Corso di Sistemi Distribuiti
Prof. S. Balsamo

Seminario di Sistemi Distribuiti:
"RPC su SOAP"

Massimiliano Vivian [777775]

1 INTRODUZIONE 3

2 RPC 3

3 SOAP (SIMPLE OBJECT ACCESS PROTOCOL) 3

4 UTILIZZO DI SOAP COME PROTOCOLLO PER RPC 6

4.1 RPC E IL CAMPO <BODY> DI SOAP 7
4.2 RPC IL CAMPO <HEADER> DI SOAP 7

5 CONSIDERAZIONI FINALI 7

5.1 VANTAGGI E SVANTAGGI DI SOAP 7

1 Introduzione

La comunicazione delle informazioni rappresenta l'elemento fondamentale dello sviluppo dei sistemi; ciò è diventato particolarmente evidente con la nascita di Internet.

Il linguaggio SOAP e le tecnologie ad esso associate, consentono agli sviluppatori di creare sistemi dotati di maggiore flessibilità.

Con la diffusione di Internet, i dati strutturati si affermano sempre più come lo standard per la comunicazione. I dati strutturati generalmente sono facili da leggere e da gestire e non presentano difficoltà a livello di programmazione. Vengono creati mediante un linguaggio di markup, usato per descrivere il contenuto dei dati all'interno di una struttura gerarchica. Il markup attuale si presenta sotto forma di tag contenenti i dati, i dati all'interno del tag iniziale e del tag finale sono detti valori e l'insieme dei valori e dei tag costituisce un elemento.

Le funzioni principali di un'architettura remota consistono nel convertire uno stack di chiamate in una rappresentazione via cavo, nel trasmettere i dati lungo una connessione di rete e nel ricostruire uno stack di chiamate all'altra estremità prima di eseguire il metodo richiesto. Nelle architetture remote attuali la trasformazione da stack a rappresentazione via cavo è compito del Proxy (o di un equivalente).

2 RPC

RPC (Remote Procedure Call) è una semplice estensione del concetto di chiamata di procedura, consente cioè l'interazione tra procedure di applicazioni diverse oppure che girano su macchine distinte.

Concettualmente non c'è differenza tra una chiamata di procedura locale ed un chiamata di procedura remota, quest'ultima però richiede un'implementazione diversa, ha prestazioni diverse (RPC è più lenta) e quindi è usata per applicazioni di tipo diverso.

Le chiamate remote sono serializzate in un formato comprensibile da entrambe le parti che comunicano; un particolare approccio consente la comunicazione tra macchine Windows e macchine Unix/Linux.

L'importanza di RPC non è nel formato a cui si appoggia, che può essere liberamente scelto, ma nella modularità delle procedure software e nella loro gestione.

La RPC consente di invocare un metodo su un oggetto remoto (in modo simile ad RMI), le specifiche SOAP richiedono che venga identificato l'oggetto da chiamare (tramite il suo URI), il nome del metodo (eventualmente anche la firma) ed i parametri di chiamata.

Alternativamente è possibile passare informazioni di header.

Le specifiche consigliano di utilizzare l'encoding SOAP per codificare queste informazioni, ma il suo uso non è obbligatorio. Nell'area RPC sembra che le specifiche lascino una certa variabilità all'implementazione.

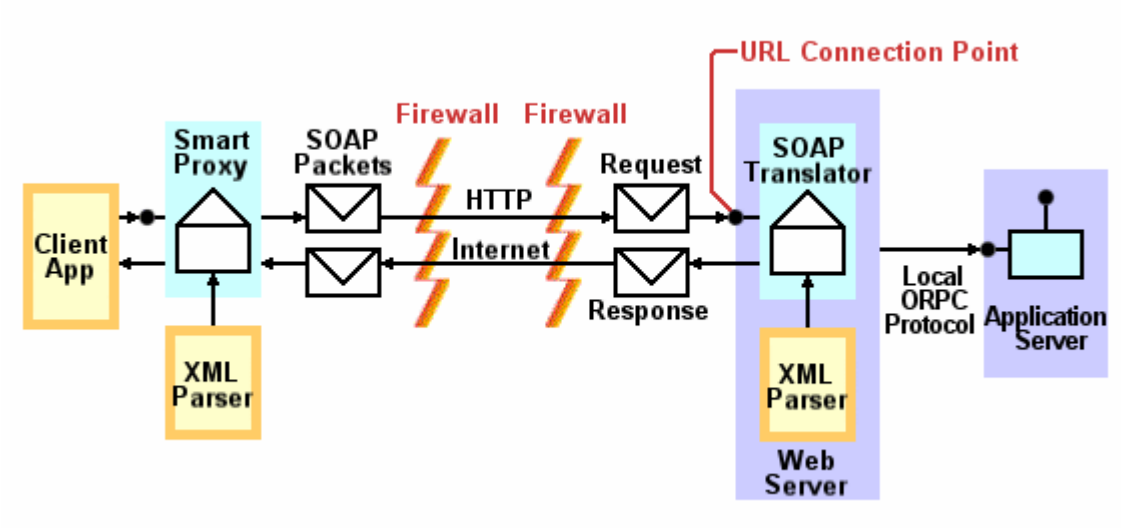
3 SOAP (Simple Object Access Protocol)

La tecnologia SOAP, è un elemento chiave per integrare i sistemi di acquisto, condividere dati aziendali o partecipare alle alleanze commerciali nate nel mondo globale dell'e-business. SOAP è uno standard di RPC (Remote Procedure Call) basato sulla tecnologia XML e sviluppato in origine da Microsoft, Develop Mentor e User Land Software (successivamente raggiunti anche da IBM e Lotus), che può semplificare notevolmente lo sviluppo di applicazioni che utilizzano più linguaggi e coprono più settori di mercato.

SOAP è un protocollo "leggero" per lo scambio di informazioni in un ambiente distribuito. Si tratta di un protocollo via cavo basato su XML che ha lo scopo di migliorare il trasporto di dati su un sistema remoto, eliminando gli ostacoli che intralciano gli attuali pacchetti dei sistemi distribuiti.
La gran parte delle attuali applicazioni basate su Web comunicano usando RPC (Remote Procedure Call), tramite oggetti CORBA o DCOM.

Generalmente, sia i Firewall che i Proxy sono programmati per bloccare questo tipo di traffico. Il protocollo HTTP non è assoggettato alle intercettazioni dei Firewall e per di più, quasi tutti i Web Browser ed i Server lo supportano in modo nativo. SOAP utilizza l'XML e l'HTTP, per aggirare il Firewall e permettere di processare le RPC lanciate dalle applicazioni provenienti sia dalle Intranet che dal Web.

Normalmente, i dati vengono incapsulati all'interno dell'HTTP o di un altro protocollo di trasporto, ed inviati ai Server. I Server estraggono i dati SOAP, li elaborano come richiesto ed inviano i risultati in forma di risposta SOAP.



Dal momento che i protocolli Web sono installati su tutte le principali piattaforme operative, l'HTTP e l'XML costituiscono una soluzione immediatamente disponibile per risolvere il problema dell'interscambio di comunicazioni tra programmi in esecuzione su sistemi operativi diversi. SOAP specifica esattamente come creare un Header HTTP ed un File XML, in modo tale che un programma installato su una macchina possa invocare un programma posto su di un'altra macchina, scambiandosi tutte le informazioni del caso. Nello stesso modo, si può definire come il programma chiamato deve restituire i dati, frutto delle proprie elaborazioni.

In sintesi, SOAP può gestire qualsiasi tipo di messaggio, permettendogli di trasportare dati di ogni genere e scopo. Può operare con tutti i linguaggi di programmazione o di scripting, con tutti i modelli di oggetti e qualsiasi protocollo di connessione Internet.

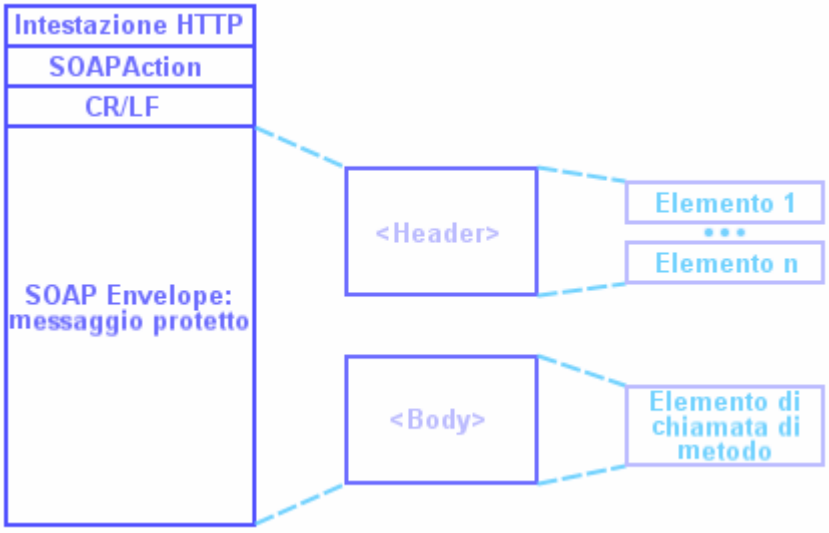
Permette inoltre la comunicazione, indipendentemente dai singoli modelli applicativi: due computer possono lanciarsi reciprocamente delle RPC senza dover conoscere alcunché dei sistemi di Back-end sui quali queste verranno eseguite.

SOAP si compone di tre parti:

1. envelope construct: definisce un framework (struttura), per la descrizione del contenuto del messaggio e della modalità di elaborazione,
2. encoding rules: definisce un'insieme di regole di codifica per lo scambio di tipi di dati definiti dalle applicazioni,
3. RPC representation: definisce una convenzione per la rappresentazione di chiamate e risposte di una procedura remota.

SOAP necessita di un protocollo di trasporto: nella sua versione iniziale si appoggiava all' HTTP standard mentre nelle ultime versioni utilizza anche FTP e altri standard.

La seguente figura illustra il layout del pacchetto SOAP:

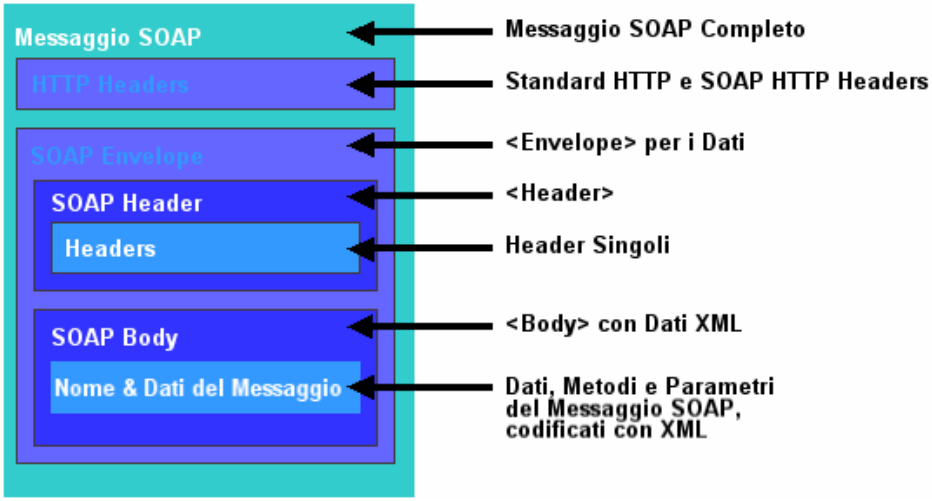


HTTP supporta diversi modi per la richiesta di informazioni mediante un'intestazione di richiesta: utilizza cioè dei metodi per descrivere le intenzioni del server, oppure alcuni campi dell'intestazione per includere le coppie nome-valore dei dati di richiesta. Quando il server risponde genera un messaggio costituito da un'intestazione di risposta che include una riga di stato e i campi dell'intestazione per includere coppie nome-valore dei dati di risposta. Metodi e intestazioni forniscono un framework flessibile, semplice e succinto.

Il protocollo SOAP è adatto a supportare un'architettura client-server: i dati richiesti ed elaborati fra client e server sono organizzati in messaggi SOAP, e vengono trasportati attraverso il protocollo HTTP o un altro protocollo di trasporto. Tali sezioni consentono di implementare applicazioni di tipo RPC, nelle quali un messaggio SOAP contenente una funzione richiamabile, nonché i parametri da attribuire a tale funzione, viene inviato dal client e il server restituisce un messaggio con i risultati della funzione eseguita.

Un vantaggio notevole offerto da SOAP consiste nell'imposizione di una struttura di dati per la richiesta e, quando disponibili, per la risposta.

La seguente figura mostra alcune caratteristiche dei messaggi SOAP:



L'ultima sezione opzionale della specifica SOAP definisce l'aspetto di un messaggio HTTP contenente un messaggio SOAP. Il binding HTTP è importante, in quanto HTTP è supportato da pressoché tutti i sistemi operativi attuali (e molti non recentissimi). Questa funzionalità HTTP è opzionale, ma quasi tutte le implementazioni SOAP la supportano perché è l'unico protocollo standardizzato per SOAP. Ciò ha provocato la diffusione di un pregiudizio: che SOAP richieda HTTP. Questo è impreciso in quanto SOAP supporta (a seconda della versione) diversi protocolli di trasporto, ma visto che quasi tutti gli attuali Web Service utilizzano HTTP perché è estremamente comune, anche SOAP si è adeguato a questa tendenza.

La caratteristica più interessante di SOAP è la possibilità di implementarlo su numerose piattaforme hardware e software diverse. Ciò significa che, grazie a SOAP, si possono collegare tra loro sistemi eterogenei, all'interno e all'esterno di un'organizzazione. In passato si è spesso tentato di elaborare un comune protocollo di comunicazione che fosse utilizzabile per l'integrazione dei sistemi, ma nessuno ha raggiunto una diffusione tanto estesa quanto SOAP. Questo perché SOAP è molto più semplice da implementare rispetto ai protocolli precedenti, quali DCE e CORBA.

Naturalmente SOAP non è in grado di svolgere tutte le funzioni di DCE e CORBA, ma la mancanza di complessità fa trascurare questo limite, rendendo SOAP più rapidamente disponibile.

L'estrema diffusione di HTTP e la semplicità di SOAP fanno di questi due protocolli la base ideale per implementare Web Service in grado di ricevere chiamate da quasi tutti gli ambienti.

Ecco uno schema di invocazione di un Web Service:

4 Utilizzo di SOAP come protocollo per RPC

Uno degli obiettivi principali di SOAP è di incapsulare e scambiare le RPC, utilizzando la flessibilità di XML.

L'attributo di encoding di SOAP può essere utilizzato per indicare lo stile di codifica del metodo di chiamata e/o la risposta, usando la rappresentazione descritta in questa sezione.

Per eseguire una chiamata ad un metodo remoto, SOAP necessita delle seguenti informazioni:

- l'URI dell'oggetto richiesto,
- il nome del metodo da invocare,
- la firma del metodo (opzionale),
- i parametri del metodo,
- intestazioni opzionali.

4.1 RPC e il campo <Body> di SOAP

I metodi di chiamata e risposta RPC, sono entrambi posti nel campo Body di SOAP, mediante la seguente rappresentazione:

- l'invocazione al metodo è modellata come una struttura,
  - l'invocazione al metodo è vista come una singola struttura, contenente un ingresso per ogni parametro di input o di input/output. Questa struttura ha lo stesso nome e tipo del metodo,
    - ogni parametro è visto come un ingresso, con un nome e un tipo, corrispondenti rispettivamente al nome e al tipo del parametro. Questi compaiono nello stesso ordine della firma del metodo,

- la risposta del metodo è modellata come una struttura,
  - la risposta del metodo è vista come una singola struttura, contenente un ingresso per il valore di ritorno, e per ogni parametro di input o di input/output. Il primo ingresso è per il valore di ritorno, seguito dai parametri, posizionati nelle stesso ordine della firma del metodo,
    - ogni parametro è visto come un ingresso, con un nome e un tipo, corrispondenti rispettivamente al nome e al tipo del parametro. Il nome dell'ingresso del valore di ritorno non è significativo. Allo stesso modo, il nome della struttura non è significativo. Per convenzione, si usa dare un nome alla struttura, dopo il nome del metodo, con l'aggiunta della stringa "Response".

- Il fallimento di un metodo, è codificato utilizzando l'elemento SOAP Fault.

Un'applicazione può elaborare delle richieste con dei parametri mancanti, ma può anche ritornare un errore.

Dato che un risultato indica successo, e un errore indica fallimento, è un errore se la risposta del metodo contenere sia il risultato che l'errore.

4.2 RPC il campo <Header> di SOAP

Informazioni addizionali relative alla codifica della richiesta del metodo, ma non facenti parte della formale firma del metodo, possono essere espresse nella codifica RPC. Se questo si verifica, devono essere espresse come sotto-elemento del campo Header di SOAP.

5 Considerazioni finali

Indipendentemente dal protocollo utilizzato, gli sviluppatori hanno necessità di creare un sistema che sia in grado di interoperare con qualsiasi piattaforma e linguaggio di programmazione. SOAP è una soluzione di tipo intermedio, perché è un protocollo di semplice comprensione che fornisce una buona estendibilità, inoltre è già disponibile.

5.1 Vantaggi e svantaggi di SOAP

Tutti gli strumenti presentano alcuni aspetti vantaggiosi e altri che possono apparire svantaggiosi.

La visione che si ha di SOAP dipende interamente dal problema che si intende risolvere e dalle limitazioni imposte dalla situazione in cui si opera.

Un'architettura di oggetti distribuiti richiede l'implementazione di:

- un meccanismo di serializzazione che converta la chiamata di metodo in un formato adatto alla trasmissione in rete,
- un livello di trasporto che trasferisca i dati del metodo tra i sistemi remoti,
- un meccanismo che supporti l'attivazione, la disattivazione ed il rilevamento dell'oggetto,
- l'implementazione di un modello di protezione sia per il sistema remoto sia per il sistema locale.

SOAP gestisce un meccanismo di serializzazione (utilizzando la semantica XML) e di trasmissione (principalmente con il protocollo HTTP). SOAP non è stato progettato per sostituire l'intera architettura distribuita: ad esempio SOAP non implementa la protezione, tuttavia può utilizzare il protocollo HTTP consentendo l'impiego della protezione a livello delle applicazioni.

Diversamente da architetture di oggetti distribuiti, quali ad esempio CORBA, SOAP è semplicemente un protocollo via cavo e quindi per confrontarlo con altre architetture è necessario implementare il protocollo SOAP come parte di un'architettura distribuita autonoma, oppure sostituirlo al protocollo via cavo di un'architettura esistente. In generale quando si confronta il protocollo via cavo SOAP con altri protocolli via cavo viene riscontrata una perfetta adattabilità in diverse aree: SOAP è relativamente efficiente ed è flessibile ad eventuali sviluppi grazie a proprietà derivate da XML.

Consideriamo alcuni vantaggi connessi all'utilizzo di SOAP:

- SOAP è basato su tecnologie aperte e quindi incoraggia le applicazioni distribuite,
- la specifica SOAP può consolidare diversi protocolli HTTP in una singola specifica, inoltre per SOAP l'uso di HTTP non è obbligatorio,
- utilizzando il protocollo, le modifiche apportate all'infrastruttura di SOAP non compromettono le applicazioni,
- SOAP consente di avere un sistema con elevata scalabilità, cioè con grande capacità dell'architettura remota di gestire un numero cospicuo di client concorrenti. In modo particolare SOAP è più scalabile di CORBA, DCOM, e RMI di Java, soprattutto se si usa il modello richiesta/risposta di HTTP.

D'altronde SOAP presenta anche aspetti svantaggiosi:

- sebbene l'ultima versione della specifica SOAP riduca l'uso del protocollo HTTP, è ancora possibile trovare svariate implementazioni che utilizzano HTTP,
- la deserializzazione globale del metodo deve essere eseguita mediante operazioni di rilevamento,
- SOAP serializza per valore ed attualmente non supporta la serializzazione per riferimento. Questo implica che diverse copie dell'oggetto conterranno nel tempo informazioni relative allo stato non coerenti.

SOAP eliminerà la complessità degli attuali approcci alla comunicazione tra più macchine, specialmente quando i sistemi sono separati da collegamenti Wan o Internet. Purtroppo, alla semplicità e alla facilità d'uso di SOAP si contrappongono l'inefficienza delle comunicazioni del suo formato testo e il suo limitato insieme di funzionalità. Ad esempio, SOAP non offre nessun supporto per gli scambi transazionali e le specifiche proposte non comprendono la sicurezza e la consegna garantita dei messaggi. Tuttavia, SOAP può essere facilmente utilizzato all'interno di infrastrutture consolidate, come i sistemi per la gestione delle code di messaggi, che forniscono queste funzionalità, purché a entrambe le estremità di una connessione SOAP si utilizzino le stesse infrastrutture.