Sistemi Distribuiti
Corso di Laurea in Ingegneria

Prof. Paolo Nesi

Parte: 2a – Web Services & REST

Department of Systems and Informatics

University of Florence

Via S. Marta 3, 50139, Firenze, Italy

tel: +39-055-4796523, fax: +39-055-4796363

Lab: DISIT, Sistemi Distribuiti e Tecnologie Internet

http://www.disit.dsi.unifi.it/

nesi@dsi.unifi.it paolo.nesi@unifi.it

http://www.dsi.unifi.it/~nesi, http://www.axmedis.org

Web Services and REST

- WS aims
- Architecture e Protocolli
- Gli standard dei WS
- SOAP Call
- Building SOAP-based Applications
- Architetture basate su WS
- Architettura REST

Aims of Web Services

- To realize RPC with standard interfaces
- To allow discovering of possible remote services
- Supposing not having a status or marginal status, loosely coupled applications
- Possibly changing interface to call
- Internet scale communications
- Substitution of CORBA and RMI for direct method invocation, **not for the distributed objects.**
- XML messaging over HTTP → SOAP

Web Service Architecture

- **Major roles** involved in WS architecture
  - Service Provider
  - Service Registry/Broker
  - Service Consumer/User
- **Major operations** of web services
  - Publishing – making a service available
  - Finding/Discovering – locating web services
  - Binding – using web services

Gli standard alla base dei WS

- **UDDI:** Universal Description Discovery and integration, a platform independent open framework per la descrizione di servizi, descrizione di business e integrazione di servizi, Proposto da IBM e Microsoft, www.uddi.org
- **WSDL:** Web Service Description Language, developed by IBM, un XML schema per descrivere metodi e parametri di un servizio WEB
- **SOAP** (1999), Simple Object Access Protocol, proposed by W3C (world Wide Web Consortium). SOAP use XML on HTTP to send requests and receive responses.
  - Nato da Microsoft e Userland come RPC
  - E' un protocollo RPC standardizzato da W3C
  - E' un protocollo XML based, basato su HTTP

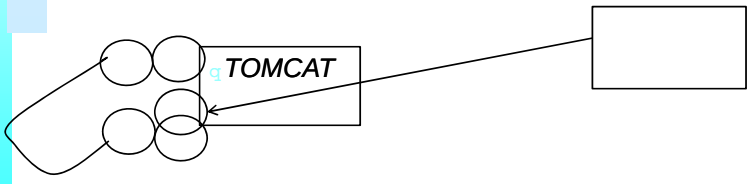
Web Services Protocols

Accessibility of WS

- To make it discoverable
  - Publishing a service in a registry (UDDI)
  - Publishing the service description of its interface (methods & arguments) in some manner
- To make it accessible by using a WSDL formalization of the Web Service.
  - IDL description
  - Platform independent formal description
  - Extensible language

Rendere un servizio accessibile

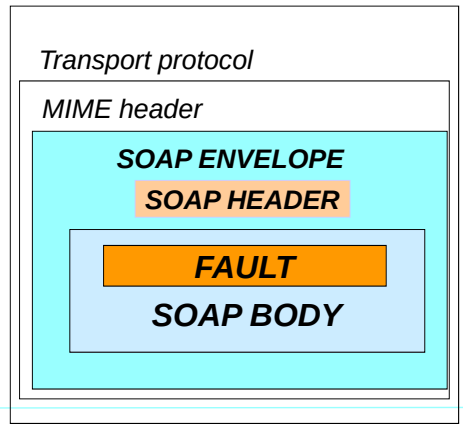
- Una volta che una descrizione del servizio e' stata pubblicata, ci deve essere un Web Server che la metta a disposizione
- Il WEB server puo' mettere a disposizione il servizio attraverso una applicazione custom, per esempio una Servlet, WAR (jsp based) deployed su un Web Application Server come Tomcat.
- Le funzionalita' dei WS vengono messe a disposizione
  - Le Web App possono essere:
    - Deployed, started, stopped, etc.



SOAP and WS

- Il corpo del messaggio e' definito tramite WSDL
  - La grammatica del messaggio si basa su XML di W3C
- il contenitore del messaggio e' SOAP

- *The body element contains the core of the SOAP document – this will contain either the RPC call or the XML message itself*
- *The fault information may contain any exception information*



SOAP message

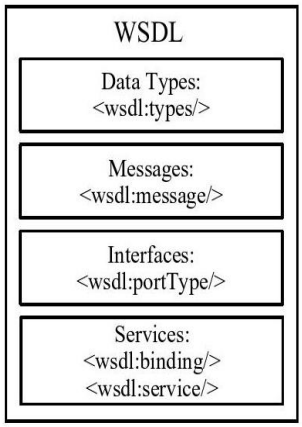
```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope xmlns:SOAP_ENV="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:xsi="http://www.w3c.org/1999/XMLSchema-instance"
        xmlns:xsd="http://www.w3c.org/1999/XMLSchema">
        <SOAP-ENV:Header>

        </SOAP-ENV:Header>

        <SOAP_ENV:Body>


        </SOAP-ENV:Body>


</SOAP-ENV:Envelope>
```

SOAP as RPC on HTTP

- SOAP RPC message contains XML that represents a call or a response
- SOAP XML is converted into a call on the server
  - The response is encoded into SOAP XML to be returned to the client
- SOAP calls pass through firewalls since it is based on HTTP level protocol
- SOAP protocol supports different bindings of WSDL
- SOAP errors are handled using a specialised envelope known as a Fault Envelope
  - a SOAP Fault is a element which has to appear as an immediate child of the body element
  - <faultcode> and <faultstring> are required.


WSDL: Web Service Description Language

- An extended IDL
- Definition of standard interfaces
- Permitting advertising and publication of WS
- Permitting the definition of SLA, Service Level Agreements
- Publication of formal interface call on repositories, UDDI registry
- Requests on UDDI registry




Realizzazione di una applicazione

- Nelle prossime due slide è mostrato come è possibile realizzare un'applicazione C++ composta da server e client che comunicano tramite WS via SOAP
- Questa soluzione è piu' statica di quando descritto in precedenza, cioe' prevede la presenza di un WSDL per produrre il Client compilato
- Si parte dal Server
  - Si decide di pubblicare una certa funzione F(int a, string b)
- Per arrivare ad avere tale funzione come RPC sul client

GSOAP server side

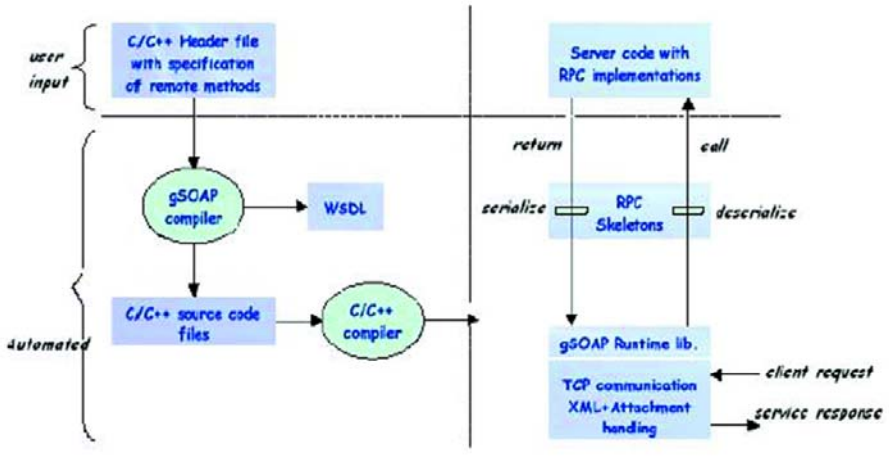


Fig.3- gSoap server-side

GSOAP client side

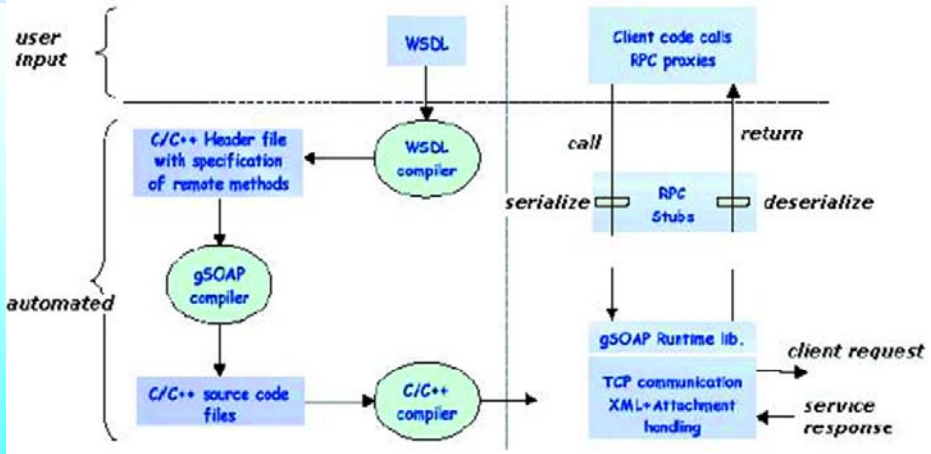


Fig.4- gSoap client-side

Architetture WS, SOA

- **SOA: Service Oriented Architecture**
  - Soluzioni distribuite che mettono a disposizione servzi attraverso Web Application che espongono vari WS
- **Specificita'**
  - WS possono chiamare altri WS
  - Si vengono a creare dipendenze evidenti e nascoste fra WS
  - Applicazioni PHP, C++, Java possono chiamare WS
  - Si possono aprire connessioni dirette e passarci dentro SOAP
- **I WS possono** essere implementazioni di procedure con stato o senza stato
  - Lo stato deve essere tipicamente memorizzato all'interno di una database, file o che altro di persistente
  - E' meglio non avere stato sul protocollo ma di fatto alcuni WS lo hanno. Lo stato di protocollo significa avere svariate richieste collegate fra loro che implicano una seguenza di stati non espliciti nella chiamata
  - Si auspicano transazioni atomiche

Esempi di WS

- Tipicamente senza stato:
  - Dammi il tempo a Firenze Domani
  - Dammi I valori della borsa Milano riguardo a XX, YY
  - Dammi l''IP fisico e il GPS di http://WWW.pippo.it
  - Puoi consegnarmi questo tipo di lavoro <descrizione> entro XX
  - Dammi il mio profilo sono XX con password ZZZ
- Tipicamente con stato persistente ma non di protocollo:
  - Ho visto questo film
  - Ecco il mio profilo
  - Mi autorizzi a usare questo contenuto ?
  - Sono passato da questo punto
  - Ho rilevato questa temperatura alle coordinate GPS X,Y
  - Accetto la tua offerta <identificativo> per YY Euro


Esempi di WS non corretti

- Sono tipicamente sequenze, per esempio:
  - Call1: Sono Paolo Nesi
  - Call2: Dammi lo stato del mio conto
  - Call3: fai questa operazione
- Protocolli complessi portano a problemi di sicurezza
  - E' possibile proteggere la comunicazione tramite soluzioni come SSL, HTTPS, etc.
  - Lo stato implicito sul procollo (come nell'esempio sopra) puo' portare a dover tenere conto di tempi di attesa per mantenere lo stato, etc.


REST

- Representational State Transfer (REST)
  - an approach for getting information content from a Web site by reading a designated (ok, you need a URI) Web page that contains an XML file that describes and includes the desired content.
  - To use HTTP (get,put,…) to a URI, with XML as the payload …
- REST is an architecture strategy and not a protocol
- REST+XML is appropriate for all of the applications that SOAP/WSDL are designed/used for.
- Suitable for exploiting/encapsulating legacy


REST design

- Components in a REST system obey the following constraints:
  - Servers are stateless and messages can be interpreted without examining history.
  - resources are identified through URIs
  - resources manipulated through representations
  - Uniform Interface for all resources:
    - GET (Query the state, idempotent, can be cached)
    - POST (Modify, transfer the state)
    - PUT (Create a resource)
    - DELETE (Delete a resource)
  - self-descriptive messages


REST design

- REST emphasizes the role of intermediaries: caches, proxies, gateways, etc.
  - The solution has to work in all these conditions.
- Main correspondences

	SQL	REST
CREATE	Insert	PUT
READ	Select	GET
UPDATE	Update	POST
DELETE	Delete/Drop	DELETE

Confronto: REST vs WS

- REST:
  - Maggiore dipendenza fra server e client
  - Minor costo di realizzazione per cose semplici
  - Minor espressività del modello di chiamata
  - Problemi con dati strutturati
- WS:
  - Maggior costo di realizzazione del server e del client
  - Maggiore formalizzazione
  - Pubblicazione formale dell'interfaccia, verifica formale di consistenza, gestione degli errori, vedi soap fault
  - Minor dipendenza fra server e client
  - Si possono sviluppare chiamate complesse con dati anche molto strutturati

References

- SOAP http://www.w3c.org/TR/soap
- WSDL http://www.w3c.org/TR/wsdl
- UDDI http://www.uddi.org
- REST Resources:

http://www.prescod.net/rest

http://internet.conveyor.com/RESTwiki