SOA

Aspetti e sicurezza

(a cura di **Alessandro Sinibaldi – CISSP, Assistente**

CC, MCSE: Security, CEH, OCP, CWSP, ...)

INDICE DELLA PRESENTAZIONE :

Parte I Il presente e il futuro di SOA

1. SOA e i suoi principi
2. La struttura di SOA
3. Il modello di business di SOA
4. SOA e Web Services
5. Un esempio: la Cooperazione Applicativa nella PA
6. Il futuro di SOA
   - Grid
   - Web Sematico

Parte II La sicurezza di SOA

7. I requisiti di sicurezza
8. Le minacce
9. Le contromisure
   - Standards
   - Architetture di sicurezza
10. Le soluzioni di mercato

Riferimenti bibliografici e sitografici
Varie – Q&A

Cosa è SOA

SOA (Services Oriented Architecture) è un'architettura che rappresenta le funzionalità software come servizi scopribili sulla rete.

Un **servizio** è un meccanismo per abilitare l'accesso a una o più funzionalità di business attraverso un'interfaccia predefinita e in modalità coerente con vincoli e policies.

In SOA il sistema opera come una collezione di servizi. Ciascun servizio può interagire con altri servizi per compiere una certa attività.

I principi di SOA

Distributed Computing

I servizi sono rappresentati come componenti specializzati distribuiti in rete. Ogni servizio processa l'input che gli viene eventualmente fornito e elabora un output

Accoppiamento debole (Loose Coupling)

I servizi comunicano tra loro nascondendo i dettagli implementativi. Una misura dell'accoppiamento tra due sistemi è rappresentata dal numero di cambiamenti che uno qualsiasi dei due sistemi può sopportare senza che la loro interconnessione venga interrotta. Alcuni dei vantaggi principali dell'accoppiamento debole sono la possibilità di isolare meglio i problemi quando si verificano e quello di sostituire un servizio con un altro senza riscrivere l'intera applicazione

Interoperabilità

I servizi sono in grado di cooperare tra loro, indipendentemente dalla tecnologia utilizzata per costruirli

Trasparenza di rete

È possibile utilizzare un servizio indipendentemente da dove esso sia fisicamente localizzato sulla rete

Asincronismo

Un servizio è in grado di utilizzarne un'altro inviando alle sue interfacce pubbliche la richiesta di elaborazione, disconnettersi e ricevere successivamente la notifica dell'avvenuta elaborazione con l'eventuale output

Trasparenza rispetto al vendor

L'interoperabilità, l'accoppiamento debole e la trasparenza di rete sono possibili solo se i servizi usano metodi standard per comunicare tra loro. L'unica cosa che necessita di essere definita e standardizzata è l'interfaccia, mentre l'implementazione può essere tranquillamente proprietaria.

Separation of Concerns

Rappresenta la separazione della logica di business dalla logica dei computer. L'informatica è un mezzo, non un fine e il business, che invece è l'obiettivo, deve essere in grado di evolvere senza vincoli tecnologici.

Le conseguenze

Una delle conseguenze più importanti dei principi SOA è il ***riutilizzo*** della logica applicativa e la minimizzazione degli ambienti di produzione.
Il riutilizzo passa attraverso la realizzazione di opportuni ***adapters*** (adattatori) che da una parte si innestano sugli applicativi esistenti e dall'altra espongono interfacce orientate a SOA. Essi di fatto fungono da traduttori.

Il modello che ne deriva è simile alla ***programmazione per componenti***.
In quell'ambito, si costruiscono componenti software specializzati il cui scopo è fare una cosa sola ma, possibilmente, farla bene. A questo punto, in uno sviluppo tipo Lego, i componenti vengono uniti insieme per fare funzionalità più complesse.

Cambiare la logica di business, vuol dire soltanto cambiare il modo in cui i servizi vengono interrogati, eventualmente aggiungendoli o rimuovendoli.

La struttura di SOA

SOA Registry

è il posto dove le informazioni sui servizi di business vengono pubblicate, a beneficio di clienti, fornitori e partners. Un servizio fa conoscere la sua esistenza al mondo solo se compare nel SOA Registry. Sono le “Pagine Gialle” dell’architettura SOA.

Workflow Engine

è un componente software che gestisce il workflow tra i vari processi di Business.

Service Broker

è il componente che lega i processi tra di loro. Usa le informazioni nel SOA Registry e unisce i servizi insieme per il Workflow Engine. Di fatto è il motorino di avviamento di SOA e appartiene alla categoria Middleware. SOA Registry e Service Broker identificano insieme un **dominio SOA**.

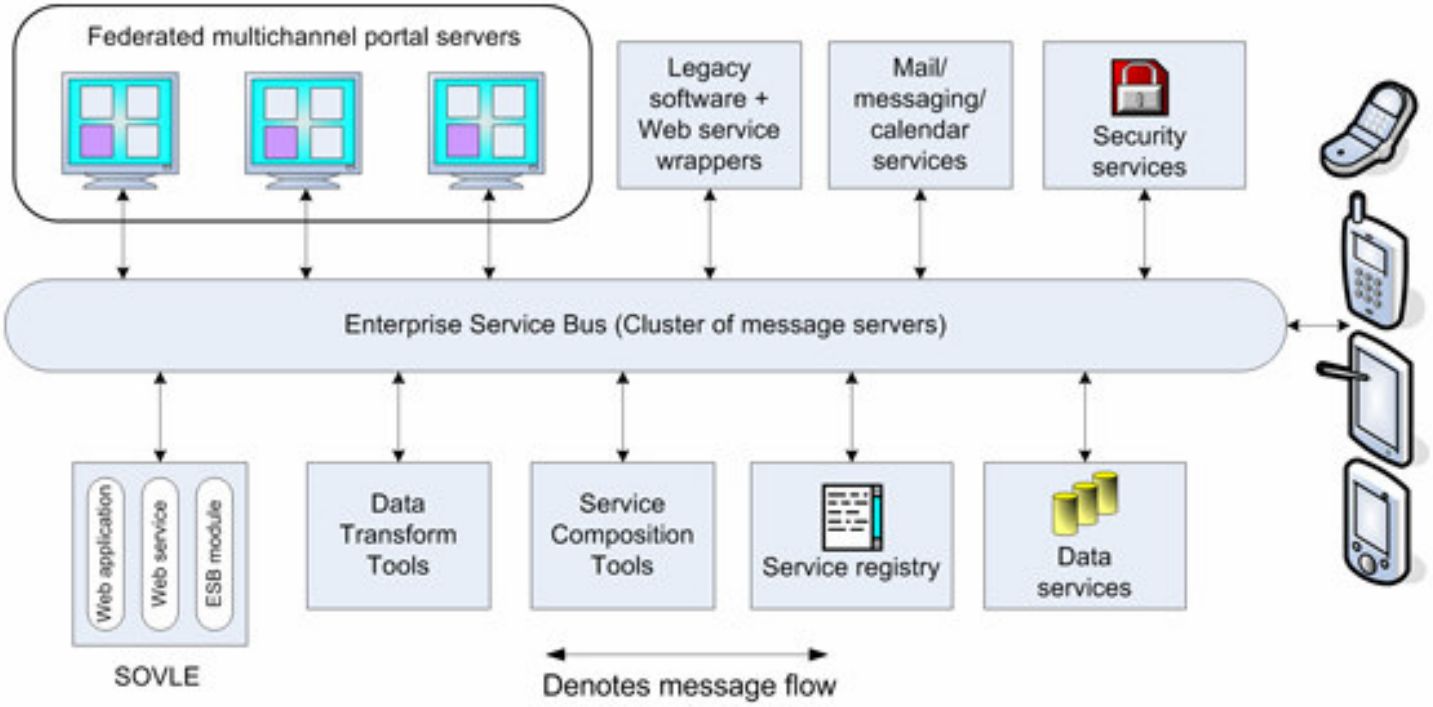
Enterprise Service Bus (ESB)

Affinché SOA funzioni correttamente, è necessario che i servizi siano debitamente connessi tramite un’infrastruttura affidabile. Suo compito è quello di permettere ai servizi di scambiarsi messaggi. E’ una specie di service broker.

SOA Supervisor

È il direttore d’orchestra, un componente cioè che mette in connessione i vari servizi a tempo debito, instaura transazioni e effettua il monitoraggio delle prestazioni. Esso viene attivato dal Service Broker e deve monitorare i tempi di risposta dei singoli servizi, in accordo ai vari Service Level Agreement, identificando eventuali guasti o malfunzionamenti.

La struttura di SOA



A cosa serve l'ESB?

Messaggistica

I vari servizi connessi nell'architettura SOA comunicano tra loro scambiandosi messaggi. Questi possono essere di tipi diversi:

- **Messaggio punto-punto**: va da un mittente a un destinatario
- **Messaggio punto-punto di tipo richiesta-risposta**: il mittente resta in attesa della risposta.
- **Messaggio di tipo broadcast**: da un mittente a tutti
- **Messaggio di tipo broadcast di tipo richiesta-risposta**: da un mittente a tutti ma richiede risposta
- **Messaggio di tipo Publish-Subscribe**: un messaggio viene depositato dal mittente in un'area condivisa con più sottoscrittori, che hanno manifestato l'interesse a ricevere il messaggio e che ne controlleranno periodicamente l'esistenza. È il modello della Pubblica Amministrazione
- **Messaggio di tipo Store and Forward**: la comunicazione non richiede la presenza simultanea di mittente e destinatario ma funziona come una segreteria telefonica, immagazzinando temporaneamente i messaggi che non è stato possibile recapitare.

A cosa serve l'ESB?

Gestione dei servizi

effettua il monitoraggio delle performances, eventualmente fornendo Quality of Service (QoS) in base alle priorità

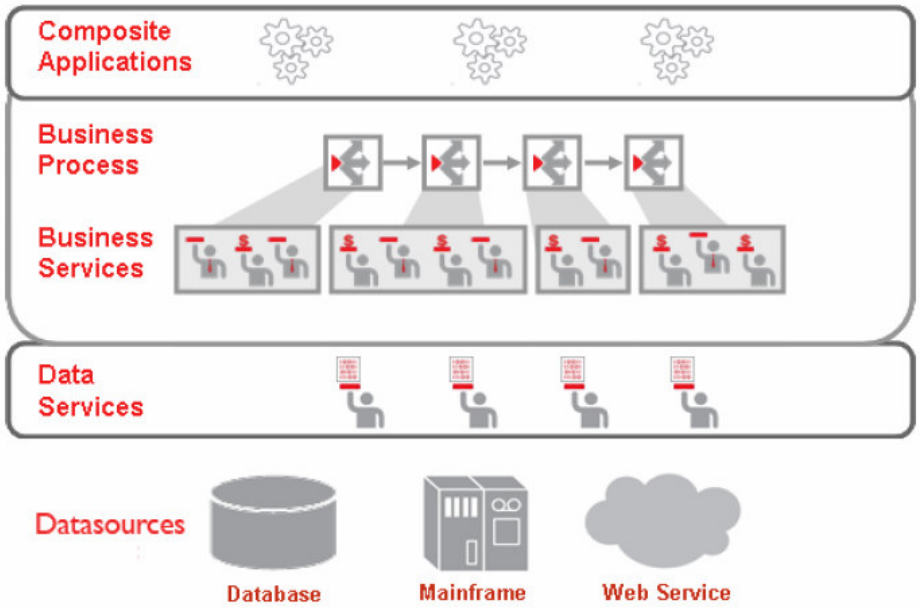
Mediation

Trasforma i messaggi tra i formati usati dalle applicazioni mittente e destinatario. Un esempio può essere la conversione tra più tipi data, come tra un database Oracle e un database SQL Server.

Sicurezza

Connette i servizi tra loro rispondendo di eventuali esigenze di sicurezza come riservatezza, integrità, disponibilità. Per accedere a un determinato servizio potrebbe così richiedere autenticazione e autorizzazione.

Quali servizi?



Processi di Business

rappresentano applicazioni complesse ottenute tramite l'aggrgazione di Servizi di Business e fanno eventualmente uso di Workflows, per gestire il flusso di attività. Via via che cresce la complessità dei servizi aggregati, aumentano i rischi di perdita di affidabilità e disponibilità. Se, ad esempio, un servizio è costruito a partire da 6 servizi, ciascuno dei quali ha un 99% di uptime, il servizio risultante non supererà il 94% di uptime

Servizi di Business

contengono la logica di business per attività ben precise e specifiche ed accedono ai dati tramite i Servizi Dati quando necessario.

Servizi Dati

incapsulano l'accesso ai dati fornendone una vista consistente e consolidata in base alle esigenze di business.

Il modello di business di SOA

Il focus di SOA è permettere una rapida riconfigurazione dei servizi a fronte di nuove esigenze di Business.

Cosa chiede il Business all'IT

Efficienza e incremento di produttività	85%
Vantaggio competitivo	63%
Soddisfazione della clientela	59%
Innovazione di business	58%
Generazione di fatturato	38%
Miglioramento globale delle attività operative	18%
Miglioramento della supply chain	16%
Adeguamento a leggi e normative	14%
Business Continuity	11%
Privacy e sicurezza	5%
Altro	13%

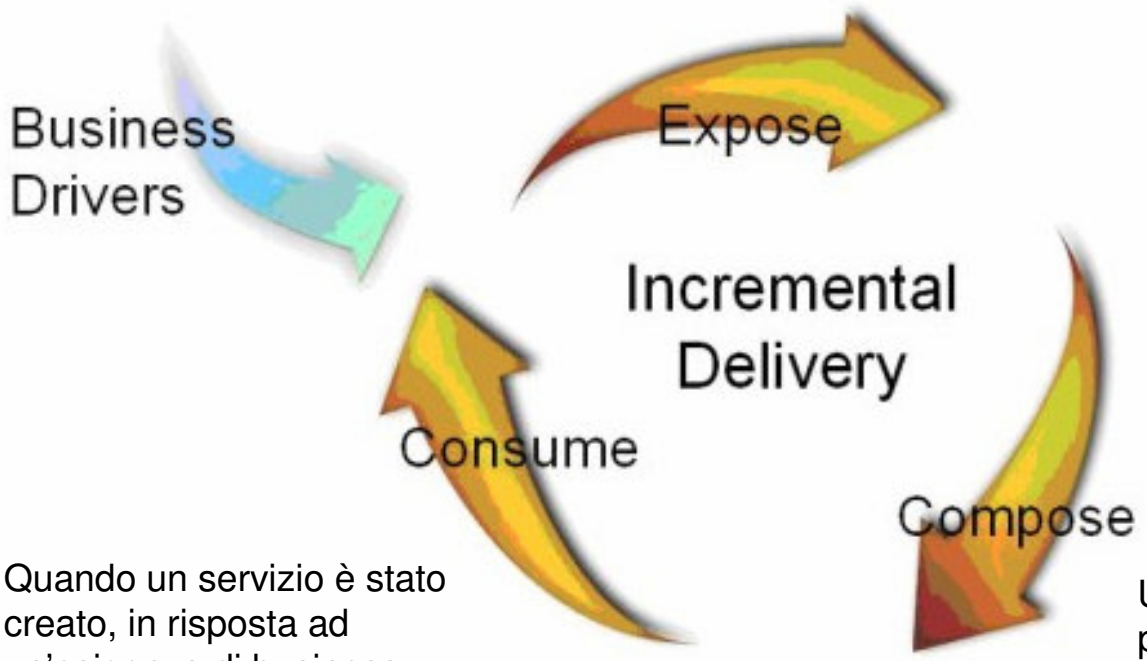
CIO Magazine, da un campione di 100 CIO (Chief Information Officer) USA (possibili risposte multiple).

Quali sono i problemi dell'IT

- ***Budget vincolati***
  Necessità di riutilizzare i sistemi esistenti piuttosto che sostituirli e migliore utilizzo delle risorse (massimizzazione del ROI)
- ***Acquisizioni e merging***
  Necessità di integrare sistemi IT eterogenei
- ***Globalizzazione***
  Necessità di fornire servizi anytime e everywhere, con una varietà di interfacce utente diverse (multicanalità e localizzazione)
- ***Risposte veloci a nuovi requisiti***
  Necessità di accorciare i tempi di sviluppo e necessità di virtualizzare l'infrastruttura per allocare dinamicamente le risorse, spostandole laddove risultino più necessarie
- ***Aderenza a policies e normative***
- ***Sicurezza***
- ***Soddisfazione del cliente***
  Necessità di effettuare il monitoraggio del servizio erogato garantendo i Livelli di Servizio (SLA) contrattati

Il ciclo di vita di SOA

Questa fase si focalizza su quali servizi creare a partire dalle applicazioni e dai dati e su come esporli



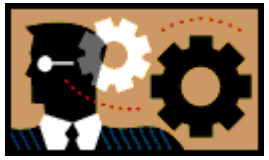
Quando un servizio è stato creato, in risposta ad un'esigenza di business, esso deve essere reso disponibile ad altri sistemi e servizi

Una volta creati, i servizi possono essere uniti in servizi più complessi

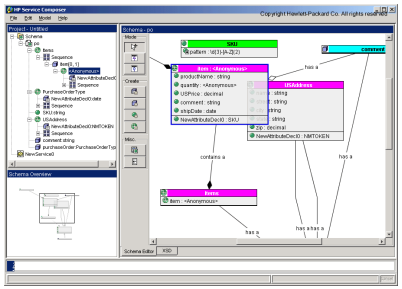
I ruoli

Service Providers

Service Consumer



Service Composer



1. **raccoglie** i requisiti dal service consumer
2. **individua** sul mercato i servizi già esistenti che possano essere utilizzati allo scopo
3. **contatta** i service providers e stabilire dei contratti con essi
4. **integra** i servizi
5. **espone** verso il consumer il servizio risultante come se fosse esso stesso un service provider, nascondendo così la complessità sia tecnologica che contrattuale sottostante

Software and Services

Applicazioni tradizionali

consistono nel normale software installato sul pc e che è dotato di tutte le funzionalità necessarie. Non necessita a priori di una connessione a Internet. L'utente paga il possesso materiale del software. Un esempio è Microsoft Office "classico"

Software + Services

sono applicazioni web composte sia di un software installato sul pc che di una serie di servizi web a cui l'applicativo può connettersi per usufruire di funzionalità aggiuntive (spesso di tipo collaborativo). Un esempio è Microsoft Project usato in combinazione con Microsoft project server e Sharepoint Services, oppure le Office Business Applications (OBA)

Software-as-a-Service (SaaS)

in tal caso l'applicativo risiede interamente in rete ed è acceduto on demand con una tipica formula ad abbonamento o pay-per-use. Il Provider del Servizio si prende carico dell'hosting, degli aggiornamenti, eventuali backup ecc. Il termine sostituisce un'altra vecchia espressione, che era quella di ASP (Application Service Provider). Esistono due modalità con cui l'utente può accedere al servizio: tramite Web Services o tramite REST(Representational State Transfer)

Internet Service Bus e Mash-up

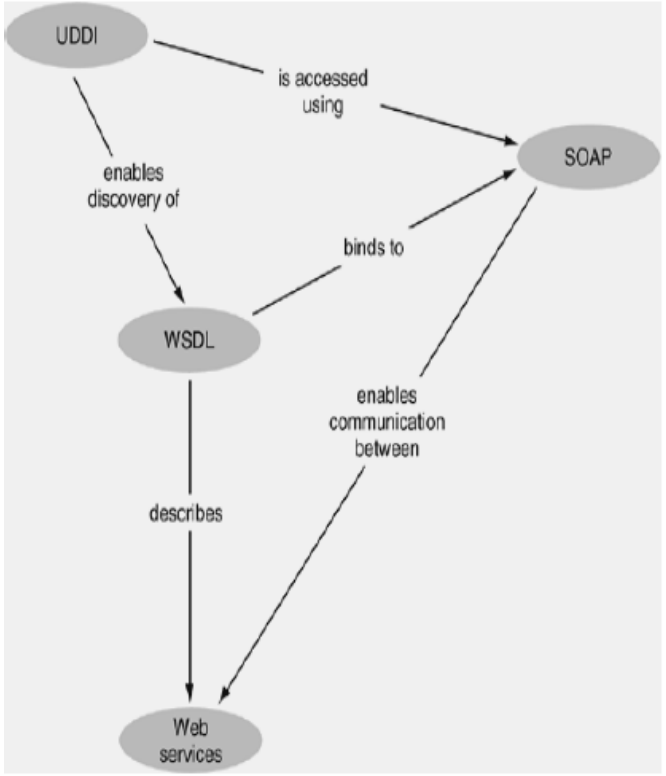
Il modello a servizi sparsi in rete e il Web 2.0 aprono agli utenti nuovi modi di pensare le applicazioni su internet. Inizia l'era dei **mash-up**, cioè delle applicazioni web che combinano i dati provenienti da più servizi per realizzare applicazioni innovative, ad esempio attraverso l'unione delle Google maps con i dati meteo di weather.com o l'elenco dei ristoranti stellati della guida Michelin e i blogs con le recensioni dei gastronauti e il feed RSS del Gambero Rosso. Anche il modello di programmazione cambia: non più applicazioni offerte as-is agli utenti, che le utilizzano e basta, ma sono questi ultimi che hanno la possibilità di costruirsi, utilizzando facili strumenti (i **mashup editor**), come ad esempio Teqlo (http://www.teqlo.com) o Google Mash-up Editor (http://editor.googlemashups.com) o Intel Mash Maker (http://mashmaker.intel.com) o JackBe Presto (http://www.jackbe.com), le web application di cui hanno bisogno, magari rendendole a loro volta disponibili attraverso il proprio sito web personale.

Con i mash-up, Web 2.0 diventa anche condivisione di applicazioni, oltre che di contenuti e di esperienze e internet diventa l'infrastruttura con cui i servizi collaborano. Il concetto di Service Bus evolve dall'ambito Enterprise verso un ***Internet Service Bus***, che diventa il middleware su cui si basa il Web programmabile (http://www.programmableweb.com).

SOA e WEB Services

I web services sono implementazioni di SOA perchè si basano su una logica orientata ai servizi. D'altronde, SOA non è necessariamente realizzata tramite Web Services. La differenza che corre tra i due è che SOA è un modello architetturale, mentre web services sono un modello di implementazione.

Un Web service è un applicativo identificato da un URI, che può essere acceduto attraverso Internet mediante la sua interfaccia esposta. La descrizione dell'interfaccia dichiara le operazioni che possono essere compiute dal servizio, i tipi di messaggi che devono essere scambiati durante l'interazione con il servizio, e la posizione fisica delle porte, dove l'informazione dovrebbe essere scambiata.



SOAP

(Simple Object Access Protocol) è una specifica di formattazione di messaggi in formato XML e si appoggia a protocolli come HTTP e SMTP per la trasmissione di questi via Internet. Ogni messaggio è costituito da un mittente, un destinatario e da un numero arbitrario di nodi intermedi che processano il messaggio e lo instradano verso il destinatario. Il corpo del messaggio, obbligatorio, contiene l'informazione destinata al ricevente mentre l'header (intestazione), che è opzionale, contiene informazioni addizionali che vengono processate dai nodi intermedi.

UDDI

(Universal Description Discovery and Integration) è un meccanismo che permette ai clients di trovare i web services in modo dinamico. È simile a un server DNS ma, al contrario di questo, che permette di identificare macchine sulla rete, UDDI permette di identificare applicazioni di business. UDDI è strutturato su quattro livelli: al livello più alto, o Pagine Bianche, è la lista delle organizzazioni; i livelli seguenti, o Pagine Gialle, sono la lista dei servizi esposti da ogni organizzazione, suddivisi per categorie; infine l'ultimo livello, o Pagine Verdi, sono le informazioni tecniche necessarie affinché il client possa connettersi al singolo servizio.

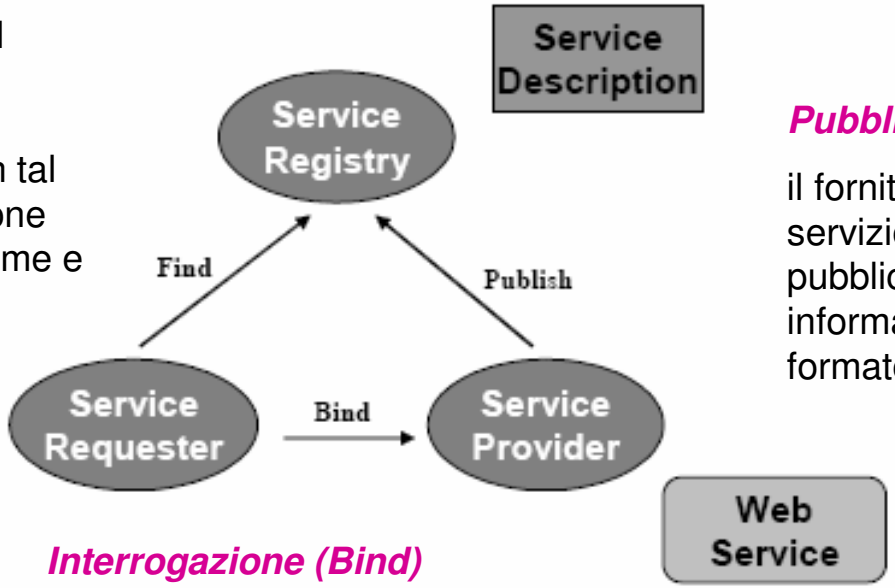
WSDL

(Web Service Description Language) definisce i servizi come una serie di endpoint sulla rete, o porte, che possono essere opportunamente interrogati fornendo parametri di input e ottenendo dati in output

Il ciclo di vita dei WEB Services

Reperimento (Find)

quando un fruitore deve usare un servizio, trova il servizio desiderato costruendo una query o consultando il registro. In tal modo, ottiene la definizione dell'interfaccia, e cioè nome e messaggi da scambiare



Pubblicazione (Publish)

il fornitore del servizio crea il servizio e le sue interfacce e pubblica nel registro UDDI le informazioni relative in formato WSDL

Interrogazione (Bind)

il fruitore effettua una chiamata per invocare il servizio. Lo scambio di messaggi utilizza lo standard SOAP che viene trasportato su un protocollo di rete come HTTP, SMTP o altri ancora.

Un esempio: la Cooperazione Applicativa

Il modello di riferimento è quello di una Pubblica Amministrazione con competenze amministrative decentrate e tendente al federalismo.

L'obiettivo è l'erogazione integrata dei servizi di ogni amministrazione pubblica, sia centrale che locale e indipendente dal canale di erogazione.

Per fare questo, è necessario stabilire le modalità di interazione tra i vari soggetti costituenti la pubblica amministrazione, salvaguardando la libertà di effettuare le singole scelte tecnologiche a livello locale.

Un esempio: la Cooperazione Applicativa

Ogni singola Amministrazione definisce un ***Dominio dei servizi applicativi***, inteso come l'insieme delle risorse (in particolare le procedure, i dati e i servizi) e delle politiche di una determinata organizzazione, nonché il suo confine di responsabilità .

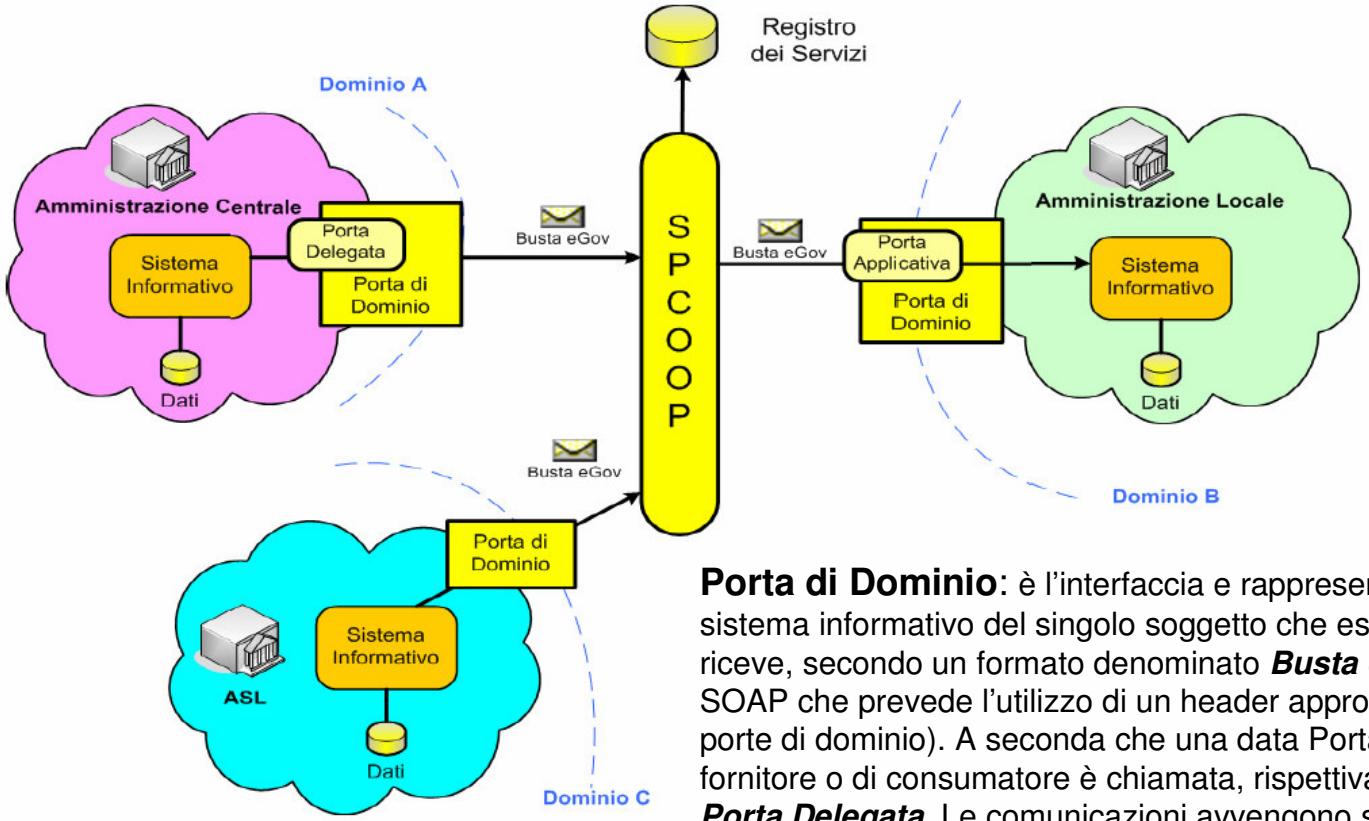
Un servizio applicativo è erogato in seguito ad un accordo tra due soggetti, un fornitore e un consumatore. Tale accordo viene formalizzato in linguaggio xml e prende il nome di ***Accordo di Servizio***. Affinché il servizio venga effettivamente considerato disponibile attraverso SPCoop, tale accordo deve essere reso pubblico. I servizi applicativi vengono erogati e fruiti attraverso tecnologie e standard indicati genericamente come Web Service.

Un esempio: la Cooperazione Applicativa

Insiemi di amministrazioni che erogano un servizio comune, ottenuto tramite la cooperazione tra i singoli costituenti, prendono il nome di ***Dominio di Cooperazione***.

Il servizio così erogato, oltre a essere corredato dall'Accordo di Servizio, deve pure prevedere l'esistenza di un ***Accordo di Cooperazione***, trasparente ai fruitori, e che descrive le modalità con cui i differenti soggetti cooperano per la fornitura del servizio finale.

Un esempio: la Cooperazione Applicativa



Porta di Dominio: è l'interfaccia e rappresenta l'unico elemento del sistema informativo del singolo soggetto che espone dati verso l'esterno o ne riceve, secondo un formato denominato ***Busta di e-gov*** (è un'estensione SOAP che prevede l'utilizzo di un header appropriatamente predisposto dalle porte di dominio). A seconda che una data Porta di Dominio svolga il ruolo di fornitore o di consumatore è chiamata, rispettivamente ***Porta Applicativa*** o ***Porta Delegata***. Le comunicazioni avvengono sempre fra Porte Delegate e Porte Applicative. Ogni Porta di Dominio è logicamente suddivisa in una componente di ***Cooperazione*** ed una di ***Integrazione***: la prima, rivolta verso l'esterno della Porta di Dominio, è incaricata delle comunicazioni fra Porte di Dominio, mentre la seconda interagisce con la logica e l'architettura dell'infrastruttura che deve servire.

Un esempio: la Cooperazione Applicativa

Il modello è ulteriormente supportato da un altro elemento, denominato ***SICA*** (Servizi di Interoperabilità, Cooperazione ed Accesso), che di fatto costituisce l'infrastruttura generale che deve permettere tale cooperazione applicativa tra le amministrazioni.
SICA assolve i seguenti compiti:

- ***funzioni di sicurezza***
- ***indici di risorse e ruoli***
  sono di fatto le "Pagine Gialle" della Pubblica Amministrazione e contengono le informazioni relative a tutti i soggetti/utenti (riferimenti, mansioni, ruoli, ecc.)
- ***servizi di coordinamento e gestione***
- ***registri di servizi***
  offre funzionalità di registrazione, accesso, aggiornamento, cancellazione e ricerca degli Accordi di Servizio e di Cooperazione
- ***archivi di schemi***
  offre funzionalità di gestione della semantica dei servizi e delle informazioni da essi veicolati

Un esempio: la Cooperazione Applicativa

Un'amministrazione che voglia erogare un servizio deve specificarne tutti gli aspetti e in particolare:

1. ***interfaccia del servizio***
   intesa come insieme di operazioni (con eventuali parametri in input e tipi di dato in output) facenti parte del servizio
2. ***protocollo utilizzato dal servizio***
3. ***punti di accesso (endpoint di rete)***
   attraverso cui il servizio è fisicamente erogato
4. ***livelli di qualità del servizio garantiti***
5. ***requisiti e caratteristiche di sicurezza***
6. ***descrizione della semantica del servizio e dell'informazione veicolata dal servizio***

I punti 1 e 3 possono essere soddisfatti tramite l'utilizzo di WSDL

Open Grid Services Architecture (***OGSA***) progetto open source che unisce i concetti di GRID e di Web Services, attraverso un insieme di specifiche e di standards. Il sito di riferimento è http://www.globus.org/ogsa.

Il problema di base, che il progetto vuole affrontare, è quello di permettere alle applicazioni di accedere e condividere risorse e servizi distribuiti attraverse Wide Area Networks, come Internet.

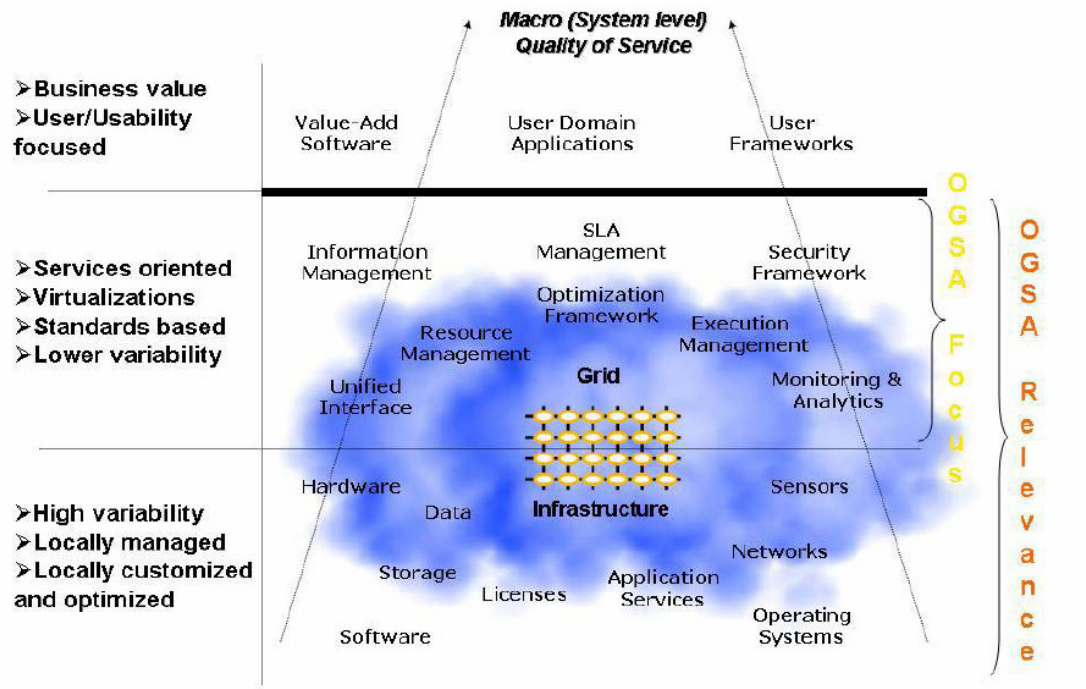
L'obiettivo è quello di creare, a partire da componenti geograficamente e organizzativamente distribuiti, sistemi di calcolo virtuali (Organizzazioni Virtuali), sufficientemente integrati da poter erogare un servizio di qualità.

Le risorse dovrebbero poter essere scoperte dinamicamente, prenotate, allocate, accedute in modo sicuro e gestite.

Il modello che ne deriva è quello di un insieme di Fornitori di Servizio (Service Providers) distribuiti via rete, a cui accedere on-demand e facendo uso di un'infrastruttura flessibile, per compiere tasks di calcolo complessi.

Il paradigma che si utilizza è quello di un'architettura basata sulla composizione di building blocks.

Il Futuro di SOA: SOA e GRID



Architettura di OGSA: il livello più basso sono le risorse fisiche, il livello intermedio rappresenta virtualizzazione e astrazione logica, mentre il livello in alto sono le applicazioni e i servizi

Il punto di partenza dell'architettura è il ***Globus Toolkit***, che è un'architettura aperta e open source, basata su un insieme di librerie software e di servizi, a supporto del GRID.

Il toolkit supporta problematiche di sicurezza, discovery dell'informazione, gestione delle risorse, comunicazione, rivelazione degli errori e portabilità.

È composto da una serie di elementi, di cui i più importanti sono:

- il protocollo ***Grid Resource Allocation e Management*** (GRAM) ed il suo servizio "gatekeeper", che provvede alla creazione ed alla gestione di un servizio sicuro ed affidabile;

- il ***Meta Directory Service*** (MDS-2), che memorizza le informazioni riguardanti le risorse in maniera *soft state* (le informazioni variano periodicamente in quanto vengono mantenute per un tempo massimo prestabilito). Tali informazioni verranno utilizzate per il discovery di una risorsa. Inoltre si occupa del data modeling e fornisce un registry locale (GRAM reporter). MDS-2 occupa, come si vede dalla definizione, in ambito GRID un ruolo paragonabile a un registro UDDI in ambito Web Services, con la differenza che, mentre il secondo è statico, MDS è dinamico, cioè assomiglia molto a un DNS dinamico in cui le risorse si registrano al volo.

- il ***Grid Security Information*** (GSI), che supporta il single sign on e la delegation.

OGSA è basato su WSDL (Web Services Desription Language) per la descrizione delle interfacce dei servizi. WSDL descrive sia la posizione dei servizi che i metodi da questi esposti, tramite un documento XML.

Gli elementi principali di un documento WSDL sono:

<portType> sono le operazioni compiute dal web service
<message> sono i messaggi usati dal web service
<types> sono i tipi di dati usati dal web service
<binding> è il protocollo di comunicazione usato dal web service

```
<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>
```

L'elemento <portType> è il più importante perchè definisce un web service, le operazioni che possono essere effettuate e i messaggi coinvolti.
OGSA definisce dei propri portType. Un esempio è Register, che ha come operazioni ammesse RegisterService e UnRegisterService, per registrare e deregistrare dinamicamente un grid service handle (cioè un puntatore di tipo URI, Uniforme Resource Identifier, al Grid Service).
Oltre a questa che abbiamo descritto, sono disponibili altri portType per la scoperta di servizi, la distruzione e la gestione (GridService), la creazione (Factory) e la notifica (NotificationSource e NotificationSink).

Un web service dotato di queste interfacce prende il nome di ***Grid Service***.
In tal modo lo strato dei web services va a sovrapporsi allo strato del Grid.

Dato +

Significato =

Informazione

"40"

"40 è il numero di
numero di partecipanti
a questo seminario"

Informazione +

Organizzazione =

Conoscenza

"Confrontando il numero dei partecipanti ISACA a ogni seminario mensile, si denota un trend di crescita"

Lo scopo del Web semantico è quello di associare un significato ai dati presenti su web, passando così da un Web formato da dati collegati tra loro (collegamento ipertestuale) in modo artificiale a informazioni collegate tra loro in modo logico, in modo da produrre ***Conoscenza****.*

L'uso di web semantico permetterà ad esempio ad un utente interessato a trovare su un motore di ricerca tutto ciò che ha a che fare con la pésca, intesa come sport acquatico, di non dover scorrere pagine e pagine di risultati che hanno a che fare con la pèsca, intesa come frutto.

Questo risultato si ottiene tenendo presente il ***contesto*** in cui la parola viene detta. Tenere presente il contesto vuol dire tenere di conto dei ***concetti*** presenti nel testo e delle relazioni tra i concetti.

Se così, all'interno del testo, si sta parlando di acqua, di pesci, di sport, probabilmente si sta parlando di un certo significato della parola pesca, contrariamente a quanto succederebbe se nel testo si stesse parlando di coltivazione, agricoltura biologica ecc.

A tale scopo, si parla di ***Intelligenza Collettiva***.

L'estensione semantica ai web services, detta ***Semantic Web Services***, è necessaria qualora si lavori in un ambiente eterogeneo distribuito.

WSDL, infatti, si limita a descrivere la sintassi dei servizi e non la loro semantica, cioè il significato dei parametri ad esempio di input o output o i vincoli applicabili.
Questo scopo richiede una ricchezza espressiva maggiore.

Il fondamento è il Web Service Modeling Framework (WSMF)

Le fasi necessarie per utilizzare un web service semantico (da confrontare con le rispettive nel caso SOA standard) sono le seguenti:

pubblicazione

permette alle applicazioni di scoprire l'esistenza di web services che soddisfano gli obiettivi (goals) posti. In questo caso si usa un registro semantico, per l'immagazzinamento delle informazioni. L'ontologia del servizio distingue tra informazioni necessarie a scoprire il servizio e informazioni necessarie ad invocarlo.

Reperimento

In questo caso, la ricerca che restituisce tutti is servizi che rispondono ai nostri obiettivi, può anche essere effettuata in base a query complesse, come ad esempio il tipo di parametri in input e output, Le precondizioni ecc.

Selezione

la selezione è necessaria, una volta estratto il gruppo di servizi che soddisfano gli obiettivi. Possono essere anche usati, come criteri per la scelta, requisiti non funzionali come il livello di servizio o il costo

Composizione (o Orchestrazione)

web services complessi possono essere definiti come composizione, anche basata su workflow, di servizi più semplici

Invocazione

dopo essere stato preparato il pacchetto di richiesta al web service, questo viene validato a fronte delle ontologie del servizio

Erogazione

Gestione dell'ontologia

Gli attori coinvolti sono ovviamente aumentati:

un reasoner

è coinvolto in tutte le fasi ed ha il compito di interpretare le descrizioni semantiche e le queries. È il "cervello" del processo

un register

fornisce il meccanismo per pubblicare e trovare servizi in un registro semantico ma anche per creare e modificare le definizioni stesse

un matchmaker

è il mediatore tra reasoner e register durante la fase di scoperta e selezione del servizio

un decomposer

è il componente che deve eseguire il modello di composizione dei servizi complessi

un invoker

è il mediatore tra fruitore e fornitore di un servizio o tra decomposer e fornitore del servizio

WSMF è basato su ***Web Service Modelling Ontology*** (***WSMO***), il cui scopo è fornire una tecnologia coerente per la scoperta semiautomatizzata, la composizione e l'esecuzione di web services e che utilizzi la logica inferenziale, capace cioè di eseguire ragionamenti che, partendo da premesse date, portino a delle conclusioni nuove rispette alle conoscenze di partenza.

Gli elementi necessari per raggiungere questo obiettivo sono 4:

ontologie

sono il modo per condividere il significato delle informazioni tra uomini e macchine e rappresentano relazioni tra concetti

Web Services

essi pubblicano le proprie ontologie. Queste devono essere confrontate con le ontologie espresse dagli obiettivi di chi richiede un servizio, per poter capire quali web services rispondano effettivamente alla richiesta e quali no.

Obiettivi (Goals)

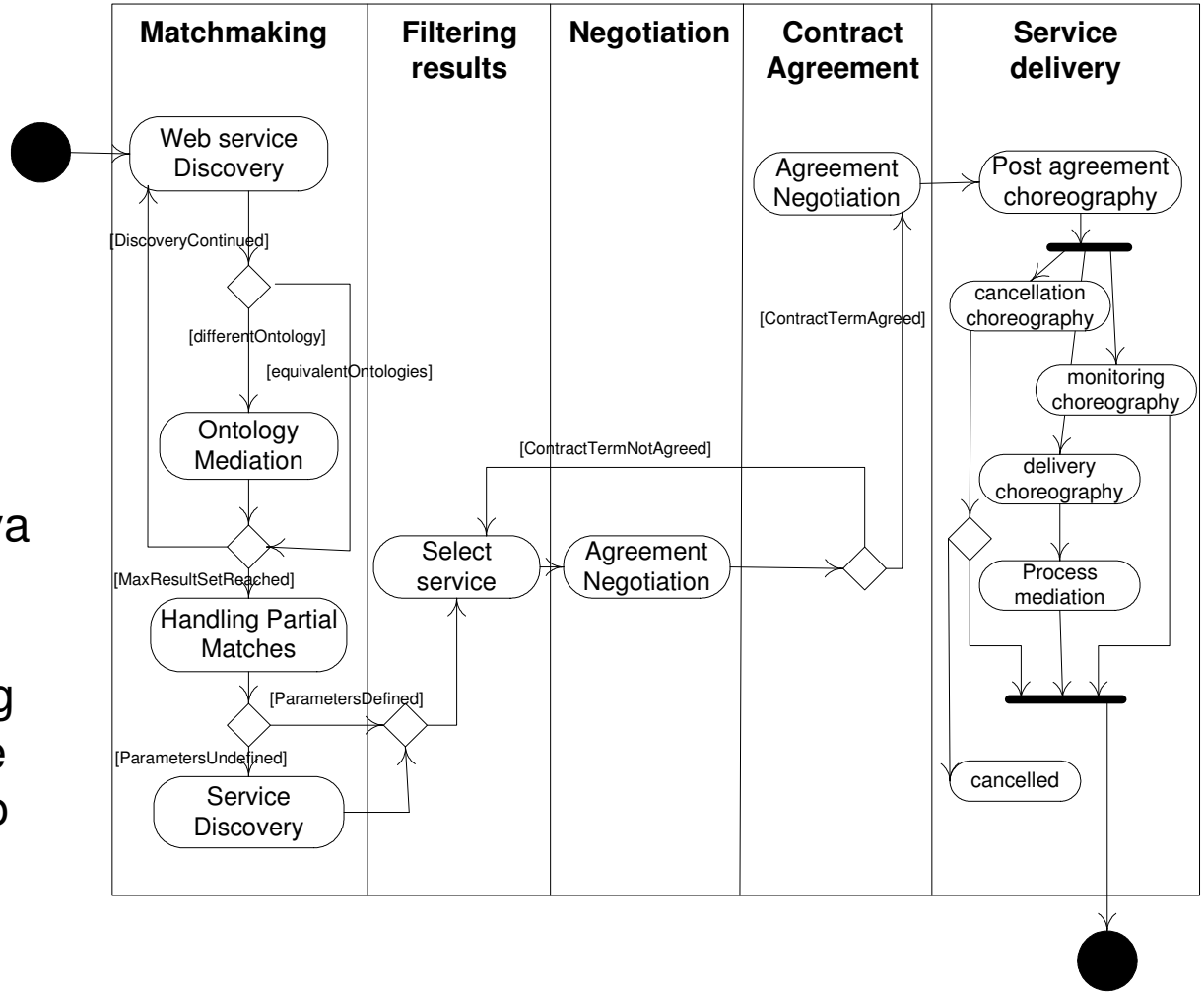
chi richiede un servizio, formula un obiettivo che deve essere soddisfatto dai web services. Questo obiettivo è espresso in forma dichiarativa che la macchina deve poter interpretare, condividendo il significato dei termini con l'uomo.

Mediatori

hanno l'obiettivo di mediare, in un ambiente eterogeneo, ontologie diverse.

Web Service Execution Environment (***WSMX***) rappresenta un'implementazione di WSMO.

Il ruolo coperto da WSDL nei web services "normali", che descriveva la sintassi, è ora ricoperto da ***WSML*** (Web Services Modeling Language) che descrive la semantica dell'utilizzo dei WEB Services



INDICE DELLA PRESENTAZIONE :

Parte I Il presente e il futuro di SOA

1. SOA e i suoi principi
2. La struttura di SOA
3. Il modello di business di SOA
4. SOA e Web Services
5. Un esempio: la Cooperazione Applicativa nella PA
6. Il futuro di SOA
   - Grid
   - Web Sematico

Parte II La sicurezza di SOA

7. **I requisiti di sicurezza**
8. **Le minacce**
9. **Le contromisure**
   - **Standards**
   - **Architetture di sicurezza**
10. **Le soluzioni di mercato**

Riferimenti bibliografici e sitografici
Varie – Q&A

I requisiti di sicurezza

Come identificare e autenticare chi richiede un servizio?

Come identificare e autenticare la sorgente di un messaggio?

Il mittente è autorizzato a spedire questo messaggio?

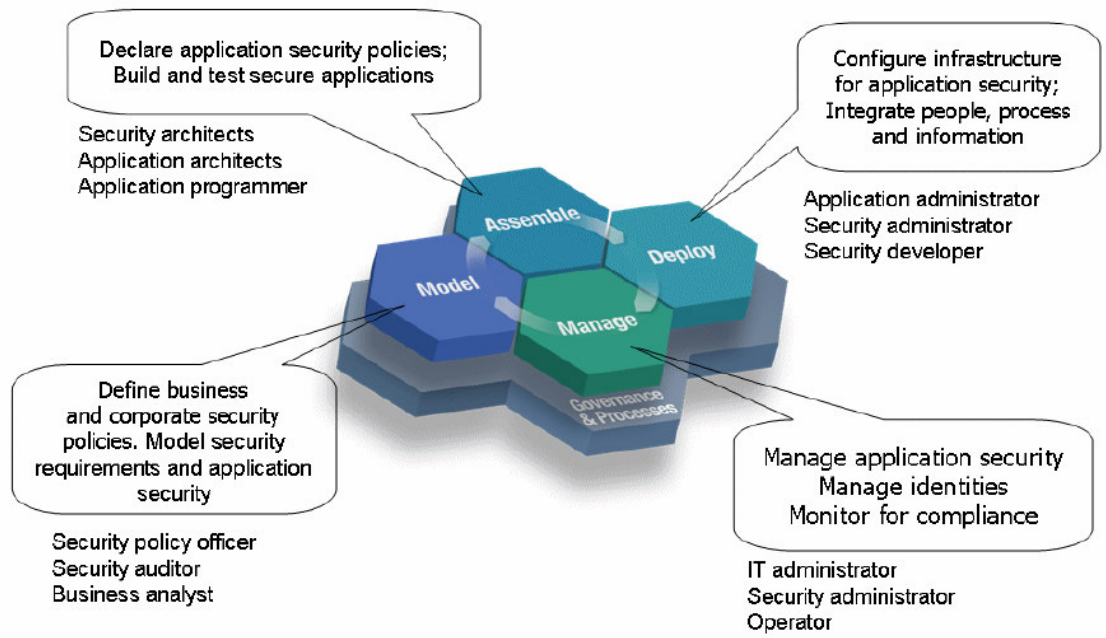
Possiamo assicurare l'integrità e la confidenzialità del messaggio?

Possiamo tracciare (Audit) l'accesso ai Web Services?

Possiamo garantire l'aderenza di un servizio a una normativa?

- necessità di **identificare** utenti e servizi e propagare queste identità attraverso l'organizzazione
- necessità di connettersi ad altre organizzazioni in **real-time e in modalità transazionale**
- necessità di assicurare un livello adeguato di **controllo** alla sicurezza dei singoli servizi e alle applicazioni composte, quando i servizi siano usati in combinazione
- necessità di **gestire** identità e sicurezza aldilà dei confini organizzativi, dei singoli sistemi e delle scelte tecnologiche di ciascuna organizzazione
- **protezione** dei dati a riposo e in transito
- necessità di dimostrare la **compliance** a vari livelli di regolamenti, leggi e normative

La sicurezza nel ciclo di vita di SOA



Modellazione

in questa fase vengono definiti i requisiti di sicurezza e le policies che i servizi devono rispettare

Assemblaggio

i servizi vengono implementati in modo aderente ai requisiti enunciati. Si effettuano i test necessari

Deployment

l'infrastruttura viene configurata in modo da rispettare la sicurezza richiesta dall'applicazione

Gestione

il servizio e l'accesso ad esso vengono gestiti quotidianamente in modo aderente ai requisiti di sicurezza e viene effettuato un monitoraggio appropriato e degli Audit di sicurezza periodici

Le minacce

Alle normali minacce presenti nel caso delle applicazioni web, si aggiungono quelle specifiche delle tecnologie utilizzate, in particolare XML, SOAP e WSDL.

- ***Attacchi al payloads/contenuto***
  Obiettivo Backend SQL Injection, BAPI (Business Application Programming Interface) protocol attack
  Obiettivo Utente Finale XSS, Malicious Active Content
- ***XML Misuse/Abuse*** Xpath Injection
- ***XML Structure manipulation*** Entity expansion, Referral attack
- ***WSDL attacks*** WSDL scanning, parameter tampering
- ***Infrastructure attacks*** Buffer overflow of Server and Universal Tunnel Misuse
- ***External or Secondary*** DNS poisoning for CA server

Le minacce

Tecnica	Descrizione	Protezione
Schema Poisoning	Manipolazione dello XML Schema per alterare il processamento dell'informazione	Utilizzare solo documenti WSDL e XML Schema da fonti fidate (autenticazione e firma digitale)
XML Parameter Tampering	Iniezione di scripts o contenuto maligni nei parametri di input	Validazione dei valori dei parametri per assicurarne la consistenza con le specifiche WSDL e lo Schema
Inadvertent XML DoS	Messaggi SOAP con codifica errata che causano il crash dell'applicazione	La Content Inspection assicura che i messaggi SOAP siano costruiti secondo le specifiche WSDL e XML Schema e le regole di intrusion prevention
WSDL Scanning	Lo Scanning dell'interfaccia WSDL può rivelare informazioni sensibili sui patterns di invocazione, le implementazioni tecnologiche soggiacenti e le vulnerabilità associate	Web services cloaking nasconde agli utenti la vera posizione dei web services
Oversized Payload	Consiste nello spedire messaggi di dimensioni eccessive per generare un attacco XDoS	Ispezionare il payload e bloccare gli elementi, i documenti e quant'altro con dimensione sovra-soglia
Recursive Payload	Spedizione di quantità massicce di dati nestati per creare un attacco XDoS nei confronti del parser XML	La Content inspection assicura che I messaggi SOAP siano costruiti in modo appropriato secondo le regole WSDL, XML Schema e le altre specifiche di sicurezza

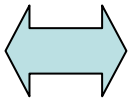
Le minacce

Technique	Description	Protection
XML Routing Detours	Redirezione di dati sensibili all'interno del path XML	La virtualizzazione WSDL specifica un routing stretto
SQL Injection	La SQL Injection permette di eseguire comandi direttamente sul database sottostante allo scopo di mostrare dati sensibili e modificarli	Filtraggio dei contenuti, ricerca di regular expression e tecniche di validazione dei dati
External Entity Attack	Attacco che sfrutta il fatto che un'applicazione ha un input XML da parte di una sorgente esterna non di fiducia	Sopprimere i riferimenti URI esterni per proteggersi da sorgenti esterne e istruzioni maligne e utilizzare solo input certificato
Malicious Code Injection	Scripts immersi in un messaggio SOAP potrebbero essere eseguiti direttamente su applicazioni e database oppure codice eseguibile, come virus, potrebbe essere posto in attach ai payloads SOAP	Content inspection degli attachments SOAP assicura la legittimità del contenuto dei messaggi come definita da WSDL, XML Schema e le policies di sicurezza del contenuto
Identity Centric Attack	Spoofing delle credenziali per accedere a dati sensibili	Stabilire autenticazione a livello di messaggio SOAP con auditing e logging attivi per analisi forense

XPATH Injection

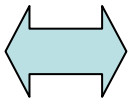
È una tecnica di attacco impiegata nei confronti di siti web che utilizzano l'input dell'utente per costruire query di tipo XPath. Concettualmente non è dissimile dal SQL Injection.
XPath (XML Path language) è un linguaggio utilizzato per effettuare ricerche all'interno di un documento XML. In particolare, XPath modella un documento XML come un albero formato da nodi. Ogni nodo è raggiungibile a partire dalla radice dell'albero (root indicata con /) seguendo un determinato cammino (path), come se fosse un file all'interno di una struttura di directories.

```
string(//utente[nome/text()='mrossi' and
password/text()='Padova_8']/pin/text())
```



```
Select pin from utente where
nome='mrossi' and password='Padova_8'
```

```
string(//utente[nome/text()='' or 1=1 or
''='' and
password/text()='pippo']/pin/text())
```



```
Select pin from utente where nome='' or
1=1 or ''='' and password='pippo'
```

Entity expansion o XML bomb

Questo tipo di attacco mira a ottenere un Denial of Service o un Buffer Overflow. Un esempio è dato dal seguente DTD (Document Type Definition), in cui viene definita un'entità ricorsiva "&x100;", che dovrebbe essere espansa nella ripetizione di 2^{100} occorrenze della parola "ciao", portando così rapidamente all'esaurimento della memoria del server. L'attacco può essere contrastato solo pre-processando il documento XML ed esaminando il DTD. D'altra parte c'è una mitigazione, e in particolare che SOAP proibisce di includere DTD, quindi l'attacco può riuscire in implementazioni non strettamente aderenti alle regole o per messaggi non SOAP

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE foobar [
<!ENTITY x0 "ciao">
<!ENTITY x1 "&x0;&x0;">
<!ENTITY x2 "&x1;&x1;">
<!ENTITY x3 "&x2;&x2;">
<!ENTITY x4 "&x3;&x3;">
 ...
<!ENTITY x98 "&x97;&x97;">
<!ENTITY x99 "&x98;&x98;">
<!ENTITY x100 "&x99;&x99;">
 ]>
<foobar>&x100;</foobar>
```

External entity attack/ Referral attack

I riferimenti a entità esterne permettono di includere in un documento XML dei dati che si trovano all'esterno, rispetto al documento stesso. Per fare questo, si dichiara nel DTD il riferimento esterno tramite la sintassi

```
<!ENTITY name SYSTEM "URI">
```

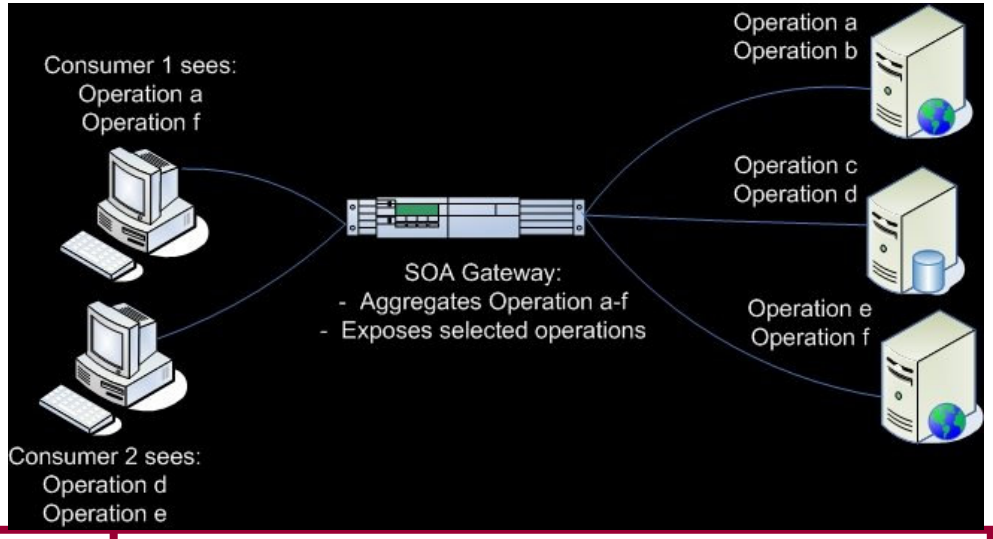
dove URI è la posizione della risorsa esterna.
Quando il processore XML deve validare un documento contenente un riferimento esterno, quest'ultimo deve prima essere sostituito dal testo che gli corrisponde. Questa operazione non viene, invece, necessariamente effettuata qualora il processore non stia facendo la validazione. L'attacco consiste nel costringere il processore XML a fare il parsing di dati che provengono dall'attaccante. In questo modo si può costringere l'applicazione ad aprire files arbitrari o connessioni TCP/IP. Gli obiettivi dell'attacco possono essere molteplici, dall'accesso a files riservati a DoS.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY>
  <!ENTITY xxe SYSTEM "file:///c:/boot.ini">
]>
<foo>&xxe;</foo>
```

il parsing del documento XML produce l'apertura del file boot.ini e l'inserimento del suo contenuto nel tag foo

Routing detour

In SOAP, non è possibile specificare la strada che verrà percorsa da un messaggio XML. Un eventuale attaccante potrebbe così modificare il percorso di instradamento del messaggio (routing), ponendosi in questo modo come man-in-the-middle e accedendo a informazioni potenzialmente di carattere riservato. Il problema viene affrontato usando ***WSDL virtualization***, con cui è possibile imporre un instradamento ben preciso. WSDL virtualization consiste nella capacità di creare un WSDL virtuale a partire da più files WSDL. Questo tipo di tecnica viene utilizzata quando si impieghi un'architettura in cui il Service Requester non parla direttamente con il Service Provider ma con un SOA Gateway, che si interpone tra essi, raccogliendo la richiesta del Client e aggregando i WSDL. In questo tipo di soluzione, eventuali policies vengono applicate al gateway. Inoltre, il SOA gateway potrebbe anche nascondere dettagli sensibili presenti nei WSDL originari.



Le contromisure

Le due organizzazioni che si sono occupate degli standards di sicurezza per i web services sono il **World Wide Web Consortium** (http://www.w3.org) e **Oasis Open** (http://www.oasis-open.org).
Si usa il concetto di sicurezza multilivello.
Il livello più basso, a cui applicare la sicurezza, è quello dell'***XML*** stesso, per il quale sono stati concepiti gli standards seguenti:

Message-level security

XML-encryption

supporta la criptazione sia di un intero documento che di singole porzioni, fino alla più piccola che è l'elemento. L'unico elemento obbligatorio è <CipherData> che contiene il dato criptato sia direttamente che come link esterno. L'elemento <EncryptionMethod> indica invece l'algoritmo di criptazione e la chiave, le cui informazioni stanno in <KeyInfo>.

XML Signature

permette di firmare dati XML e includere la firma risultante al documento stesso, che può essere firmato per intero o solo in porzioni singole. È anche possibile includere firme differenti per parti differenti del documento stesso. Una firma è composta di due elementi obbligatori, SignedInfo, che indica quali oggetti dati sono firmati, e SignatureValue, che è la firma vera e propria, intesa come digest criptato di SignedInfo.

XML-encryption

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>4019 2445 0277 5567</Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```

Cripto tutto l'elemento

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <EncryptedData Type='http://www.w3.org/2001/04/xmlenc#Element'
   xmlns='http://www.w3.org/2001/04/xmlenc#'>
    <CipherData>
      <CipherValue>A23B45C56</CipherValue>
    </CipherData>
  </EncryptedData>
</PaymentInfo>
```

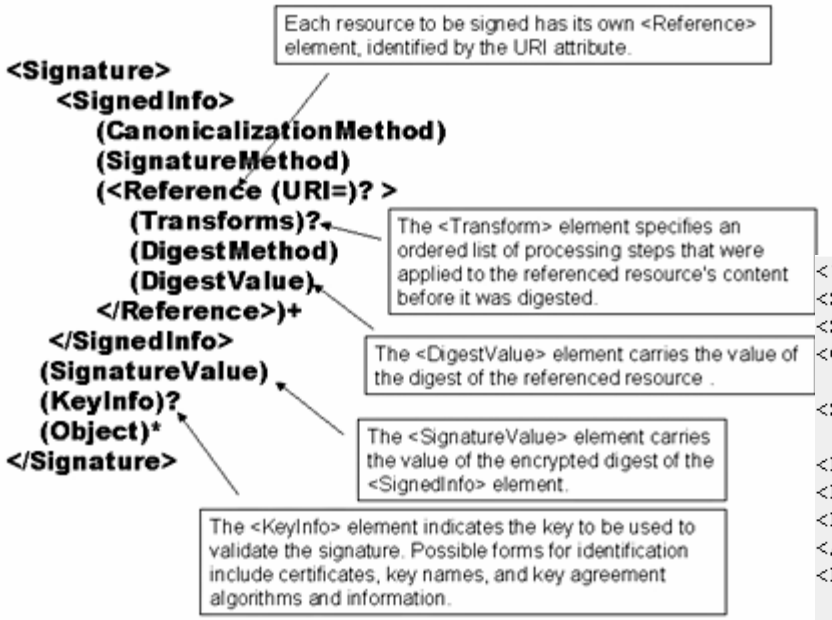
Cripto un blocco

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
     Type='http://www.w3.org/2001/04/xmlenc#Content'>
      <CipherData>
        <CipherValue>A23B45C56</CipherValue>
      </CipherData>
    </EncryptedData>
  </CreditCard>
</PaymentInfo>
```

Cripto un dato testuale

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
       Type='http://www.w3.org/2001/04/xmlenc#Content'>
        <CipherData>
          <CipherValue>A23B45C56</CipherValue>
        </CipherData>
      </EncryptedData>
    </Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```

XML-signature




```
<?xml version="1.0" encoding="UTF-8"?>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
<SignedInfo Id="foobar">
<CanonicalizationMethod
  Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
<SignatureMethod
  Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1" />
<Reference URI="http://www.abccompany.com/news/2000/03_27_00.htm">
<DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
<DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</DigestValue>
</Reference>
<Reference
  URI="http://www.w3.org/TR/2000/WD-xmldsig-core-20000228/signature-example
<DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
<DigestValue>UrXLDLBIta6skoV5/A8Q38GEw44=</DigestValue>
</Reference>
</SignedInfo>
<SignatureValue>MC0E~LE=</SignatureValue>
<KeyInfo>
<X509Data>
<X509SubjectName>CN=Ed Simon,O=XMLSec Inc.,ST=OTTAWA,C=CA</X509SubjectName>
<X509Certificate>
MIID5jCCAO+gA...lVN
</X509Certificate>
</X509Data>
</KeyInfo>

</Signature>
```

Le contromisure

Autenticazione e autorizzazione

SAML (Security Assertion Markup Language)

definisce una modalità standard per rappresentare le informazioni di autenticazione e autorizzazione. Ha l'obiettivo di superare le limitazioni dei cookies, che sono ristretti a un determinato dominio, ponendosi come strumento per condividere l'autenticazione tra più domini

XACML (eXtensible Access Control Markup Language)

permette di esprimere le policies di accesso e le autorizzazioni in XML. Definisce un vocabolario per specificare soggetti, diritti, oggetti e condizioni.

SPML (Service Provisioning Markup Language)

definisce un linguaggio standard per lo scambio di messaggi di provisioning come ad esempio per aggiungere, modificare o distruggere account utente, abilitare o disabilitare l'accesso, dare o revocare privilegi, cambiare password e altre cose ancora

XKMS (XML Key Management Specification)

ha l'obiettivo di fornire uno strato intermedio di astrazione tra una soluzione PKI e un applicativo, in modo da permettere all'applicativo stesso di appoggiarsi a implementazioni PKI diverse (X.509, PGP, SPKI, PKIX)

XrML (eXtensible Rights Markup Language)

fornisce un modo per esprimere le policies associate con un contenuto, se ad esempio un utente può usufruirne una volta o più, se può condividerlo o meno ecc.

DSML (Directory Services Markup Language)

permette agli sviluppatori di esprimere funzioni LDAP e recuperare i dati in formato XML

Approfondimento: SAML

Lo standard prevede l'esistenza di un client che faccia una richiesta, concernente l'identità di un soggetto, a una SAML Authority, che, a sua, volta, risponde tramite delle affermazioni (assertions) sul soggetto stesso.
Esistono 3 tipi di Authorities:

- ***authentication authorities***
- ***attribute authorities***
- ***policy decision points (PDPs)***

Le Authorities possono fare 3 tipi di affermazioni:

SAML authentication assertions

quando una SAML authentication authority riceve una richiesta di verifica delle credenziali di un certo soggetto, il risultato viene restituito tramite una affermazione di questo tipo che, di fatto, stabilisce che un certo utente si è autenticato in un certo dominio (es. azienda.com), utilizzando un certo mezzo (es. password) a un certo istante

SAML attribute assertions

una volta che le credenziali del soggetto sono state verificate, potrebbero venire richiesti a una SAML attribute authority una parte degli attributi (es. email, indirizzo, numero di codice fiscale ecc.) del soggetto stesso. Un attributo non è altro che una coppia nome, valore. Questi attributi vengono restituiti tramite un'affermazione di questo tipo che, di fatto, asserisce che un certo soggetto è associato a certi attributi aventi certi valori

SAML authorization assertions

è il tipo di affermazione che viene restituita da un PDP a fronte di una richiesta di verifica dei permessi di un certo soggetto su una determinate risorsa. Di fatto, essa asserisce che che un certo soggetto ha, o non ha, i permessi necessari per compiere una certa azione su una risorsa e ne fornisce l'evidenza

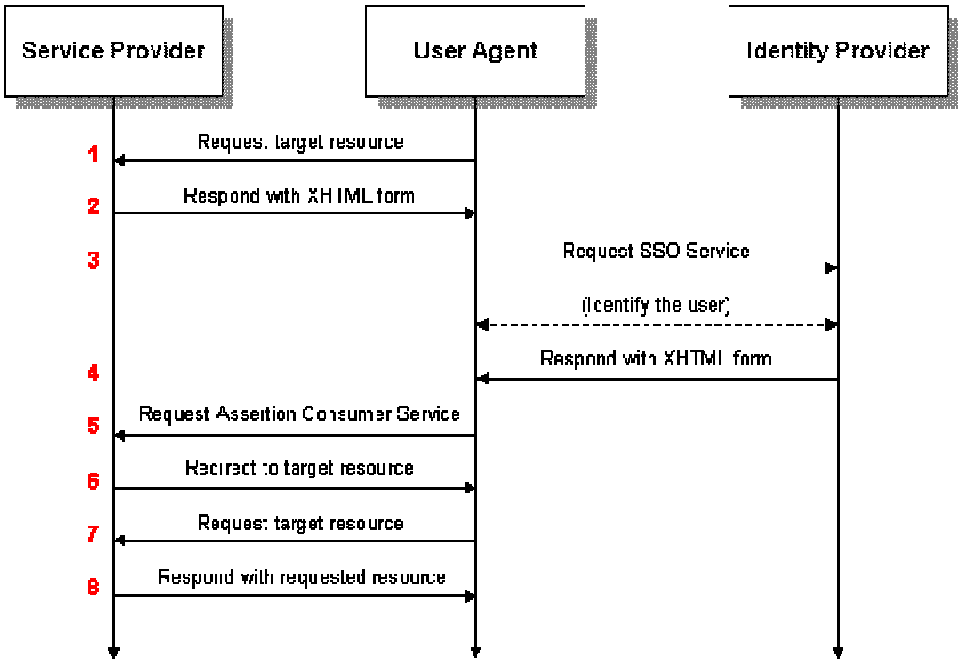
L'esempio seguente mostra una richiesta di attributi in SAML, in particolare il numero di telefono dell'utente Giovanni@esempio.com.

```
<samlp:Request ...>
     <samlp:AttributeQuery>
               <saml:Subject>
                 <saml:NameIdentifier
                 SecurityDomain="esempio.com"
                 Name="cn=Giovanni"/>
               </saml:Subject>
               <saml:AttributeDesignator
            AttributeName="telefono"
            AttributeNamespace="esempio.com"/>
     </samlp:AttributeQuery>
</samlp:Request>
```

In quest'altro esempio, invece, abbiamo la risposta ad una richiesta di autenticazione, tramite password, per lo stesso utente

```
<samlp:Response
   MajorVersion="1" MinorVersion="0"
   RequestID="192.128.1.123.8181881"
   InResponseTo="215.11.121.67.2254338"
   StatusCode="Success">
  <saml:Assertion
     MajorVersion="1" MinorVersion="0"
     AssertionID="215.11.121.67.2254338"
     Issuer="ITWellness spa"
     IssueInstant="2008-02-11T14:00:35Z">
     <saml:Conditions
        NotBefore="2008-02-11T14:00:35Z"
        NotAfter="2008-02-11T14:20:35Z" />
     <saml:AuthenticationStatement
          AuthenticationMethod="Password"
          AuthenticationInstant="2008-02- 11T14:00:35Z">
          <saml:Subject>
             <saml:NameIdentifier
              SecurityDomain="esempio.com"
              Name="cn=Giovanni" />
          </saml:Subject>
    </saml:AuthenticationStatement>
   </saml:Assertion>
 </samlp:Response>
```

SAML



1. L'utente richiede una risorsa, ad es. https://www.sito.com/risorsa
2. Il service provider riconosce che per accedere alla risorsa è necessario essere autenticati e risponde con una form XHTML, che impacchetta una richiesta SAML passata in POST all'Identity Provider

```
<form method="post"
     action="https://idp.example.org/SAML2/SSO/POST" ...>
     <input type="hidden" name="SAMLRequest"
     value="request"/>
 ...
    <input type="submit" value="Submit" />
</form>
```

3. L'Identity Provider, dopo aver ricevuto la richiesta, autentica l'Utente
4. L'Identity provider crea una form XHTML in cui è contenuta la risposta SAML

```
<form method="post"
     action="https://sp.example.com/SAML2/SSO/POST" ...>
<input type="hidden" name="SAMLResponse" value="response"
     />
 ...
<input type="submit" value="Submit" />
</form>
```

5. ...e la passa in POST, tramite il browser agent dell'utente, al Service Provider
6. A questo punto l'utente viene rediretto verso la risorsa che aveva chiesto originariamente
7. L'utente richiede nuovamente la risorsa https://www.sito.com/risorsa e dal momento che adesso esiste il necessario contesto di sicurezza...
8. La risorsa viene erogata

Approfondimento: XACML

XACML (eXtensible Access Control Markup Language) è uno standard, basato su XML, per descrivere e condividere policies di accesso. Le policies possono esprimere regole complesse, basate sulla combinazione booleana di regole più semplici che sono basate su:

- attributi posseduti dall'utente - come ad esempio l'appartenenza ad un certo dipartimento
- ora del giorno – ad esempio l'accesso a un sistema è permesso solo dalle 8 del mattino alle 6 del pomeriggio
- tipo di azione – ad esempio la lettura piuttosto che la scrittura di un documento
- il meccanismo di autenticazione – ad esempio è permessa solo l'autenticazione con certificato
- il tipo di protocollo usato per accedere alla risorsa – ad esempio permettere HTTPS ma non HTTP

In questo modo si possono costruire policies del tipo:
"Possono aggiornare il documento tutti gli utenti dei dipartimenti Marketing e Commerciale, che si autenticano con smart card dalla rete locale e non da remoto. L'aggiornamento è previsto nella fascia oraria 15-18, in cui non ci sono sistemi attivi che stanno processando il documento".

Approfondimento: SPML

SPML (Service Provisioning Markup Language) è uno standard basato su XML per esprimere richieste e risposte di attività di provisioning, come la creazione o la rimozione di identità.
Lo standard prevede l'esistenza di 3 ruoli:

- ***Requesting Authority (RA)***
  è l'entità che fa la richiesta di Provisioning
- ***Provisioning Service Provider (PSP)***
  è il servizio che risponde alla richiesta della RA. Deve essere in grado di capire SPML e, di fatto, è un server di front-end che, da una parte si interfaccia con la RA, cioè il client, comunicandoci in SPML, e dall'altra con il PST, parlandoci con un linguaggio anche diverso
- ***Provisioning Service Target (PST)***
  è l'entità che esegue effettivamente il provisioning. E' un server di back-end e non deve necessariamente "parlare" SPML. Ad esempio, il PST potrebbe essere un server LDAP o un database, o anche un server di posta su cui deve venire creata la casella di posta dell'utente

Un esempio di provisioning potrebbe essere il seguente. Un utente (RA) si registra su un portale WEB (PSP). Questo scatena un certo numero di operazioni sui server (PST) appartenenti al portale, come la creazione di un'utenza sul server LDAP, la creazione di una casella di posta accessibile da web, la creazione di uno spazio disco riservato per permettere all'utente di fare l'upload da internet dei propri files, la creazione di un server web virtuale, tipo myspace, o la creazione di un blog, e così via.

SPML

A richiesta....

```
<addRequest>
      <attributes>
        <attr name="objectclass">
          <value>urn:...:SPML:interop:interopUser</value>
        </attr>
        <attr name="cn">
          <value>Mario Rossi</value>
        </attr>
        <attr name="mail">
           <value>Mario.Rossi@esempio.com</value>
        </attr>
        <attr name="description">
          <value>Attivazione utenza Mario Rossi</value>
        </attr>
        <attr name="memberLevel">
          <value>2</value>
        </attr>
        <attr name="company">
          <value>Esempio spa</value>
        </attr>
        <attr name="registrationTime">
          <value>22-Jul-2008 22:05 </value>
        </attr>
      </attributes>
 </addRequest>
```

..il PSP risponde

```
<addResponse result = "urn:...:SPML:1:0#success">
      <identifier type= "urn:...:SPML:1:0#EMailAddress">
        <id> Mario.Rossi@esempio.com</id>
      </identifier>
      <attributes>
        <attr name="mailBoxLimit">
          <value>2500MB</value>
        </attr>
      </attributes>
</addResponse>
```

Le contromisure

Il livello successivo sono i messaggi SOAP. In tal caso gli standards di sicurezza originati dal lavoro di OASIS prendono il nome collettivo di ***WS-**** (Web Services Specification). Essi rappresentano estensioni di SOAP e, in particolare:

WS-Security

definisce il meccanismo con cui includere integrità, autenticazione e confidenzialità nel singolo messaggio SOAP. Fa uso di XML Encription e XML Signaure e definisce come includere dati firmati, firme digitali e message digests in un messaggio SOAP. Fornisce uno strato di sicurezza di base su cui appoggiano servizi più evoluti. Il vantaggio rispetto all'uso, ad esempio di SSL, è che WS-Security protegge il messaggio end-to-end, indipendentemente da quanti servizi vengano chiamati strada facendo e da quante organizzazioni siano coinvolte. Inoltre, esso è indipendente dal protocollo di trasporto di SOAP, che sia HTTP o meno.

WS-Trust

il Trust riflette le relazioni di Business. Viene utilizzato tra due parti che usano WS-Security per lo scambio sicuro dei messaggi, per stabilire un accordo sui meccanismi di sicurezza da impiegare durante la conversazione. Essenzialmente, per assicurare l'interoperabilità tra i tokens di sicurezza utilizzati da due diversi domini (es. username/password in uno e SAML nell'altro), si impiega un Security Token Service (STS) che fa da traduttore/Gateway tra i due

Le contromisure

Soap Message to send

```
<Envelope>

  <Header/>

  <Body Id=2>
    <StockQuoteRequest>
      <symbols>

        <Symbol>“SAP"
        <Symbol>“ORACLE"

</Envelope>
```

Soap Message after addition of Security Header

```
<Envelope>
  <Header>
    <Security>
      <UsernameToken Id=1>
        <Username>"Alice"
        <Nonce>"mTbzQM84RkFqza+lIes/xw=="
        <Created>"2004-09-01T13:31:5
      <Signature>
        <SignedInfo>
          <SignatureMethod Algorithm=hmac-sha1>
          <Reference URI=#2>
            <DigestValue>"U9sBHidIkVvKA4vZo0gGKxMhA1g="
        <SignatureValue>"8/ohMBZ5JwzYyu+POU/v879R01s="
          <KeyInfo>
            <SecurityTokenReference>
              <Reference URI=#1 ValueType=UsernameTo
  <Body Id=2>
    <StockQuoteRequest>
      <symbols>
        <Symbol>"SAP"
        <Symbol>"ORACLE"
```

UsernameToken assumes both parties know Alice’s secret password *p*

<Security> header defined by OASIS **WS-Security** includes identity tokens, signatures, encrypted message parts

Each **DigestValue** is a cryptographic hash of the URI target

hmacsha1(*key*, **SignedInfo**) where *key*≈*psha1*(*p*+*nonce*+*created*)

Le contromisure

WS-Policy

descrive le caratteristiche di qualità di un servizio e i suoi requisiti di sicurezza, come ad esempio se il servizio richiede lo scambio cifrato dei dati o meno. Le policies dovrebbero proteggere i dati siano in transito che a riposo.

WS-Privacy

WS-Federation

fornisce il supporto per la propagazione sicura, attraverso Internet, delle informazioni di identità, autenticazione e autorizzazione.

WS-Authorization

WS-SecurityPolicy

descrive come rendere sicuri i messaggi lungo un certo canale di comunicazione. Sono le regole di sicurezza che devono essere soddisfatte per poter accedere al servizio. Quando più servizi vengono combinati potrebbe essere necessario rivedere le policies di sicurezza, in quanto un utente potrebbe avere diritto ad accedere a due servizi separatamente, ma non alla loro combinazione, ad esempio per poter prevenire attacchi di tipo data mining e data inferring. Ogni nuova coreografia di servizi, dovrebbe così essere separatamente valutata, indipendentemente dai diritti di accesso dei servizi componenti.

WS-SecureConversation

definisce la creazione e la condivisione di un contesto di sicurezza tra due parti che comunicano tra loro. Il contesto è basato su un Security Context Token (SCT), che coinvolge l'impiego di una chiave segreta comune per cifrare e/o firmare i messaggi

Le contromisure

WS-Attribute service

WS-ReliableMessaging

è utilizzato dai servizi per avere la garanzia del corretto delivery dei messaggi e per stabilire il giusto ordine di processamento

WS-Addressing

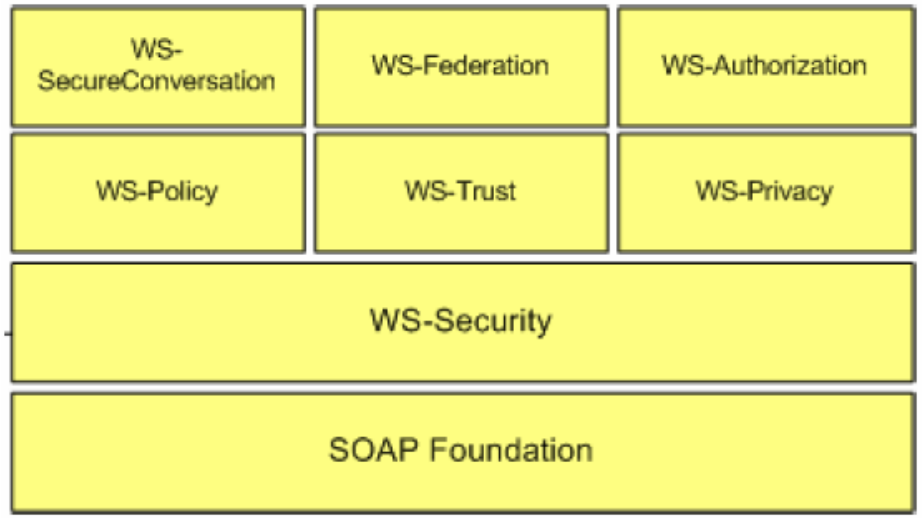
definisce gli headers applicati ai messaggi SOAP per determinare se si necessita o meno di una risposta e per stabilire delle correlazioni tra i messaggi

WS-MetadataExchange

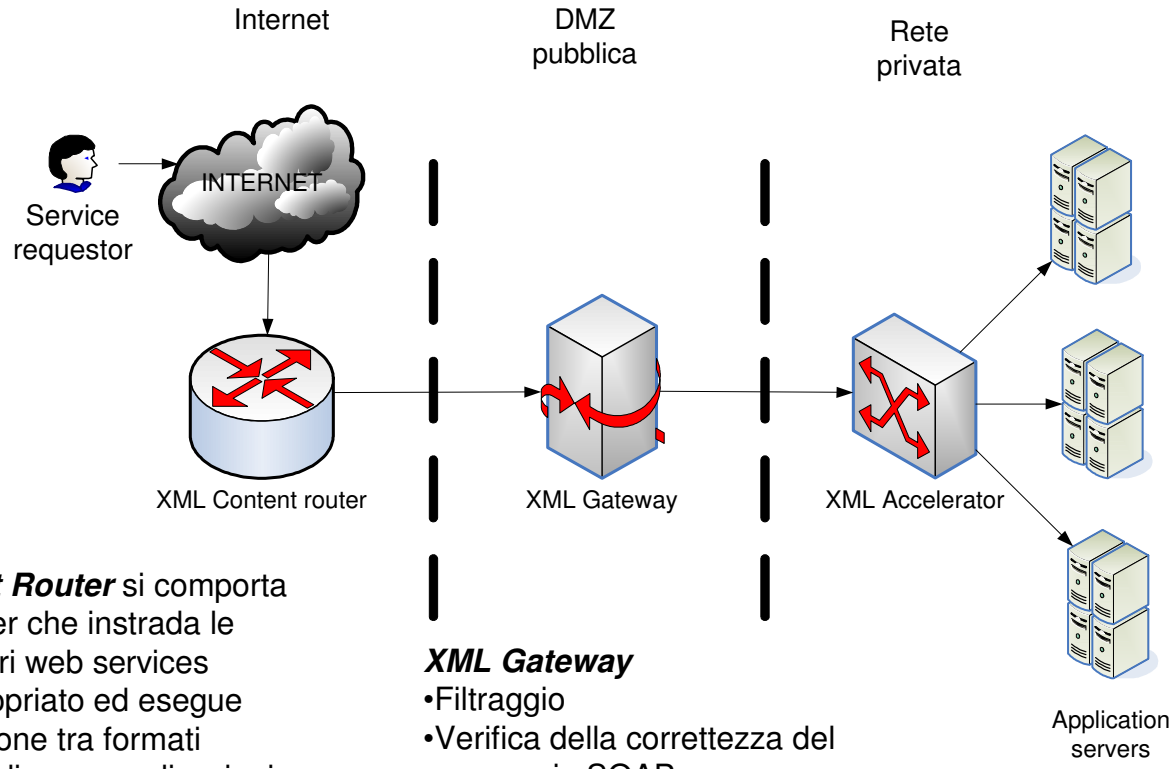
è un protocollo di handshake che permette agli utenti di recuperare i documenti WSDL e WS-Policy associati ad un servizio

WS-Provisioning

WS-BaseNotification



Le architetture di sicurezza



XML Content Router si comporta come un router che instrada le richieste ai vari web services laddove appropriato ed esegue anche traduzione tra formati eterogenei di diverse applicazioni. Un'altro compito che può assolvere è quello di terminatore di canali SSL/TLS.

XML Gateway
- Filtraggio
- Verifica della correttezza del messaggio SOAP
- Autenticazione/Autorizzazione
- Firma e controllo
- Criptazione e decriptazione
- Propagazione delle credenziali
- Virtualizzazione

XML Accelerator è in grado di processare molto velocemente (Wire Speed) XML/XSLT, con impatto positivo sulle performances e sulla disponibilità dei servizi. Inoltre, può assolvere anche il compito di validare lo Schema, eventualmente scartando il traffico con strutture e contenuti malformati o sospetti.

Le architetture di sicurezza

ROUTERS, FIREWALLS, ETC.

CONTENT SWITCHES; DEEP PACKET INSPECTION

ONLY XML-AWARE NETWORK DEVICES PARSE, PROCESS AND TRANSFORM XML

IP ADDRESS INFORMATION

LAYER 4 INFORMATION

```
<xml? versioin= "1.0" encoding = "UTF-8"?>
<Catalog>
  <Book>
    <Title>XML Proxies</Title>
    <Author>Ron Schmelzer</Author>
    <Price>$199.99</Price>
</Book>
```

SOAP MESSAGE

HTTP HEADERS

SOAP ENVELOPE

SOAP HEADER

HEADERS

SOAP BODY

MESSAGE NAME & DATA

XML EXPOSES KEY INFORMATION IN CLEAR TEXT

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

<SOAP: Header>
<t: Transaction xmlns: t="www.example.org" SOAP: mustUnderstand="1">
</t: Transaction>
</SOAP: Header>

<SOAP: Body>
<m: Deposit xmlns: m=" www.example.org">
<m: amount>200</m: amount>
</m: Deposit>
</SOAP: Body>

</SOAP: Envelope>
```

Il modello di maturità di SOA

Alcune soluzioni di mercato

La soluzione **Oracle** per SOA prende il nome di Oracle SOA Suite ed è composta dai seguenti elementi:

Oracle BPEL Process Manager
il primo motore BPEL (Business Process Execution Language) nativo per l'orchestrazione dei servizi Web, che fornisce alle aziende la capacità di progettare, implementare e attuare nuovi processi di business

Oracle Enterprise Service Bus
un prodotto standard che collega i sistemi IT e i business partner sotto forma di set di servizi

Oracle Web Services Manager
che fornisce una console unificata per definire e applicare policy relative ai servizi Web

Oracle Business Rules Engine
che fornisce gli strumenti per definire e gestire le regole di business

Oracle Business Activity Monitoring
che assicura la visibilità in tempo reale sulle attività di business

Oracle Enterprise Manager
per implementare e gestire le applicazioni orientate ai servizi all'interno di un ambiente operativo

Oracle JDeveloper
è un ambiente di sviluppo integrato per creare e comporre applicazioni, e che può fungere anche da toolset unificato per tutti i componenti di Oracle SOA Suite.

Oracle fornisce anche un'ampia gamma di connettori per l'accesso a sistemi legacy quali CICS, VSAM e IMS nonché servizi Web per accedere ad applicazioni pacchettizzate come SAP, PeopleSoft, Oracle e-Business Suite e Siebel.

Alcune soluzioni di mercato

Gli elementi che costituiscono l'offerta di mercato di **IBM** relativamente a SOA sono:

IBM Tivoli Federated Identità Manager
supporta la gestione delle identità e fornisce all'utente un accesso semplificato a informazione e servizi

IBM Websphere Data Power SOA Appliances
sono perifieriche di rete con compiti specifici. Ad esempio:

WebSphere DataPower XML Security Gateway XS40
è l'XML Gateway

WebSphere DataPower XML Accelerator XA35
è l'XML Accelerator

WebSphere DataPower Integration Appliance XI50
è l'XML Content Router

IBM Tivoli Composite Application Manager for SOA
facilita la gestione dei servizi scoprendo, monitorando e gestendo il flusso dei messaggi

IBM Tivoli Composite Application Manager for Websphere
è una soluzione di gestione delle applicazioni. Fornisce alta affidabilità e performances per i vari server di un ambiente SOA

Alcune soluzioni di mercato

La soluzione SOA di **BEA** si esprime attraverso 3 famiglie di prodotti:

BEA Aqualogic suite

BEA Aqualogic User Interaction è il componente che permette di creare portali di tipo enterprise, comunità collaborative e applicazioni composte a partire dai singolis ervizi disponibili sulla rete

BEA Aqualogic BPM suite è il componente per modellare, gestire eseguire, automatizzare e misurare I processi di business

BEA Aqualogic Service Bus

BEA Aqualogic Integrator aiuta ad integrare le applicazioni esistenti e a razionalizzare i processi IT, automatizzandoli ed esponendoli, allo scopo di facilitarne il riuso attraverso l'organizzazione

BEA Aqualogic Data services Platform è il componente che fornisce servizi di mediazione e astrazione tra dati di tipo diverso e di origine differente

BEA Aqualogic Registry Repository è uno dei componenti essenziali per la Governance di SOA ed è composto da un UDDI registry integrato con altre funzionalità per la misura, il monitoraggio e la gestione dei servizi e degli asset dell'architettura SOA. Combina insieme le funzionalità di Service Registry con quelle di Enterprise Repository

BEA Aqualogic Service Registry

BEA Aqualogic Enterprise Repository

BEA Aqualogic SOA Management è la soluzione per il monitoraggio e la gestione dei servizi

BEA Aqualogic Enterprise Security permette di esternalizzare , unificare e semplificare la gestione delle policies di sicurezza integrando l'Identity Management

BEA Aqualogic Commerce Services è la soluzione che permette di costruire portali di e-Commerce partendo dall'integrazione dei singoli servizi sulla rete, dalal gestioen del magazzino, al CRM, all'ERP ecc.

BEA Aqualogic Pathways aiuta l'utente a cercare, estrarre e assemblare l'informazione da diverse sorgenti
BEA Aqualogic Pages è un Web page authoring system
BEA Aqualogic Ensemble è un framework per la gestione delle risorse web allo scopo, ad esempio, di creare widgets e mash-up

BEA Weblogic suite

BEA Weblogic Portal è la soluzione di BEA per la realizzazione di portali capaci di integrare servizi diversi, in una logica Web 2.0, come ad esempio mash-up e interfacce utente ricche (Ajax)
BEA Weblogic Integration è il prodotto per l'integrazione di serevr, applicazioni e dati
BEA Weblogic Server è l'application server di BEA

BEA Tuxedo suite

è lo strumento per il monitoraggio e il processamento transazionali di applicazioni in C, C++ e Cobol

Alcune soluzioni di mercato

La soluzione SOA di **Microsoft** è basata su ***Biztalk server e services***.

Su questo si appoggiano prodotti e servizi sviluppati secondo le logiche SaaS e S+S.
È dotato di un buon numero di adapters e si integra nativamente con EDI e RFID.
La soluzione di Identity e Policy Management è Microsoft Active Directory.
Dal punto di vista dello sviluppo, è integrato con .NET framework e si appoggia a Visual Studio per la realizzazione di applicazioni, per quanto tramite l'interfaccia del prodotto è possibile gestire l'orchestrazione e l'integrazione dei servizi, nonché gestire i processi.

Alcune soluzioni di mercato

Il prodotto di riferimento si **SAP** è ***SAP Netweaver***. È composto da:

SAP Auto-ID Infrastructure
integrazione con RFID

SAP Business Intelligence
è il componente per l'integrazione delle informazioni aziendali

SAP Enterprise Portal
attraverso un solo punto d'accesso e con modalità di fruizione personalizzate per le diverse funzioni aziendali, consente agli utenti di accedere a informazioni provenienti da applicazioni SAP e di terze parti, da data warehouse e documenti individuali, da contenuti e Web interni ed esterni e strumenti di collaborazione.

SAP Exchange Infrastructure è il prodotto per l'integrazione e l'orchestrazione dei processi

SAP Master Data Management
permette il consolidamento, l'estrazione, la consistenza, la sincronizzazione, la distribuzione e la gestione di dati diversi provenienti da sorgenti diverse

SAP Mobile Infrastructure
è il componente per l'erogazione di servizi in mobilità

SAP Web Application Server è l'application server, al cuore delle applicazioni web e dei web services

SAP Identity Management

SAP Netweaver Developer Studio
è l'IDE (Integrated Development Environment) usato per lo sviluppo di applicazioni J2EE multilivello. È integrato con Web DynPro per la realizzazione dell'interfaccia utente

SAP NetWeaver Composition Environment
permette di sviluppare applicazioni composte a partire da ambienti eterogenei e servizi distribuiti

Bibliografia e sitografia :

1. “Architetture Enterprise”, A. Sinibaldi, Infomedia, 2008
2. "Building Trustworthy Semantic Webs", B. Thuraisingham, Auerbach Publications, 2007
3. http://soasecurity-ajw.blogspot.com/
4. http://soasecurity.net/
5. http://www.innoq.com/soa/ws-standards/poster/innoQ%20WS-Standards%20Poster%202007-02.pdf
6. “SOA security”, R. Kanneganti, P. A. Chodavarapu, Manning Publications, 2007
7. http://www.cgisecurity.com/ws/
8. http://www.owasp.org/index.php/Category:OWASP_Web_Services_Security_Project

Grazie a tutti per la cortese attenzione!

Domande?

alessandro_sinibaldi@virgilio.it