necessario): contiene il messaggio vero e proprio
  - Fault (opzionale): contiene informazioni sugli eventuali errori riscontrati durante la computazione

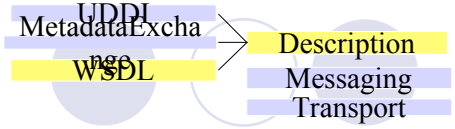


SOAP – cont.



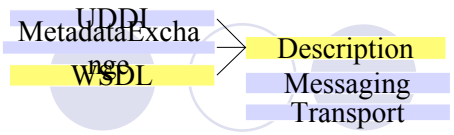

```
<?xml version="1.0"?>
<soap:Envelope
   xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
   soap:encodingStyle="http://www.w3.org/2001/12/soapencoding">
<soap:Header>
   <m:Trans
   xmlns:m="http://www.add.com/trans/”soap:mustUnderstand="1">
    234
   </m:Trans>
</soap:Header>
<soap:Body>
   <m:GetPrice xmlns:m="http://add.com/prices">
   <m:Item>Apples</m:Item>
   </m:GetPrice>
   <soap:Fault> … </soap:Fault>
</soap:Body>
</soap:Envelope>
```

WSDL



- Web Service Description Language
- Documento XML
- Cosa descrive:
  - L’interfaccia di un Web Service come insieme di possibili operazioni
    - Un insieme di: port type (WSDL 1), interface (WSDL 2.0),
    - Un insieme di binding per ogni port type/interface,
    - Un insieme di: porte (WSDL 1), endpoint (WSDL 2.0) per ogni binding
- Cosa non descrive:
  - Informazioni sul comportamento
    - Semantica
    - Ordine delle operazioni
- Riassumendo:
  - Web Service come insieme di endpoint che scambiano messaggi
  - Connessioni **stateless** dove tutti i dati per una richiesta devono essere nella richiesta stessa

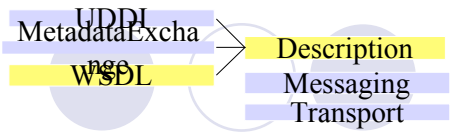
WSDL – cont.



Gli elementi di base per descrivere un WS:

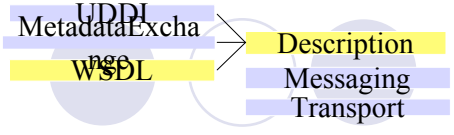
- <types> definisce i tipi di dato usati per descrivere i messaggi scambiati
- <message> definisce i messaggi usati dal Web Service
  - definizione astratta dei dati trasmessi (data element delle operazioni)
  - messaggio composto da una o più parti logiche ciascuna associata ad una definizione in un qualche type system
  - ```
    <message name="termValues">
       <part name="term" type="xs:string"/>
       <part name="value" type="xs:string"/>
    </message>
    ```

WSDL – portType



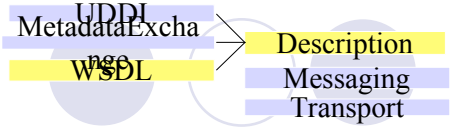
- <portType> definizione astratta di un insieme di operazioni
  - Ogni portType è descritto come un **insieme** di possibili **operazioni** (ed i messaggi coinvolti in tali operazioni)
  - Le azioni possono essere:
    - One-Way: ricezione di un messaggio
    - Request-Response: ricezione di un messaggio seguita dall'invio di un messaggio correlato
    - Solicit-Response: invio di un messaggio e attesa di un messaggio correlato
    - Notification: invio di un messaggio

WSDL – cont.



- <binding> descrive i protocolli e il formato dei dati per le operazioni e i messaggi definiti di un portType particolare
- <port> implementazione di una portType che identifica dove effettivamente è localizzata l'implementazione del servizio (indirizzo del binding)
- <service> aggrega insiemi di porte correlate

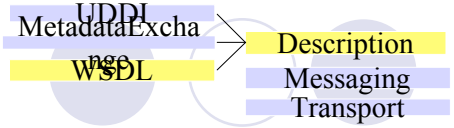
WSDL – cont.




```
<definitions …>
   <wsdl:message name="sayHello_INPUT">
    <part name="nome" type="xsd:string" />
   </wsdl:message>
   <wsdl:message name="sayHello_OUTPUT">
    <part name="ciao" type="xsd:string" />
   </wsdl:message>

   <wsdl:portType name="HelloWorldInterface">
    <wsdl:operation name="sayHello">
        <wsdl:input message="tns:sayHello_INPUT" />
        <wsdl:output message="tns:sayHello_OUTPUT" />
    </wsdl:operation>
   </wsdl:portType>
</definitions>
```
```

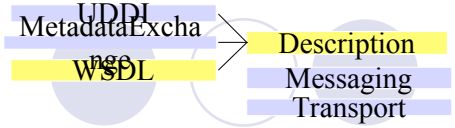
WSDL – cont.




```
<wsdl:binding name="HelloWorldBinding"
              type="tns:HelloWorldInterface">
    <soap:binding style="rpc"
          transport=http://schemas.xmlsoap.org/soap/http/>

      <wsdl:operation name="sayHello">
          <soap:operation soapaction="urn:Hello" />
              <wsdl:input>
                  <soap:body use="encoded"
                             namespace="_" encodingStyle="_" />
              </wsdl:input>
              <wsdl:output>
                  <soap:body use="encoded"
                             namespace="_" encodingStyle="_" />
              </wsdl:output>
     </wsdl:operation>
</wsdl:binding>
```

WSDL – cont.




```
<wsdl:service name="HelloWorldService">


    <wsdl:port name="HelloWorldPort"
          binding="tns:HelloWorldBinding">
            <soap:address location="http://localhost:8080" />
     </wsdl:port>

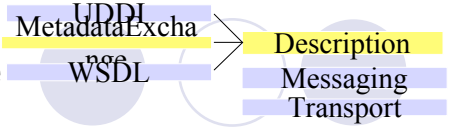
    <wsdl:port name="HelloWorldPort_Java"
          binding="tns:HelloWorldBinding">
            <soap:address
                location="http://localhost/soap/servlet/rpcrouter" />
     </wsdl:port>



</wsdl:service>
```

Più istanze dello stesso server

WS-MetadataExchange

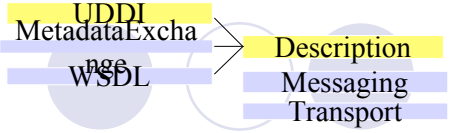


- Dato il riferimento ad un Web Service

  ⇩

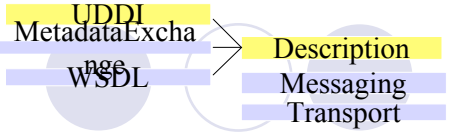
  voglio capire cosa fa
- L'interfaccia di un Web Service contiene operazioni che permettono ad altri servizi di accedere ai suoi metadati

UDDI

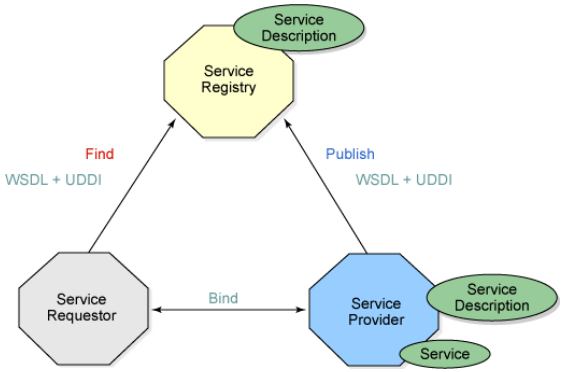


- Scoprire Web Services
  - Tramite direct publish: il Service Provider invia il Service Description direttamente al Service Requestor attraverso meccanismi diretti (e-mail, CD-ROM,…)
  - Tramite dynamic publish: il Service Requestor recupera il Service Description attraverso un URL conosciuto
  - Tramite service registry: si interroga un database UDDI che fornisce il Service Description più idoneo

UDDI

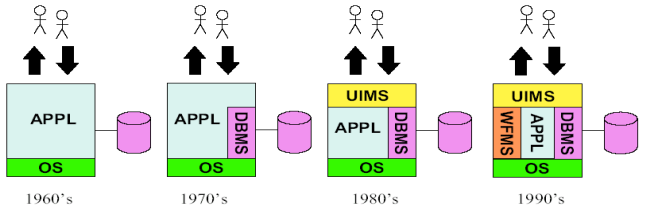


- UDDI specification definisce un servizio di raccolta di metadati



Oltre WSDL: la composizione di WS

- WSDL descrive le operazioni…ma cosa c'è dietro?
  stateless service, subservient object, agente-instanza con proprio ciclo di vita...
- In molti contesti business-to-business è cruciale
  - descrivere le possibili sequenze di operazioni e così il loro ordine
  - supportare interazioni stateful e long-running tra Web service
- La presenza di stato si può gestire
  - a livello di application code
  - in modo ortogonale (Gelenter: applicazione = computazione + coordinazione)



WS e presenza di stato

- La presenza di stato può essere introdotta a due livelli
  - Nei messaggi (come i cookie nella Web Architecture)
  - Nel business process

Lo stato nei messaggi	Lo stato nel business processe
Service assurance	Orchestration - Choreography

Composable Service Assurances

Composition
Assurances

- Security:
  - WS-Security
    - Solitamente affidata al livello di trasporto (HTTPS)
    - Funzionalità più complesse p.es. encrypted security token A→B→C
  - WS-Trust
    - STS (Security Token Service) è un WS che scambia e valida Securty Token
    - Meccanismo per accordarsi sui server fidati
  - WS-SecureConversation
    - HTTPS usa chiavi pubbliche per stabilire chiavi specifiche per la conversazione
    - WS-Security per iniziare una sessione o "conversazione"
    - WS-SecureConversation per accordarsi sulle chiavi specifiche per la sessione
  - WS-Federation
    - Più organizzazioni con un dominio di sicurezza
    - Definire proprietà comuni
- Reliable Messaging
  - WS-ReliableMessaging (NB le comunicazioni sono inaffidabili→numerazione msg…)
- Transactions
  - WS-Coordination
  - WS-AtomicTransaction
  - WS-BusinessActivity
  - BTP

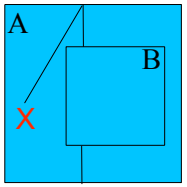
Transazioni

- Più messaggi scambiati tra i partecipanti costituiscono un unico “task” logico
- Le parti devono
  - Iniziare un task coordinato
  - Associare le operazioni con i loro task logici
  - Accordarsi sull’outcome della computazione

Composition
Assurances

Transazioni ACID

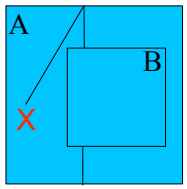
- Transazioni ACID: le proprietà
  - Atomicity: tutto o niente
  - Consistency: il sistema passa ad un altro stato consistente
  - Isolation: due transazioni hanno lo stesso effetto se eseguite in serie o in parallelo
  - Durability: una volta stabilito un outcome, la decisione è immutabile
- Le proprietà vengono garantite bloccando risorse



Composition
Assurances

Transazioni Long Running

- Loosely coupled environments:
  - Interazioni long running
  - Possiamo non avere il controllo delle risorse coinvolte
- Long running transaction e compensazione
- Atomicity e isolation vengono rilassate

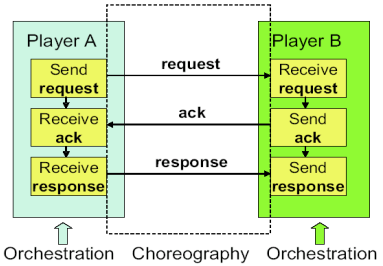


Composition
Assurances

WS-Coordination

- Meccanismo generale per iniziare ed accordarsi sull’outcome di un Web Service task con più participanti
- Esiste un servizio detto coordinator service che permette di iniziare, terminare, associarsi ad un determinato task…
- In tutti i messaggi è presente un coordination context
- 2 principali estensioni
  - WS-AtomicTransaction
  - WS-BusinessActivity
- Altre proposte
  - Business Trensaction Protocol

Orchestrazione e Coreografia



- L'orchestrazione caratterizza business process che possono essere eseguiti presso un particolare servizio e che rappresentano la prospettiva di una parte
- La coreografia descrive il pattern di interazioni tra più business process distribuiti

Composition Assurances

Orchestrazione/Coreografia - Riferimenti

- BPEL4WS

  Business Process Execution Language for Web Services
  http://www-106.ibm.com/developerworks/webservices/library/ws-l

- WS-Choreography Description Language 1.0

  http://www.w3.org/TR/2004/WD-ws-cdl-10-20041012/

Composition Assurances

BPEL4WS

- Da dove proviene:
  - Forti radici nei tradizionali flow models
  - Si basa su WSDL
  - Unisce WSFL e XLANG
- Definisce sia il comportamento astratto che eseguibile dei processi (aspetti implementativi vengono nascosti con il non determinismo)
  - Abstract processes per specifiche di e-commerce
  - Executable processes forniscono un modello per integrare applicazioni
- Business Processes e composizione: tre aspetti
  - Struttura
  - Informazione
  - Comportamento

Composition Assurances

Struttura di un processo BPEL

```
<process …>

   <partners> … </partners>
    <!-- web services the process interacts with -->

   <containers> … </containers>
    <!-- data used by the process -->

   <correlationSets> … </correlationSets>
    <!– used to support asynchronous interactions -->

   <faultHandlers> … </faultHandlers>
    <!– alternate execution path to deal with faulty conditions -->

   <compensationHandlers> … </compensationHandlers>
    <!- code to execute when undoing an action -->

   (activities)*
    <!– what the process actually does -->

</process>
```

Composition Assurances

BPEL4WS - struttura

- Struttura
  - WSDL
  - Si stabiliscono dei ruoli per I partecipanti: associazione con nome tra il servizio composto e un partecipante

```
<partner name="..." serviceLinkType="..."
         partnerRole="..." myRole="..."/>

<serviceLinkType name="...">
  <role name="...">
    <portType name="..."/>*
  </role>
  <role name="...">
    <portType name="..."/>*
  </role>
</serviceLinkType>
```

Composition Assurances

BPEL4WS

- Informazione
  - Il web service composto definisce un insieme di variabili, lo stato del servizio dipende dal valore delle variabili
- Comportamento p.es.

```
<sequence>
<!– execute activities sequentially-->
<flow>
<!– execute activities in parallel-->
<while>
<!– iterate execution of activities until condition is violated-->
```

Composition Assurances

Creare istanze

- Creare istanze di un servizio che ha il proprio ciclo di vita e il proprio stato.
- Possiamo definire delle *start activities* in un Business Process che creano una nuova istanza:

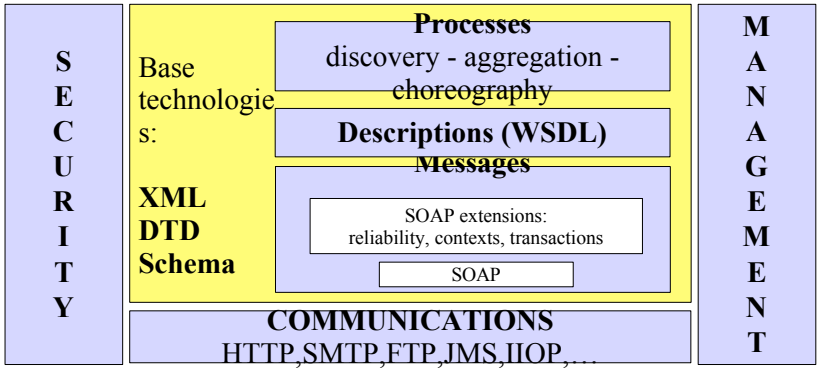
```
<action operation="actionName" createinstance="yes">
            …
</action>
```

Composition Assurances

Correlation set

- Quando comunichiamo con un Business Process dobbiamo comunicare non solo con un dato port number ma anche con la corretta istanza del servizio
- Specifcare **gruppi correlati** di operazioni in una istanza.
- I correlation group sono definiti con correlation set di token condivisi da tutte le operazioni nel gruppo

Web Service – il quadro completo



Grid

- Lo scopo: fornire accesso ad una rete di risorse (memoria, calcolo, rete) distribuite ed eterogenee come se fosse un'unica risorsa globale
- 2002 il Global Grid Forum propone la Open Grid Service Architecture
- OGSA è un'istanza di SOA basata sulle tecnologie dei Web Service
- Convergenza tra Grid e Web Service
- OGSA è basata su Web Service Resource Framework (WS-RF)
- WS-RF estende le funzionalità di base dei Web service
- Le funzionalità aggiunte sono incorporate nel concetto di Grid Service

Grid Service

- Un Grid Service è un Web service che presenta caratteristiche aggiunte
- Interfaces
  - Discovery (risposta a domande sullo stato interno)
  - Dynamic service creation (service Factory → Transient Web Service)
  - Lifetime management
  - Notification
- Conventions:
  - Address naming
  - Upgradeability
- Future:
  - Authorization
  - Concurrency
- External to core Grid
  - Authentication
  - Reliable invocation → transaction protocols

Limiti e speranze…

- Semantica dei servizi: per ora l'accordo sulla semantica dei servizi spetta l'uomo.

  Web Services Architecture: http://www.w3.org/TR/ws-arch
- OWL-S
- Automatizzare in parte questo processo permetterebbe
  - Composizione dinamica
  - Controllo di proprietà (i.e. assenza di deadlock, liveness…)
- Alcune possibili aree di ricerca:
  - Semantic Web
  - Behavioral Types

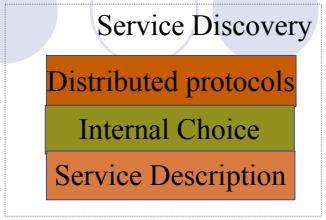
    "*Description based on behavioural types can form the basis of a new kind of discovery mechanism, specifically one based on partial behavioural descriptions*".

L. Bocchi, P. Ciancarini, R. Moretti, V. Presutti, and D. Rossi.
An OWL-S Based Approach to Express Grid Services Coordination.
To appear in SAC 2005

L.G. Meredith and S. Bjorg, *Contracts and Types.*
Communication of the ACM, October 2003/Vol. 46. No. 10

Limiti e speranze…

- Estensioni ad OWL-S
  - Grid
  - Transazioni
- Mapping da BPEL ad OWL-S
- Algoritmi di matchmaking
- Transazioni (e protocolli di negoziazione)
  - Descrizione
  - implementazione



Riferimenti

- Outsourcing
  http://ist-socrates.berkeley.edu/~fmb/articles/outsourcingtrends.html
- D. Ferguson, “IBM Web Services: Technical and Product Architecture Roadmap,” IBM Corporation (2001)
  http://www-4.ibm.com/software/solutions/webservices/pdf/roadmap.pdf
- WSDL
  http://www.w3.org/2002/ws/desc/
- Reliable message delivery in a Web services world:A proposed architecture and roadmap
  (A joint white paper from IBM Corporation and Microsoft Corporation)
  http://www-106.ibm.com/developerworks/webservices/library/ws-rmdev/
- BPEL4WS
  http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/
- WS-Choreography
  http://www.w3.org/TR/ws-chor-reqs/
- UDDI
  http://www.uddi.org

Riferimenti – cont.

- Collaxa 2.0 release candidate 1 (October 17th 2003)
  http://www.collaxa.com/developer.welcome.html
- BP Wizard 1.0 (by Eclipse)
  http://www.bpwizard.com/products/#top
- H.Foster, S. Uchitel, J.Magee and J.Kramer, **Model based verification of Web Service Composition**
- Web Services Architecture
  http://www.w3.org/TR/ws-arch
- L.G. Meredith and S. Bjorg, **Contracts and Types**. Communication of the ACM, October 2003/Vol. 46. No. 10
- OGSA http://www.globus.org/ogsa/
- OGSI http://www.gridforum.org/ogsi-wg/
- DAML-S http://www.daml.org/services/daml-s/0.9/