FAKULTET ZA POSLOVNU INFORMATIKU

# Bezbednost Servisno-Orijentisane Arhitekture

**- Diplomski rad -**

Beograd, 2009.

Univerzitet Singidunum

FAKULTET ZA POSLOVNU INFORMATIKU

# Bezbednost Servisno-Orijentisane Arhitekture

**- Diplomski rad -**

Mentor:
Prof. dr xxxx xxxx

Student:
Dragan Lukač

Br. indeksa:
III-xx/2xxx

Univerzitet Singidunum
Beograd, 2009.

FAKULTET ZA POSLOVNU INFORMATIKU

**UNIVERZITET SINGIDUNUM**

**FAKULTET ZA POSLOVNU INFORMATIKU**

**Beograd, Danijelova 32**

Broj: _____/2009

**Kandidat:** xxx xxxx

**Broj indeksa:** III-xx/2xxx

**Smer:** Programiranje i projektovanje

**Tema:** Bezbednost Servisno-Orijentisane Arhitekture

**MENTOR**

_____

Prof. dr xxxx xxxx

**DEKAN**

**Datum odobrenja teme:**
Beograd

_____

Prof. dr xxxx xxxx

# Sažetak

Cilj ovog rada je da objasni i približi pojam SOA filozofije kao i pristupe projektovanju SOA sistema. Objasni funkciju i svrhu veb servisa. Pružen je kratak uvid u istoriju ideje, aplikacije kao servisa koji je dostupan što širem broju ljudi. Predstavljene su komponente i objašnjena je njihova svrhu i načini rada kao dela SOA sistema. Date su ključne tačke u osmišljavanju i projektovanju SOA sitema, kao i koji su glavni problemi i zablude u ovom procesu. Akcenat je stavljen na bezbednost kroz ceo proces osmišljavanja, projektovanja, testiranja i primene kao i daljeg napretka i nadgradnje SOA sistema, uz promišljeno planiranje i poštovanje regulative. Data je kratka studija slučaja kao demonstracija, ispravnog i pogrešnog, korišćenja i odabira tehnogija pri projektovanju sistema bezbednosti. Za kraj je istaknuta uloga i važnost sistema upravljanja.

Svrha svega gorenavedenog je da pokaže da SOA nije pitanje trenda ili hira za promenom sistema, već neophodnost daljeg napretka i unošenja reda u haotični, glomazni i preskupi način funkcionisanja IT-a.

# *Abstract*

*Goal of this paper is to explain and to get us a closer look at the term of SOA pholisophy as well as aproaches to SOA design. . Explain the end function and purpose of Web Services. A short history insight of idea, application as a servise that is approachable to a wide number of people, is given. Components and explanations of their purpose and way of functioning, as part of SOA, are presented. Key points in rationalization and design of SOA, as well as major problems and delusions in this process are given. Accentuation is on security throught the whole process of rationalization, design, testing and appliance as well as furher progress and upgrade of SOA, along with well thought planing and the respect of regulations. A Short case study is given as a demonstration of right and fault use and selection of technology in the process of security system design. For the end the role of SOA governance and the importance of this system is noted.*

*The purpose of upstate is to show that a matter of SOA is not a trend or a question of desire for a system change, but a necesity for further progress and introduciton of a order in chaotic. overgrown and too expensive way of working of IT.*

# Sadržaj

# 1.Uvod

U današnje vreme teško je zamisliti poslovanje bez dobrog informacionog sistema (IS). Svaki informacioni sistem sastoji se od mrežne infrastrukture, klijentskih i serverskih računara, aplikacija i operativnih sistema koji se izvršavaju na tim računarima i baza podataka. Informacione tehnologije (IT) su unele mnoga poboljšanja u poslovne procese ali kako se poslovanje širi, tako i informacioni sistem postaje sve razuđeniji, pa ukoliko nije dobro isplaniran i fleksibilan, poslovanje počinje da gubi na efikasnosti.

Ključni problem prepoznat je u činjenici da poslovni procesi danas isuviše zavise od ograničenja IT-a i teško mogu da se međusobno povežu u integrisaniji sistem. Zbog toga je bilo neophodno jasno odvojiti IT i poslovanje i uskladiti ih da rade zajedno.

SOA (Service Oriented Architecture) je novi pristup pri projektovanju informacionih sistema od starih i već postojećih resursa. Sam cilj SOA-e je da odvoji poslovne procese od IT-a i da IS učini fleksibilnim i dovoljnim da ispuni sve zahteve modernih tokova poslovanja. Kao što ćemo videti u ovom radu, SOA nije potpuno nova ideja, ona predstavlja početak realizacije jedne ideje koja ima za cilj da spoji ceo IT sektor u jednu celinu i potčini ga interesima poslovanja. Kao relativno mlada, u praktičnoj primeni, ova filozofija zahteva izvesnu edukaciju i upoznavanje sa terminologijom, tehnologijom, metodama i pristupima izrade ovakvog sistema.

# 2.SOA Istorijat

Da bi bilo šta rekli o SOA istorijatu moramo se prvo osvrnuti na XML i Web Servise. Moglo bi se reći da je SOA nastala evolutivnim procesom iz ove dve tehnologije, koje su svojom pojavom i filozofijom otvorile mnoge mogućnosti.

Kao i HTML (Hypertext Markup Language), XML (Extensible Markup Language) je napravljen od strane W3C organizacije i izveden je iz SGML (Standard Generalized Markup Language) jezika, koji postoji još od sedamdesetih godina. Ovaj meta jezik je omogućio organizacijama da sirovim podacima daju smisao i razmenjuju obeležene (*tagged*) podatke sa drugim organizacijama.

XML je razvijen kao odgovor na eBiznis pokret, koji je nastao kasnih devedesetih. Korišćenjem XML-a omogućeno je informacijama, koje se šalju internet protokolima, dati smisao i kontekst. XML nije samo poslužio za standardizaciju podataka već je kao jezik iskorišćen za razvoj dodadnih standarda, XSD(*XML Schema Definition Language*) i XSLT (XSL Transformation Language). Pored XML jezika ova dva standarda su danas ključni delovi XML tehnologije.

Ova arhitektura predstavljanja podataka je bazni sloj SOA tehnologije. U okviru nje, XML određuje format i strukturu poruke koja se kreće između servisa. XSD šeme čuvaju integritet i validnost podataka poruke i XSLT se, kroz šematsko mapiranje koristi da bi se omogućila komunikacija između različitih formata podataka.

Dave Winer, Don Box, Bob Atkinson i Mohsen Al-Ghosein su uz pomoć Microsoft-a 1998. godine dizajnirali SOAP(*Simple Object Acces Protocol*). W3C je 2000. godine usvojio SOAP standard. Prevashodno je zamišljen da ujedini RPC(*Remote Procedure Call*) komunikaciju. Ideja je bila da se podaci, koji razmenjuju komponente serijalizuju pomoću XML-a, transportuju i onda deserijalizuju u prvobitni format. Uskoro su korporacije i prodavci softvera uvideli ogroman potencijal za unapređenje eBiznis tehnologije kreiranjem okvira za komunikaciju koji nema vlasništvo. Ovo je dovelo do ideje da se stvori distribuirana tehnologija bazirana na internet-u, koja ne koristi samo postojeće internet protokole već pruža standardizovani okvir komunikacije da bi se premostile razlike u komunikacije u okviru i među organizacijama. Ovaj koncept je nazvan WEB Servisi.

Najbitniji deo WEB Servisa je javni interfejs koji on pruža, to je ono što pruža servisu identitet i omogućava njegovo pozivanje. Baš zato je jedna od prvih inicijativa, za podršku tehnologije WEB Servisa, bila WSDL(*Web Service Definition Language*). W3C je 2001. godine primila specifikaciju za WSDL i nastavila je da razvija ovaj standard. WSDL datoteka je centralni deo informacije koja opisuje veb servis.

Posledjni u nizu standarda, veb servisa prve generacije, je UDDI(*Universal Description Discovery and Integration*) specifikacija. Razvijena je od strane UDDI.org, kasnije predata OASIS-u(*Organization for the Advancement of Structured Information Standards*), koji je nastavio da je razvija u saradnji sa UDDI.org. Ova specifikacija omogućava kreiranje standardizovanih registara za opis servisa, u okviru i van organizacije. ovo pruža mogućnost registracije veb servisa na centralizovanoj lokaciji gde ih mogu otkriti korisnici.

Kako je interesovanje za veb servise raslo, najveće softverske kompanije su užurbano dodavale nove specifikacije. Ove je poznato kao druga generacija veb servis standarda, ove specifikacije su se pre svega odnosile na konkretne oblasti funkcionalnosti, sa konačnim ciljem dovođenja tehnologije veb servisa na preduzetnički nivo.

Uskoro su organizacije počele da shvataju da veb servisi, pored prilagođavanja postojećih distribuiranih aplikacija, mogu postati osnova za potpuno novu platformu, koja bi omogućila koncept deljenja poslovanja na seriju autonomnih servisa. Tako je nastao koncept koji danas zovemo SOA.

# 3.Definisanje SOA-e

Dati jasnu, konciznu i lako razumljivu definiciju SOA-e predstavlja pravi poduhvat, mnogi autori koje se bave ovom tematikom su dali svoje definicije.

Juric i saradnici u svojoj knjizi, *SOA Aproach to Integration,* definišu SOA kao "preduzetničku arhitekturu gde se dizajniraju aplikacije koje pužaju *krupnozrne* servise, koje koriste poslovni procesi ili druge aplikacije. SOA je dizajnerski koncept i arhitektura" (Juric i saradnici, 2007).

U knjizi, *The New Language of Business SOA & WEB 2.0,* se kaže da je "SOA pristup IT arhitekturi podstaknut poslovanjem, koji podržava integraciju poslovanja u povezane i ponovljive poslovne zadatke i servise" (Carter, 2007).

Autori knjige, *Service Oriented Architecture for Dummies,* definišu SOA kao "softversku arhitekturu za pravljenje aplikacija koje implementiraju poslovne procese ili servise koristeći skup labavo povezanih crnih kutija kao komponente usklađene da pruže zadovoljavajući nivo usluge" (Hurwitz i saradnici, 2007).

Čak nam i *WIKIPEDIA* nudi svoju definiciju pojma SOA. "Možemo definisati SOA kao grupu servisa koji međusobno komuniciraju. Proces komunikacije može biti jednostavno razmenjivanje podataka ili koordinisanje dva ili više servisa u izvrašavanju neke aktivnosti.

Međusobna komunikacija ukazuje na to da su nam neophodna sredstva za međusobno povezivanje dva ili više servisa".

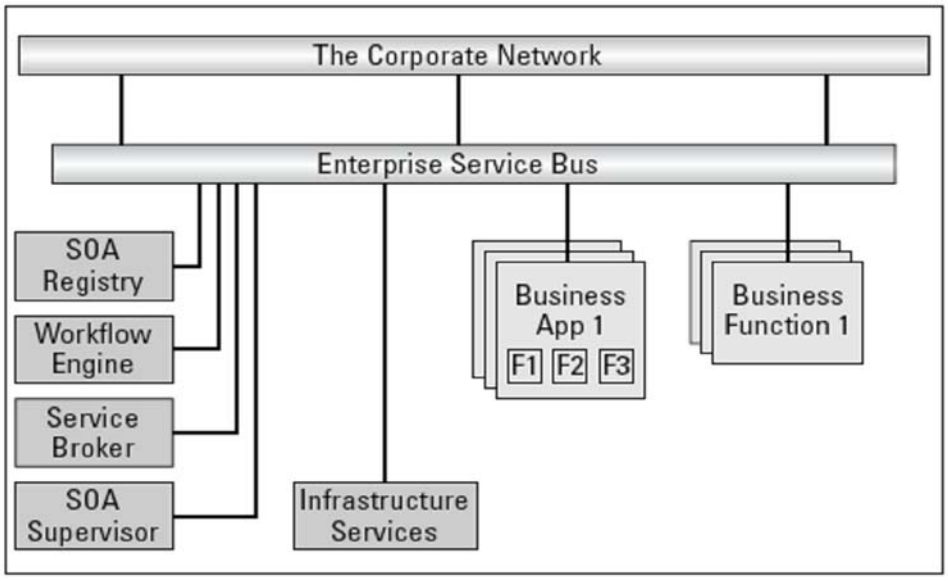
Sve ove definicije su ispravne i manje više razumljive, sve ukazuju na donekle slične stvari. SOA je arhitektura koja je tu da pospeši poslovanje razlaganjem poslovnih procesa na više servisa koji međusobno sarađuju da bi se izvršila neka aktivnost. Međutim da bi se SOA razumela u potpunosti neophodno je bolje upoznavanje sa njenim komponentama, čemu služe, kako funkcionišu i na koji način su povezane.

# 4.Komponente

Osnovne komponente SOA arhitekture su:

- ESB(*Enterprise Service Bus*)
- SOA Registar
- *Workflow Engine*
- Servis Broker
- SOA Supervizor

Svaka od ovih komponenti ima određenu, nezavisnu od drugih komponenti, ulogu u sistemu kao i u saradnji sa ostalim delovima. ESB se stara o porukama koje se prenose između komponenti SOA sistema. SOA registar sadrži informacije o lokacijama SOA komponenti. *Workflow engine* je tehnoloigja koja povezuje ljude sa ljudima, ljude sa procesima i procese sa procesima, dok servis broker povezuje servise sa servisima što na kraju omogućava rad poslovnih procesa. Uloga SOA supervizora je da se postara da ceo sistem funkcioniše skladno i predvidivo.




Slika 1: komponente SOA arhitekture (Hurwitz, et al, 2007, slika 4-1)

# 4.1.ESB (Enterprise Service Bus)

ESB predstavlja centar za komunikaciju servisa u okviru SOA sistema. Dizajniran je da funkcioniše kao posrednik između SOA komponenti, infrastrukturnih servisa i poslovnih procesa. Povezuje se sa različitim tipovima srednjeg sloja(*middleware*) sistema, skladištima definicijama metapodataka, registrima i interfejsima aplikacija. Bitno je shvatiti da ESB nije hardverska komponenta već skup softverskih komponenata.

Postoje različite vrste ESB-a i razlikuju se po svojim mogućnostima. Svaki ESB ima apstraktni sloj koji je odgovoran za upravljanje porukama i na taj način omogućava spajanje i komunikaciju softverskih komponenti. Neki od njih mogu da funkcionišu sa raznim vrstama poruka od e-mail do SOAP-a, neki čak implementiraju i enkripciju. ESB može funkcionisati i kao servis broker, može posedovati i svoj registar pa čak može i preuzeti neke funkcije SOA supervizora. Ovo je dobar pokazatelj da se ESB razvijao nezavisno od SOA-e. Tako da je moguće imati ESB bez implementacije SOA sistema i obrnuto, naravno ovo je moguće samo u manjim SOA sistemima.

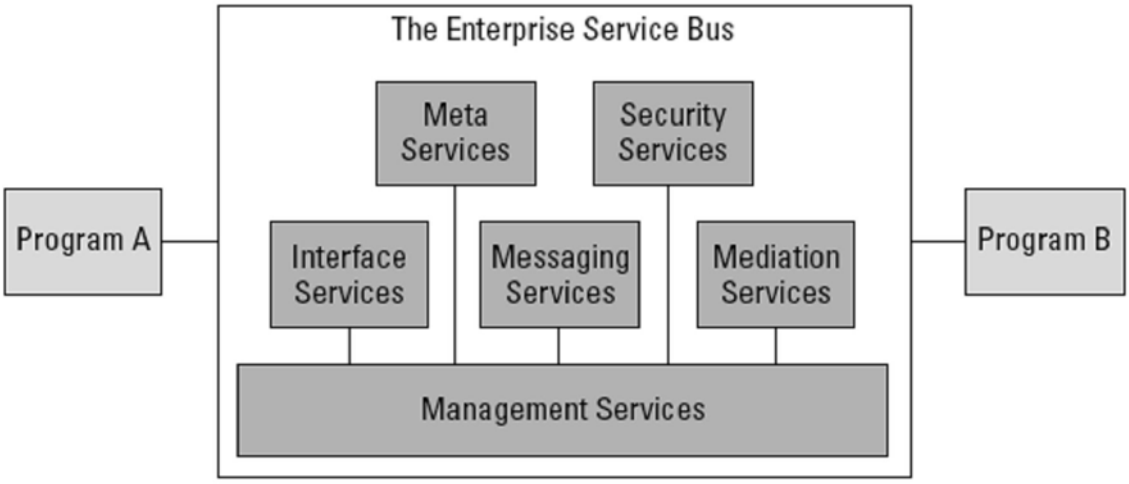
## 4.1.1.Servisi ESB-a

ESB izvršava infrastrukturne poslove koji bi inače morali da budu implementirani u samu aplikaciju. Servisi koje nudi ESB su:

- ***Messaging service*(Servis poruka)** – podržava širok spektar poruka, pruža inteligentno rutiranje na osnovu konteksta i garantuje isporuku. Takođe može kombinovati i deliti poruke.
- ***Managment service*(Upravljački servis)** – prati performanse i reaguje u slučaju velike *latencije*, implementira prioritete i globalna pravila aplikacijama i svim komponentama koje povezuje
- ***Interface service*(Interfejs servis)** – potvrđuje validnost poruka upoređivanjem sa XSD-om(XML Schema Definition) koja se nalazi u ESB registru. Podržava standarde veb servisa i pružaju adaptere za interfejse koji nemaju podršku za veb servise.
- ***Mediation service*(Posrednički servis)** – menja format poruke radi usklađivanja sa aplikacijom.

- ***Metadata service*(Servis meta podataka)** - povezan sa posredničkim servisom, takođe menja formate poruka koristeći se definicijama meta-podataka koje se nalaze u registru.

- ***Security service*(Sigurnosni servis)** – vrši enkripciju podataka i uključuje standardizovani sigurnosni metod autorizacije, autentifikacije i pregleda svih aktivnosti ESB-a.




Slika 2: Povezvanje dva programa pomoću ESB-a (Hurwitz, et al, 2007, slika 9-2)

#### 4.1.1.1.Servis poruka

Postoje šest osnovnih tipova poruka koje ESB koristi, oni se mogu kombinovati i mogu se primenjivati razna pravila da bi se dobili kompleksniji prenosi. Osnovni tipovi poruka su:

- ***Point-to-point* (poruke od tačke-do-tačke)** – Najjednostavniji vid poruka. Šalje je jedna osoba drugoj i nije neophodan odgovor.

- ***Point-to-point request/response*(poruke od tačke-do-tačke sa zahtevom za odgovor) –** Jedna osoba šalje poruku drugoj sa očekivanim odgovorom. ESB je svestan da je odgovor očekivan i poruke se šalju dok se komunikaicja ne završi, ovo predstavlja vrstu transakcije.

- ***Broadcast*(difuzni prenos poruka) –** Jedna osoba šalje poruku ka više ljudi, slično kao prvi primer samo što poruka ide na više lokacija.

- ***Broadcast request/response*(difuzni prenos poruka sa očekivanim odgovorom) –** Jedna osoba šalje poruku ka više ljudi sa očekivanim odgovorom, transakcija se ne završava dok svaki primalac ne odgovori.

- ***Publish subscribe*(objavljivanje poruka na oglasnoj tabli) –** Poruka se postavlja na „oglasnu tablu“ i vide je samo „pretplatnici“.

- ***Store and forward*(skladištenje i prosleđivanje poruka) –** ESB skladišti poruke ako korisnik nije trenutno u mogućnosti da ih primi i kasnije ih prosleđuje korisniku.

#### 4.1.1.2.Upravljački servis

Sve komponente ESB-a treba da funkcionišu zajedno da bi izvršavali svoje funkcije i neophodan je skup servisa koji kontroliše rad u okviru ESB-a. U „saobraćaju“ podataka uvek može doći do izvesnih grešaka, neki servis dolazi u koliziju sa drugim, transakcija prestaje bez ikakvog razloga ili nagli pad performansi. U ovakvim situacijama servisi moraju da se isprave i ponovo počnu sa radom.

Upravljački servis pre svega mora upravljati sobom, ovim pruža visoko pouzdani servis za sve poruke sa kojima radi. ESB je sposoban da balansira svoj rad upravljajući svim raspoloživim resursima i ima mogućnost da se oporavi od greške.

#### 4.1.1.3.Interfejs servis

Pored upravljanja sobom i porukama glavni zadatak ESB-a je da omogući komunikaciju dva ili više programa. Programi međusobno pričaju prosleđivanjem informacija i komandi jedni drugima. Ova komunikaicja nije jednostavna i zahteva izvesna pravila komunikacije, pre pojave XML-a korišćene su druge metode. ESB povezuje Web servise koristeći SOAP i WSDL ali može podržati i druge načine povezivanja, CORBA(*Common Object Request Broker Architecture*) – standard koji je povezan sa objektno-orjentisanim programiranjem, JMS(*Java Messaging Service*) – standard povezan sa Java programerskim okruženjem.

Broj različitih interfejsa koje ESB pruža je veoma bitan zato što to daje mogućnost povezivanja većeg broja programa bez dodatnog kodiranja. Interfejs servisi su skup adaptera koji omogućavaju povezivanje programa na ESB. Bitno je razlikovati SOA interfejs servise koji su takođe adapteri ali daleko kompleksniji od ESB adaptera.

### 4.1.1.4.Posrednički servis

Računarski sistemi se sastoje od računarskog hardvera, operativnog softvera i lokalnog softvera. Računarska okruženja se razlikuju i povezati ih međusobno nije jednostavno. ESB može predstavljati posrednika između ovako različitih okruženja kao što su IBM mejnfrejm i Windows server. Ova dva okruženja se potpuno razlikuju ali ESB treba da ih poveže tako da program pod jednim okruženjem može da komunicira sa programom iz drugog. Ovo se izvršava tako što ESB prevodi iz jednog protokola u drugi. Takođe je neophodno posredovati u samoj sadržini poruke. Ako npr. Jedan program definiše datum u formatu dd/mm/yy i poveže se sa programom koji definiše datum u formatu mm/dd/yy neophodno je promeniti format datuma i to izvršava posrednički servis.

### 4.1.1.5.Servis meta podataka

Meta podaci predstavljaju informaciju o strukturi i značenju podataka. Ako npr. imamo podatke o mušterijama takođe je neophodno da znamo kako je definisana mušterija u sistemu. Da bi programi komunicirali međusobno neophodno je da imaju istu definiciju podataka, sve se još više komplikuje kada je neohodno da naš softver komunicira sa softverom druge firme koja je saradnik. Jedini način za prevazilaženje ovog problema je da čuvamo podatke o definiicjama svih podataka za svaki program sa kojim naši programi komuniciraju, upravo ovo izvršava servis meta podataka. ESB može imati svoj registar svih programa koje povezuje, može da koristi eksterni SOA registar ili može kombinovati ova dva pristupa.

### 4.1.1.6.Sigurnosni servis

Sigurnost je jako bitna stavka u bilo kom računarskom sistemu tako da se i ESB bavi pitanjem sigurnosti. Sigurnost nije tako bitna tema kada povezujemo programe u okviru jedne organizacije ali čim se naši programi povežu sa programima drugih organizacija ovo postaje bitan prioritet.

U okviru sigurnosnog servisa ESB-a ne postoje sami sigurnosni mehanizmi nego ovaj servis pruža okvir za softver koji je posebno namenjen da se bavi pitanjem sigurnosti. Bitne stavke koje se tiču sigurnosti u okviru ESB-a su:

- **Autentifikacija –** provera autentičnosti korisnika ili poruke
- **Autorizacija –** provera prava pristupa korisnika
- **Privatnost –** zaštita informacija od neautorizovanih korisnika
- **Integritet –** provera autentičnosti podataka

Iako sigurnosne servise posmatramo kao zasebne servise oni su samo specijalizovani posrednički servisi.

### 4.1.2.Karakteristike ESB-a

Kada kompanija implmentira ESB obično počinje sa jednim ESB-om i on je sasvim dovoljan da zadovolji trenutne potrebe kompanije. Kako raste SOA sistem tako je neophodno uvesti nove ESB sisteme i povezati ih federaciju. Federacija je skup ESB sistema koji pružaju uslugu na nivou celog poslovanja. Svaki ESB poseduje svoja pravila i polise ali ona potpadaju pod viši skup SOA pravila.

U suštini ESB pruža mogućnost povezivanja softverskih komponenti i servisa u fleksibilnu vezu(*loose coupling*) . Pod ovim se podrazumeva da ESB pruža sloj koji omogućava komponentama da pozivaju i koirste servise drugih komponenti na jednostavan način. ESB pruža mogućnost korišćenja već postojećih i starih aplikacija njihovim povezivanjem. Fleksibilna veza omogućava povezivanje neograničenog broja korisničkih interfejsa, koji mogu pozivati aplikacije i servise koji se nalaze na različitim paltformama, u zajedničku mrežu.

## 4.2.SOA Registar

SOA Registar je centralana veza koja omogućava otkrivanje poslovnih servisa i pruža opise svih servisa koji imaju referencu u tom registru. Svaki servis koji je zabeležen u registru je morao da prođe kroz verifikaciju od strane poslovnog i IT menadžmenta. U registru je takođe zabeležena istorija svakog servisa, ko je kreator servisa, ko ima mogućnost promene servisa, kako se koristi i ko ima pravo pristupa.

SOA Registar nije pasivni direktorijum već se menja u realnom vremenu(real-time registry), menja se u skladu sa promenama poslovnih pravila. Poseduje tri ključne funkcije:

- **Objavljivanje i pružanje mogućnosti otkrivanja servisa**
- **Sakupljanje i upravljanje meta podacima poslovnih servisa**
- **Upravljanje korišćenjem poslovnih servisa**

### 4.2.1Objavljivanje

SOA Registar pruža mogućnost pregleda raspoloživih servisa od strane korisnika. Servisi mogu biti od koristi i osobama koje rade u drugim organiazacijama kao što su mušterije ili poslovni partneri. Pored smeštanja definicija i opisa servisa, registar služi i za objavljivanje(*publish*) poslovnih servisa, na taj način im mogu pristupiti svi zainteresovani za usluge koje pruža neki servis.

### 4.2.2Upravljanje meta-podacima

SOA Registar upravlja meta-podacima koji se odnose na neki poslovni servis. Meta-podaci predstavljaju opise poslovnih servisa, poslovnih pravila vezanih za servis i pravila za upravljanje serivisima. Meta-podaci uključuju tehničke opise kako se jedan servis povezuje sa drugim.

### 4.2.3Upravljanje korišćenjem

Neophodno je obezbediti da servisi koji su objavljeni budu u skladu sa poslovnim pravilima organizacije. Svaki servis koji je objavljen mora proći kroz sva pravila registra ili razvojnih delova SOA-e.

### 4.2.4Komponente SOA Registra

Sve što je smešteno u registru je u formi meta-podataka i tiče se pravila upravljanja servisima u okviru SOA okruženja. SOA Registar se sastoji od:

- **Komponente za opisivanje interfejsa Web servisa –** Koristi se UDDI(*Universal Description, Discovery, and Integration*) standard koji podržavaju svi SOA Registri. Kreiran je pomoću XML tehnologije
- **Komponente za opisivanje ostalih tipova iterfejsa –** Pored interfejsa kreiranih XML-om moguće je koristiti već postojeće interfejse. Dovoljno je imati podatke o pravilima korišćenja postojećih interfejsa.
- **Definicije poslovnih procesa –** Za izvršenje nekog poslovnog procesa neophodno je povezati više komponenti. Da bi se one povezale na pravi način i da bi se izvršio poslovni proces neophodno je da imamo mapu i potpunu definiciju poslovnih procesa.

- **Pravila poslovnih procesa –** Svaki proces mora se izvršavati u okviru određenih pravila. Kako ne bi morali da za svaki proces definišemo posebno pravila ona su centralitovana i smeštena u SOA Registar. Ovo takođe olakšava pronalaženje pravila koja se tiču određenog poslovnog procesa.

- **Opisa nivoa performansi i dostupnosti servisa –** Skup pravila koja opisuju nivo preformansi i dostupnosti koje računarska mreža mora obezbediti za rad određenog servisa.

- **Pravila upravljanja –** Sadržaj registra mora se povinovati određenim pravilima upravljanja. Svaka promena se mora kontrolisati da ne bi došlo do grešaka koje se mogu odraziti na funkcionalnost celog sistema.

## 4.3.Workflow Engine

Predstavlja softversku komponentu koja povezuje ceo poslovni proces u jednu celinu(*from end-to-end*), prenoseći rad sa jedne individue ili procesa ka drugoj dok se ceo proces ne izvrši. Tehnologije za razvoj *Workflow-a* omogućavaju modeliranje poslovnih procesa da bi se dobila *workflow* šema(*workflow pattern*). Ova šema je zapravo skup instrukcija koje izvršava *workflow engine*.

*Workflow* razvoj je nastao daleko pre SOA-e i postoje mnogi proizvodi ovog tipa, često sa akcentom na specifičnu oblast kojoj su namenjeni. Neki od njih su:

- ***jBPM*** – Ovo je *workflow engine* kompanije *jBoss* koji izvršava procese opisane u BPEL-u(*Business Process Execution Language*) ili u sopstvenom jeziku za definisanje rocesa jPDL-u.

- ***OSWorkflow*** – Ovo je proizvod *open source* projekta *OpenSymphony*. Nastoji da pruži fleksibilnost u razvoju i ne poseduje sopstveni grafički interfejs.

- ***Apache ODE (Orchestration Director Engine) –*** Proizvod softverske *open source* fondacije *Apache*. Izvršava poslovne procese napisane u skladu sa standardom *WS-BPL(Web Services Business Process Execution Language*).

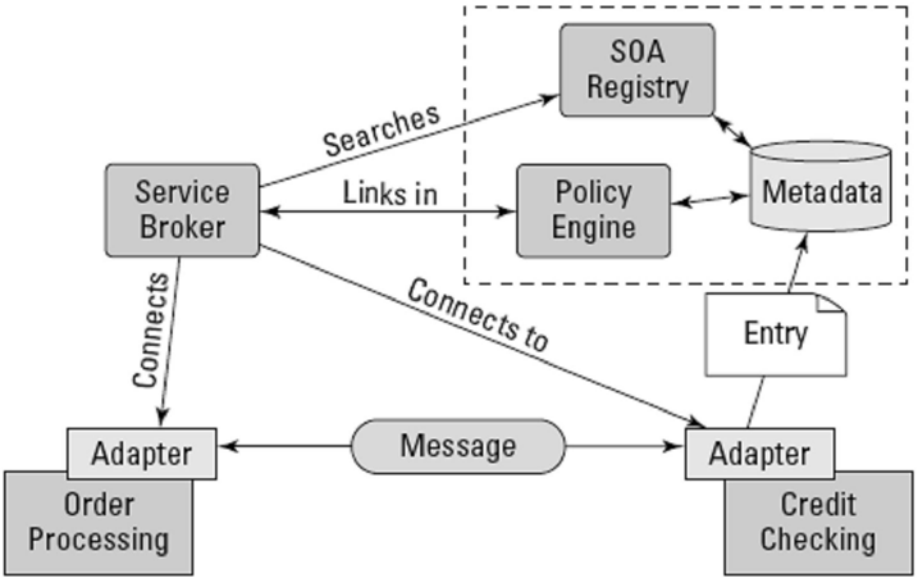
- ***W4 BPM Embedded Edition –*** Proizvod Evropske softverske kompanije W4 koji podržava veliki broj operativnih sistema(Linux, Solaris, Windows, HP-UX, Aix, iSeries - AS400 i zSeries) i baza podataka(MySql, PostgreSQL, Oracle, SQLServer, DB2 i Informix)

## 4.4.Servis Broker

Servis broker je komponenta koja omogućava funkcionalnost veza između svih ostalih komponenti. Spaja više komponenti u niz koje čine poslovni proces. Koristi informacije o komponentama koje nalazi u SOA registru i spaja ih pripremajući ih za rad w*orkflow engine-a*. Servis broker započinje sve procese i kada završi sa spajanjem komponenti u poslovnom procesu prelazi na sledeći zadatak spajanja.

Servis broker je neophodna komponenta zato što je SOA sistem sačinjen od slabo povezanih(*loosely coupled*) komponenti koje se moraju povezati u smislenu celinu da bi se izvršio neki poslovni proces. On orkestrira rad komponenti praveći vezu između njih pomoću informacija iz SOA registra.

Na slici 3 je ilustrativno prikazano kako broker funkcioniše u saradnji sa registrom. Slika prikazuje povezivanje servisa porudžbine sa servisom provere kreditne sposobnosti. Servis porudžbine, preko brokera koji koristi informacije iz registra o servisu provere kreditne sposobnosti, vrši zahtev za usluge servisa provere kreditne sposobnosti. Servis broker koristi informacije iz registra kako da pronađe ova dva servisa ali i po kojim pravilima rade i kako da objedini dva skupa pravila u jednu funkcionalnu celinu. Na slici vidimo *policy engine* koji implementira pravila rada i koji je zapravo deo registra. Informacija o servisu provere kreditne sposobnosti je objavljena u registru, što se takođe vidi na slici.




Slika 3: Rad servis brokera u saradnji sa SOA registrom (Hurwitz, et al, 2007, slika 8-1)

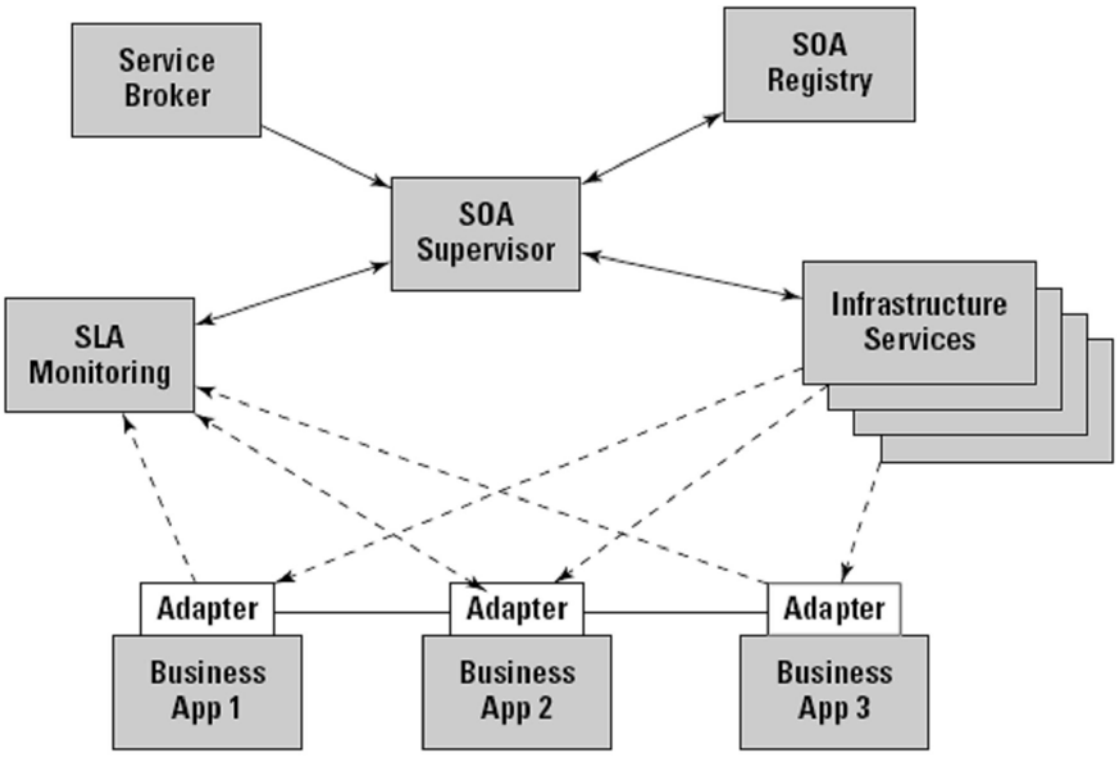
Servis brokeri su proizvodi sredjneg sloja(*middleware*). Neki su evoluirali iz ORB-a(*object request broker)*, neki iz EAI-a(*Enterprise application integration*) a neki su evoluirali dalje u ESB sisteme. Većina ESB sistema može funkcionisati kao servis broker.

## 4.5.SOA Supervizor

Sistem slabo povezanih komponenti(*loose coupling*) ne može biti toliko efikasan i pružiti nivo performansi kao sistem čvrsto povezanih komponenti(*tight coupling*). Zato je uloga SOA supervizora toliko bitna, on omogućava prihvatljiv nivo usluga i performansi servisa. Takođe je bitno napomenuti da SOA supervizor mora biti aktivan dok god je neki servis aktivan i vrši se neki poslovni proces.

SOA tehnologija je još uvek mlada, većina firmi još uvek je ne implementira u potpunosti, tako da rad SOA supervizora nije zahtevan toliko koliko će biti u budućnosti. Kako kompanija sve više implementira SOA sistem tako će se povećavati i zahtevi koje SOA supervizor mora da ispuni da bi se održao prihvatljiv nivo performansi.

Slika 4 prikazuje rad superizora u saradnji sa ostalim komponentama, ovakav sistem se ne može još uvek naći u praktičnoj primeni ali je to ono čemu se teži.




Slika 4: Rad SOA supervizora u SOA okruženju (Hurwitz, et al, 2007, slika 10-2)

Ceo proces započinje servis broker slanjem poruke SOA supervizoru da je spojio određene servise i tako omogućio početak nekog poslovnog procesa. SOA supervizor se konsultuje sa registrom oko detalja vezanih za poslovni proces da bi mogao da postavi softver za nadgledanje svih neophodnih komponenti. Ovo nadgledanje izvršava SLA(*Service Level Agreement*) monitoring komponenta koja preko lokalnih adaptera aplikacija dobija izveštaje o nivou performansi. Izveštaji se prosleđuju SOA supervizoru koji ih prikazuje na konzoli da bi se mogli pratiti u realnom vremenu.

Kada se pojavi problem SOA supervizor obaveštava infrastrukturne servise, koji bi trebali da reše problem. Ako ovi servisi nisu umogućnosti da reše problem obaveštava se osoblje o problemu koji se pojavio.

Stvari u realnosti izgledaju dosta drugačije nego što su prikazane na dijagramu. Većina kompanija nije za sada u mogućnosti da obezbedi rad SOA supervizora kako je prikazano na slici. Ovo ne znači da one nemaju funkcionalan SOA sistem, samo da im za sada to nije neophodno zato što nisu implementirali SOA-u u potpunosti. Ali vremenom će se SOA sistem širiti i da bi se postigla ovakva funkcionalnost SOA supervizora pre svega je potrebno obezbediti:

- **Dobro definisane nivoe prihvatljivih performansi – SLA(*Service Level Agreement*)**
- **Potpuno funkcionalan infrastrukturni softver**
- **Mape poslovnih procesa**
- **Popis hrdverske opreme i softvera koji se koristi**

# 5.Bezbednost SOA sistema

Bitna stavka svakog informacionog sistema je bezbednost, pogotovo kada se radi o otvorenom informacionom sistemu koji ima veliki broj korisnika i koji pruža veliki broj usluga. Ovo pravilo takođe važi i za SOA sisteme. Nijedna tehnologija ne nudi savršeno rešenje i da bi se postigao zadovoljavajući nivo bezbednosti, SOA sitema, neophodno je razumeti osnovne principe sigurnosti informacija i razumeti dizajn i arhitekturu SOA sistema. Ključno za dobar sistem bezbednosti je da se sigurnosna strategija i plan razvoja usvoje u ranim fazama implementacije SOA sistema. Dobri sigurnosni sistemi omogućavaju poslovnim aplikacijama da ispune sva očekivanja organizacije uz implementaciju osnovnih bezbednosnih ciljeva, kao što su:

- **Autentifikacija**
- **Autorizacija**
- **Federativni identitet**
- **Privatnost**
- **Integritet**
- **Nemogućnost poricanja(*non-repudiation)* –** Garancija da pošiljalac poruke ne može moreći da je poslao poruku i da primalac nemože poreći da je primio poruku
- **Preglednost**
- **Dostupnost**

# 5.1.Autentifikacija

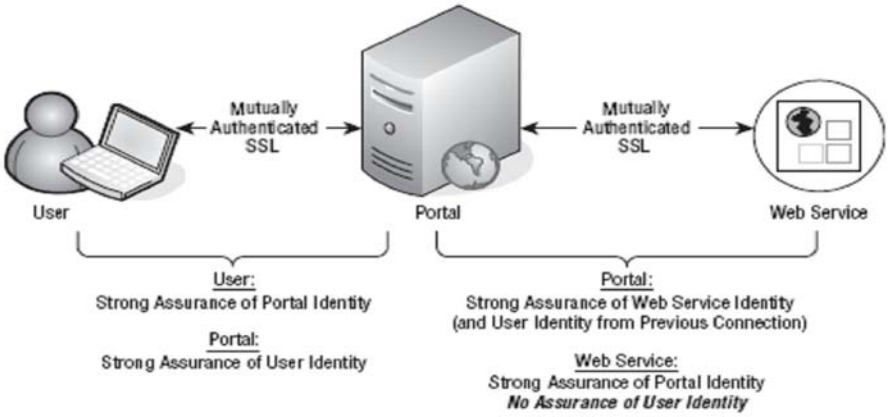
Predstavlja proveru validnosti identiteta subjekta. Subjekat može biti korisnik, web servis, računar ili aplikacija. Autentifikacija je prvi korak u kontroli prava pristupa. Da bi se omogućila kontrola prava pristupa neophodno je da sistem identifikuje subjekat i da poseduje izvestan nivo uverenja u autentičnost identiteta subjekta. Međusobna autentifikacija predstavlja dvosmernu autentifikaciju i omogućava dokazivanje identiteta obe strane uključene u komunikaciju.

Postoji više mehanizama autentifikacije, uobičajene metode su korišćenjem:

- **Korisničkog imena i šifre**
- **Digitalnog sertifikata**
- **Biometrijskih uređaja**

Uobičajeni kriptografski protokoli koji se koriste su SSL(*Secure Sockets Layer*)/TLS(*Transport Layer Security*). Ovi protokoli imaju ugrađene mehanizme autentifikacije kao što su provera korisničkog imena i šifre i provera digitalnim sertifikatom. Iako pružaju relativno visok nivo bezbednosti mogu se koristiti za, jednosmernu i dvosmernu, proveru autentičnosti identiteta samo dva subjekta istovremeno.

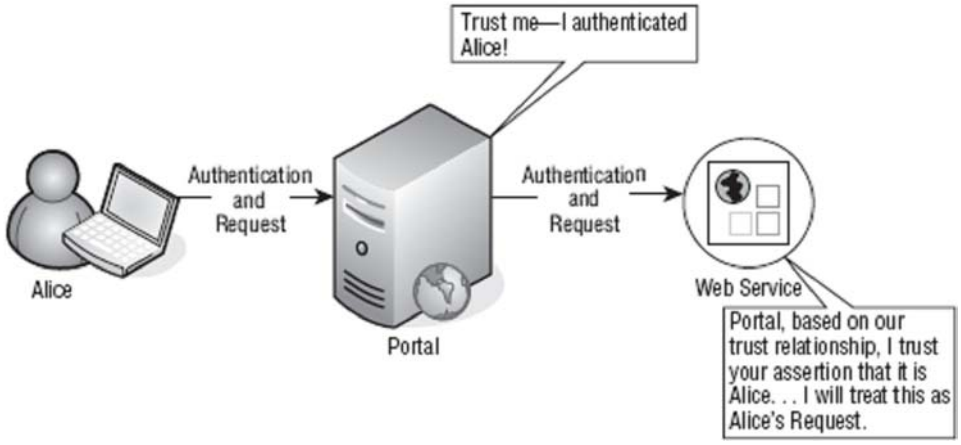
Na slici 5 imamo korisnika koji digitalnim sertifikatom vrši autentifikaciju na neki portal, koristeći SSL, i pokreće određenu aplikaciju. Portal vrši autentifikaciju na web servis kako bi mogao da upotrebi njegove resurse. Svaka komponenta u komunikaciji ima poverenje u identitet druge sa kojom direktno komunicira, tako korisnik i portal kao i portal i servis imaju međusobnu identifikaciju ali korisnik i servis nemaju.




Slika 5: Autentifikacija digitalnim sertifikatom pomoću SSL standarda (Rosen, M., et al, 2008, slika 11-1)

Ovo je primer sa samo 3 tačke, obično ih je mnogo više u nekom procesu. Često je jako bitno da poslednja komponenta u procesu ima jako uverenje u identitet početne komponente tj. korisnika. Ovo je jasan pokazatelj da u SOA sitemu može da se koristi SSL protokol ali da on nije dovoljan i da se on mora dopunjavati drugim sigurnosnim mehanizmima.

Sledeća slika ilustruje primer korisnika koji se autentifikuje na portal koji dalje šalje zahtev servisu. U ovoj situaciji se koristi metoda jemstva(*sender-vouch*), što znači da portal garantuje, jemči, za identitet korisnika koji zahteva neku uslugu preko portala. Ovo je jedna od metoda prenošenja identiteta(*identity propagation*) koja se koristi u WS-Security (*Web Services Security*) protokolu. Identitet se prenosi na aplikacije i servise da bi se postigao *Single sign-on (SSO)* metod. Da bi servis verovao u identitet korisnika, za koji se garantuje, mora posedovati visok stepen poverenje u portal koji pruža tu garanciju. Ovo znači da servis, pored autentifikacije portala, mora imati poverenje i u tačnost garancije portala. Ove dve vrste poverenja se moraju drugačije i odvojeno tretirati.



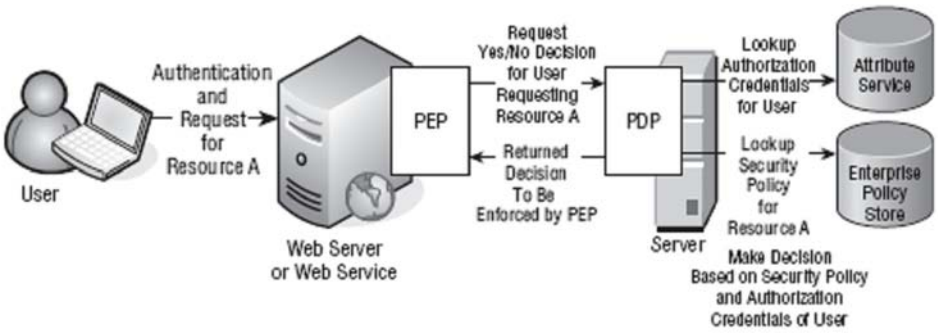

Slika 6: Prijava korisnika na web servis uz garanciju portala u identitet korisnika (Rosen, M., et al, 2008, slika 11-2)

Ovo predstavlja jednostavan primer prenošenja identiteta koji demonstrira važnost autentifikacije između svih delova nekog procesa. U komplikovanijim procesim autentifikacija postaje dosta složenija. Kao jedan od bitnijih delova sigurnosti, izbor metode autentifikacije diktira fleksibilnost celog SOA sistema.

## 5.2.Autorizacija

Autorizacija je određivanje prava i dozvola koje korisnik poseduje. Nakon autentifikacije, provere identiteta, sistem mora odrediti prava koji korisnik poseduje i koje su njegove mogućnosti korišćenja sistemskih resursa.

Fleksibilnost autorizacije se obično postiže odvajanjem zadataka na dve ključne tačke, *Policy Decision Points (PDP) i Policy Enforcement Points (PEP).* PDP je tačka u kojoj se donose odluke o pravima pristupa a PEP je zadužen za primenu tj. nametanje tih prava korisniku.




Slika 7: Autorizacija pomoću PEP-a i PDP-a (Rosen, M., et al, 2008, slika 11-3)

Autorizacija se može podeliti na dva tipa kotrole prava pristupa:

- **Diskretna kontrola pristupa – *Discretionary Access Control (DAC)***
- **Obavezna kontroila pristupa - *Mandatory Access Control (MAC)***

DAC zabranjuje pristup na osnovu dozvola, uloga, atributa i grupa kojima subjekat pripada. Dozvole pristupa resurisma su centralizovana i kada subjekat zatraži pristup, PDP određuje njegova prava. Ovaj tip kontrole pristupa se obično koristi u komercijalnim SOA sistemima.

MAC je češće korišćen državnim SOA sistemima i kontroliše pravo pristupa sigurnosnim dozvolama i sigurnosnim oznakama na resursima. Ovo znači da sami podaci moraju imati sigurnosne oznake i da PEP tačke moraju da odrede prava prustupa subjekta, traženim resurisima, upoređivanjem oznaka sa datim pravima subjekta.

RBAC(*Role-Based Access Control*) je uobičajena metoda koja se koristi u primenjivanju diskretne(DAC) kontrole pristupa. Sigurnosne uloge su definisane i dodeljene subjektima, a pravo pristupa resursima je određeno tim ulogama. PDP donosi odluke o pravima pristupa na osnovu uloga subjekata.

ABAC(*Attribute-Based Access Control*) je slicna RBAC-u ali koristi sigurnosne atribute a ne sigurnosne uloge. U ovakvom sistemu su autorizacijska prava, kao što su uloga, grupa, sigurnosno odobrenje i pripadnost, predstavljeni kao atributi. Ova metoda se koristi u diskretnoj(DAC) i obaveznoj(MAC) kontroli prava pristupa.

PADBAC(*Predetermined Authorization Decision-Based Access Control)* predstavlja strategiju kontrole pristupa subjekta, koristeći diskretnu(DAC) ili obaveznu(MAC) metodu za određivanje prava pristupa, dodeljivanjem prava u obliku karata, koje se kasnije koriste za pristupanje resursima.

## 5.3.Federativni identitet

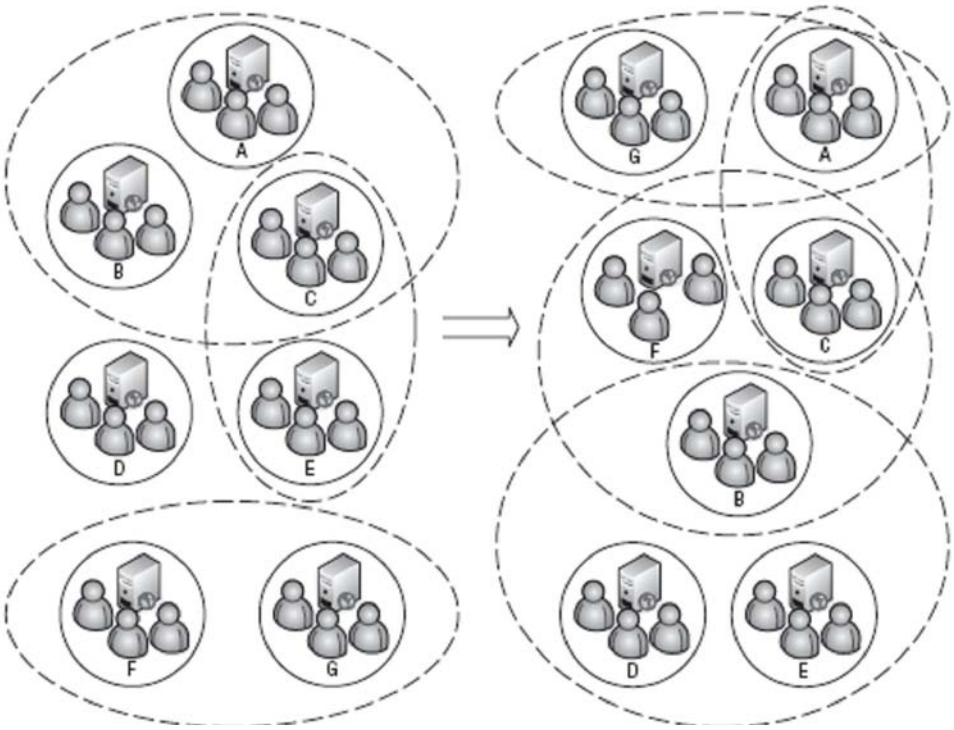
Cilj federativnog identiteta je da omogući korisnicima da sa jednom identifikacijom mogu pristupiti na više sigurnosnih domena i tako se služe resursima partnerskih kompanija. Skup partnerskih kompanija se naziva federacija, pristup korisnika drugim domenima koristeći SSO metod je omogućen putem ove federacije. U federaciji partneri dele način na koji se određuje validnost korisnika i na koji način se vrši autorizacija. Ključna tačka kod ovakvih rešenja je da se kompanija oslanja na spolja definisani identitet i nema potrebe da stvara lokalni identitet za svakog korisnika koji prilazi van okvira kompanije. Ovakav pristup umanjuje troškove upravljanja i kompleksnost sistema bezbednosti, bez njenog ugrožavanja.

Uspeh sistema federativnog identiteta zavisi od sposobnosti deljenja obimnog i zajednički razumljivog skupa zahteva između organizacija u federaciji. Ideja federativnog identiteta evoluira i širi svoje polje delovanja na druge informacije pored identiteta. U praksi postoje dva protokola za pristup u federativnom okruženju:

- ***Browser-based SSO –*** Koriste se brauzer klijenti i web aplikacije uz korišćenje HTTP(*Hypertext Transfer Protocol*) standarda.
- ***Service-based SSO –*** Koristi se između servisa.

Bitna osobina u federaciji su dinamični odnosi u federaciji. Iako je uobičajeno da federacioni identitet bude dugoročan, bitno je da moderno doba dinamičnog poslovanja zahteva fleksibilniji model. Arhitekture svih organizacija u federaciji moraju međusobno uskladiti svoje poslovne procese da bi sarađivale. Pored toga bitna je zaštita svojine i informacija koje su poverljive.

Na slici 8 je prikazan model u kome organizacije pripadaju više različitih federacija u isto vreme. Da bi ispunile potrebe autentifikacije i autorizacije partnera u ovako dinamičnom okruženju organizacije moraju usvojiti model polisa pristupa koji je centralizovan i koji prevazilazi okvire kompanije.




Slika 8: Model dinamične federacije (Rosen, M., et al, 2008, slika 11-5)

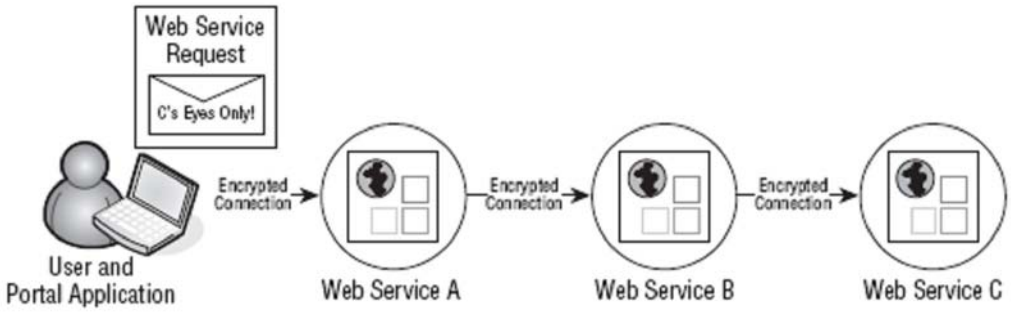
2005 godine organizacija OASIS(*Organization for the Advancement of Structured Information Standards*) je objavila SAML 2.0(*Security Assertion Markup Language*) standard, objedinivši *Shibboleth* i ID-FF(*LibertyAlliance Identity Federation Framework*) standarde, koji se odnosi na federativni identitet. Organizacija *The Liberty Alliance* je takođe objavila ID-WSF(*Identity Web Services Framework*) standard koji se oslanja na SAML 2.0. ovo su trenutno dva aktuelna standarda koji se koriste u modelu federativnog identiteta.

# 5.4.Privatnost

U svakoj komunikaciju koja se odvija između autorizovanih učesnika neophodna je njihova privatnost, ovo u konkretnom smislu znači da informacije koje se prenose mogu da vide i čitaju samo ti učesnici. Privatnost se postiže time što se osetljive informacije sakrivaju enkripcijom. U procesu enkripcije se obična tekstualna poruka šifruje kriptografskim algoritmom da bi se dobila šifrovana poruka. Korišćenjem ključa, tj. zajedničke tajne primalac poruke je može dešifrovati. Postoje mnogo različitih kriptografskih algoritama, mogu biti simetrični, sa tajnim ključem i asimetrični, sa javnim ključem.

U poslovnom SOA okruženju mogu postojati poverljivi podaci koji zahtevaju neki nivo enkripcije. Mnogi protokoli kao i SSL/TLS pružaju mogućnost masovnog kodiranja(*bulk*

*encryption*) između dve tačke. Ako ovaj nivo zaštite ispunjava zahteve poslovanja i ne predstavlja pretnju po sigurnost ovo može biti sasvim prihvatljivo rešenje. Međutim postoje SOA sistemi gde masovno kodiranje nije dovoljno. Ovaj sistem kodiranja omogućava zaštićenu komunikaciju između dve tačke ali svaka tačka koja se nalazi na putu između pošiljaoca i primaoca ima uvid u poslate informacije, slika 9.




Slika 9: Masovno kodiranje(*bulk encryption*) (Rosen, M., et al, 2008, slika 11-6)

Masovno kodiranje omogućava poverljivost između svih tačaka, ali ako postoji deo poruke koji je namenjen samo veb servisu C, masovno kodiranje ne pruža takvu mogućnost. U ovom slučaju se može koristiti XML standard za enkripciju poruka koji je propisala W3C.

U stvaranju rešenja, koje bi zadovoljile potrebe privatnosti, neophodno je uzeti u obzir:

- **Menadžment ključeva –** upravljenje distribuiranjem ključeva
- **Odabir šifara**
- **Kriptografske protokole**
- **Jačina enkripcije**

Veliki izazov je utvrđivanje tajnih ključeva za razmenu enkriptovanih informacija. Zbog sporosti i zahtevnosti kriptografije javnim ključevima ona se koristi za pregovaranje o ključevima – utvrđivanje tajnih ključeva koji se koriste za enkripciju, a ne za enkripciju samih podataka

Standardi, kao što je SSL, pružaju mogućnost uspostavljanja ključeva za sesije koji se koriste u dugoročnim HTTPS(*Hypertext Transfer Protocol Secure*) sesijama. Poruke se razmenjuju ,između dve strane, preko istog ključa, koji obezbeđuje privatnost poslatih poruka, dok god traje sesija.

Standardi kao što je *WS-Security SOAP messaging* nemaju podršku za koncept sesija i problematični su kada je neophodno uspostaviti dugoročnu razmenu poruka. Ovo je posledica togo što je neophodno da se za svaku poruku mora izvršiti pregovaranje, enkripcijom javnog ključa, da bi se ustanovio tajni ključ. Ovo je rezultiralo time da se često implementira kombinacija *WS-Security SOAP messaging* i SSL standarda između dve tače komunikacije. OASIS je, 2007 godine, propisao standard pod nazivom *WS-SecureConversation* koji se koristi za uspostavljanje dugoročnih sesija korišćenjem *WS-trust* modela.

# 5.5.Integritet

U komunikaciji, a posebno u servisnim transakcijama, neophodno je postarati se da podaci nisu falsifikovani ili menjani na bilo koji način, ovo važi i za prenos i za smeštanje podataka. Validnost integriteta podataka predstavlja tehnologiju dokazivanja da poruka nije menjana od strane neautorizovanog subjekta. Zbog mogućnosti zloupotreba poruka preko TCP/IP mreža neophodno je koristiti digitalne potpise, autentifikacione kodove ili *hash* algoritme da bi se potvrdila validnost integriteta poruke.

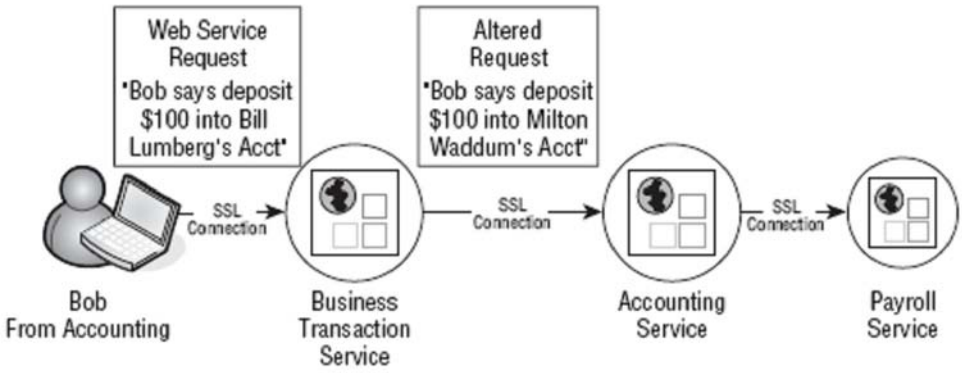
U SOA okruženju neophodni su mehanizmi, koji bi obezbedili validnost integriteta poruka, između svakog korisnika servisa i onog koji pruža taj servis. Primalac svake poruke mora imati visok nivo uverenja da ona nije izmenjena od strane nekog ko nije učesnik u komunikaciji. Pored toga neophodno je da primalac ima uverenje da poruka sa validnim integritetom nije replicirana od treće strane koja može zloupotrebiti poruku. Iz ovog razloga, dobri sigurnosni protokoli za poruke koriste mehanizmime integriteta za kombinovanje poruka sa vremenskim oznakama i identifikatorima poruka. Ovim se garantuje integritet datuma i vremena poslate poruke, identiteta poruke i same poruke. Ako integritet vremenske oznake ili identitet poruke nije validan, ako je vremenska oznaka istekla, ako je poruka menjana i ako je poruka sa istom identifikacijom poslata ona se automatski odbija od strane primaoca. Mehanizmi koje *WS-Security SOAP messaging* specifikacija pruža obezbeđuju integritet poruka pored mogućnosti sprečavanja napada ponovo poslatim porukama.

SSL/TLS protokol obezbeđuje integritet između dve tačke, kombinujući ga sa mehanizmima autentifikacije i poverljivosti. Kao i kod autentifikacije, ovo može pružiti dovoljan nivo zaštite između dve tačke ali u SOA sistemu najčešće nije dovoljno.

Slika 10 pokazuje kako SSL štiti integritet i poverljivost podataka između svake tačke ponaosob, ali ne sprečava manipulaciju podacima, od strane posredničkih servisa pre nego što ih proslede.

Na slici 10 računovodstvo, preko veb portala, zahteva depozit od 100 dolara na račun zaposlenog. Između aplikacije klijenta i servisa za poslovne transakcije postoji zajednička

autentifikovana SSL konekcija koja obezbeđuje privatnost i integritet podataka i identifikaciju identiteta učesnika komunikacije. Ovaj sigurnosni mehanizam se nalazi između svih tačaka na slici. Međutim ovo nije dovoljno da se jedan od veb servis spreči da izmeni originalnu poruku i pošalje je dalje kao validnu. Ovde vidimo da jedan od veb servisa menja poruku i uplata stiže na račun pogrešne osobe.




Slika 10: Zloupotreba transakcije SSL protokolom (Rosen, M., et al, 2008, slika 11-7)

*XML signature* standard, koji je propisan od strane W3C-a, se može koristiti kao deo rešenja prethodnog problema. Originalni zahtev je digitalno potpisan i tako se prenosi dalje, ovo pruža integritet poruci. Ako se podatak izmeni u bilo kom trenutku on ne može proći proveru integriteta. Istovrmeno se koristi XML potpis da bi se sprečilo ponovno slanje iste poruke.

Da bi se napravilo dobro rešenje za zaštitu integriteta podataka neophodno je unapred pogledati i znati sve scenarije korišćenja servisa u okviru SOA okruženja i sve zahteve po pitanju bezbednosti. Ovo je jedini siguran način da se spreče napadi.

## 5.6.Nemogućnost poricanja(*non-repudiation*)

Nemogućnost poricanja je posledica digitalnog potpisivanja poruke, i predstavlja legalni dokaz da je subjekat potpisao poruku. Digitalni potpis kriptografski veze identitet potpisivača sa sadržajem potpisanih podataka. Korišćenje kriptografije javnim ključem omogućava neoporecivost pošiljaoca da je on potpisao poruku.

Korišćenjem *XML Signature* standarda za potpisivanje poruka ili samo pojedinih delova poruka, dokazuje se integritet ali se postiže i nemogućnost poricanja potpisa. Ovaj standard se može koristiti sa servisima koji se baziraju na REST i *WS-Security SOAP Messaging* standardima.

Iako digitalni potpis pruža visok nivo sigurnosti važno je da pri implementaciji rešenja za slanje poruka u okviru SOA sistema, budemo svesni potencijalnih propusta i opasnosti primene i korišćenja digitalnih potpisa. Velika je opasnost po sigurnost je ako se određeni XML element potpiše bez naznačavanja uslova njegovog korišćenja ili bez njegovog vezivanja za specifični zahtev. U SOA porukama uobičajeno je da se naglase ograničenja i uslovi korišćenja potpisanih podataka. Kada je neophodna identifikacija identiteta on se mora kriptografski vezati sa konkretnim zahtevom pomoću digitalnog potpisa, na taj način je onemogućeno korišćenje potpisanih podataka u druge svrhe. Ovo znači da potpis, pored podataka mora uključiti i kontekst njihovog korišćenja kao i vremenska ograničenja. Postoje razne druge opasnosti pri korišćenju digitalnog potpisa u okviru SOA okruženja. Bezbednosni sistemi koji se baziraju na SOAP standardima pružaju dobro okruženje za izbegavanje takvih opasnosti.

## 5.7.Sigurnosni standardi i specifikacije u okviru veb servisa

Postoji veliki broj standarda bezbednosti koji se odnose na veb servise, dole navedeni su najvažniji:

- ***WS-Security SOAP Messaging***
- ***WS-Trust***
- ***WS-Federation***
- ***WS-SecureConversation***
- ***WS-Policy***
- ***WS-SecurityPolicy***
- ***SAML***
- ***XACML***
- ***XML*Potpis**
- ***XML* Enkripcija**

Većina navedenih standarda su široko prihvaćeni i primenjuju se u praksi. Poznavanje ovih standarda je neophodno za svako ozbiljnije projektovanje SOA sistema, tj. dela vezanog za sigurnost.

### 5.7.1. *WS-Security SOAP Messaging*

Široko prihvaćen i korišćen standard, propisan od strane OASIS-a. Pruža mogućnost implementacije provere integriteta, nemogućnosti poricanja, poverljivosti i prosleđivanja novčića(*token passing*). Obezbeđuje visok nivo fleksibilnosti spajanjem SOAP poruka sa više bezbednosnih standarda i tehnologija. Spajanjem se praktično omogućava:

- Proširivanje SOAP zaglavlja(*header*) sigurnosnim informacijama i tako se omogućava bezbedno slanje poruka
- Uspešniji rad standarda nižeg nivoa kao što su *XML Signature* i *XML Encryption*
- Rad sa više formata novčića(*tokens*) koji se koriste za utvrđivanje identiteta i provere autorizacije
- Podršku za više *trust* modela

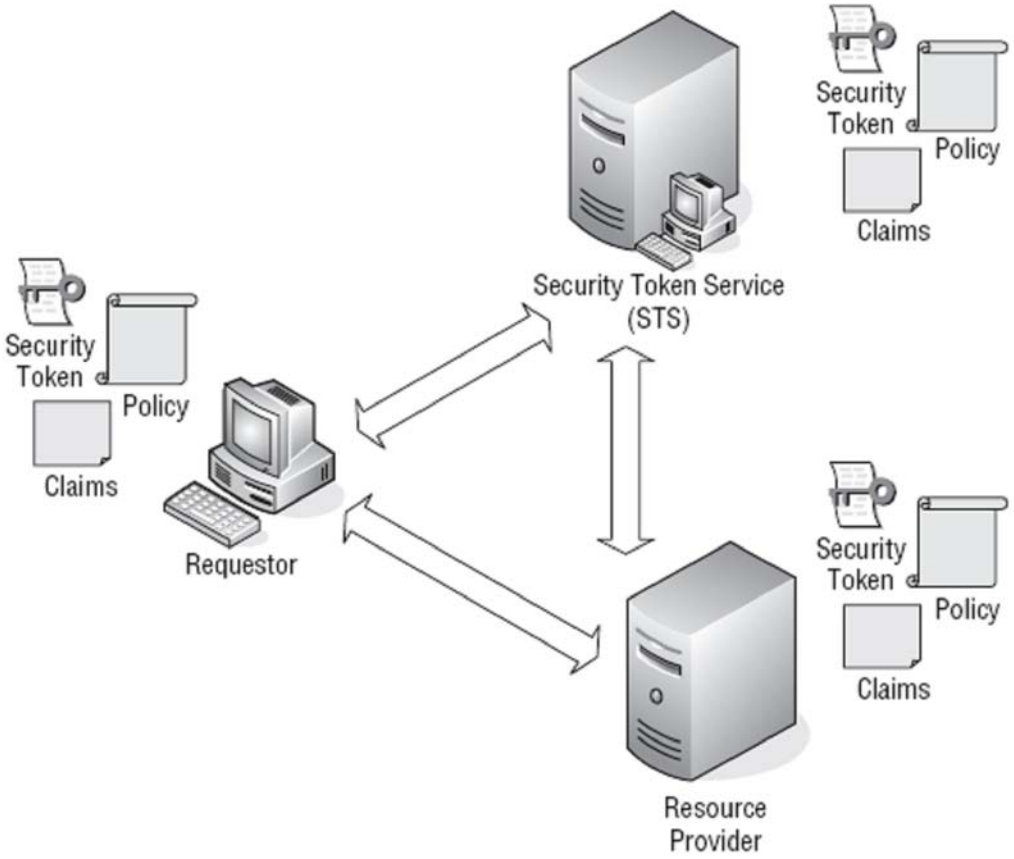
Sam po sebi ovaj standard ne pruža kompletno rešenje za pitanja bezbednosti veb servisa koji se baziraju na SOAP porukama. On predstavlja sredstvo za sigurno slanje poruka, ovo postiže spajanjem i usaglašavanjem širokog spektra tehnologija i modela poruka. Može se koristiti sa komplementarnim standardima višg nivoa za izgradnju kompletnog SOA bezbednosnog rešenja.

Mnogi standardi su napravljeni korišćenjem *WS-Security SOAP Messaging* standarda kao osnov. Svaki od ovih standarda ima drugačiju strategiju i scenarije korišćenja u širem sistemu, time je omogućena velika fleksibilnost u saradnji sa drugim standardima.

## 5.7.2.*WS-Trust*

Deo porodice *WS-Security* specifikacija sa ulogom definisanja usluge bezbednosnog novčića(*token*) – STS(*Security Token Service*) i protokol za traženje i dodelu bezbednosnih novčića koji se koriste od strane *WS-Security SOAP Messaging* standarda.

Na slici 11 je prikazan *WS-Trust* STS model gde STS pruža bezbednosni identitet. Pružaoci usluga se oslanjaju na novčiće koje je obezbedio STS da bi se uspostavilo poverenje nadolazećih zahteva. U SOA okruženjima, koje koriste ovaj model, svi pružaoci usluga i konzumenti tih usluga imaju svoje polise koje usaglašavaju pri određivanju novčića za komunikaciju.




Slika 11: *WS-Trust* model (Rosen, M., et al, 2008, slika 11-8)

Slika prikazuje konzumenta ,u želji da uspostavi konekciju sa pružaocem usluge, koji šalje upit pružaocu usluge o sigurnosnoj polisi da bi se odredili zahteve komunikacije. Obično su ovi zahtevi definisani u *WS-Security* polisi. Ove polise sadrže liste novčića koji su neophodni za komunikaciju sa servisom i opciono sadrže imena STS-a koji mogu ponuditi odgovarajuće novčiće. Ako potražioc nema odgovarajući novčić pn ga može zahtevati od STS-a, koji zatim autentifikuje potražioca i daje odgovarajući novčić koji pruža identitet potražiocu. Konačno, na kraju se potražioc povezuje na pružaoca usluga, sa odgovarajućim novčićem koji obezbeđuje izvesna prava u okviru nekog servisa. Pružaoc usluge može proveriti validnost novčića kod STS-a.

### 5.7.3.*WS-Federation*

Ova specifikacija je predstavljena 2007 godine od strane OASIS-a, kao produžetak *WS-Trust* standarda i STS modela i protokola ona se koristi u pružanju federativnog identiteta. Ovaj identitet služi za saradnju sa pojedincima i firmama koje prevazilaze okvire jedne organizacije. Pojavljivanjem i definisanjem raznih federativnih servisa, uključujući i servise autentifikacije, autorizacije, atributa i pseudonima, pružena je mogućnost korišćenja i pristupanja servisima koji se nalaze u drugim domenima-*cross domain Web Service communication*.

*WS-Federation* pruža mogućnost novog STS-a koji se naziva autorizacijski servis. Ovaj servis omogućava PDP i PEP standarde, tj. servise, za učensnike u federaciji. Takođe definiše model, koji se bazira na STS konceptu, za pristup servisima atributa. Servisi pseudonima se koriste za opis atributa neophodnih za pristup nekom resursu bez potrebe za odavanjem identiteta subjekta koji potražuje servis. Takođe je proširen i STS protokol u smislu da se izbegne prenos poverljivih podataka sa prenosom novčića.

*WS-Federation* se može koristiti i u *browser-based* i *Web Service-based* federativnom identitetu.

### 5.7.4.*WS-SecureConversation*

Ovo je standard koji je na vrhu lestvice, po važnosti, *WS-Security* standarda. Njegova uloga je da omogući bezbednu komunikaciju između servisa. Specifikuje mehanizme za uspostavljanje i deljenje bezbednosnog sadržaja iz kojih se izvode sigurnosni ključevi. Ovaj standard se može koristiti za razmenu više poruka u jednoj konverzaciji tj. uspostavljenoj vezi na duže staze- *long-lived relationship*. Ovo je veoma korisno u situacijama kada je neophodno uspostaviti ovakvu vezu(između portala i veb servisa npr.). Umesto korišćenja pojedinačnih ključeva za svaku poruku uspostavljaju se ključevi sesije i komunikacija je nekoliko puta brža i zahteva daleko manje performanse celokupnog sistema.

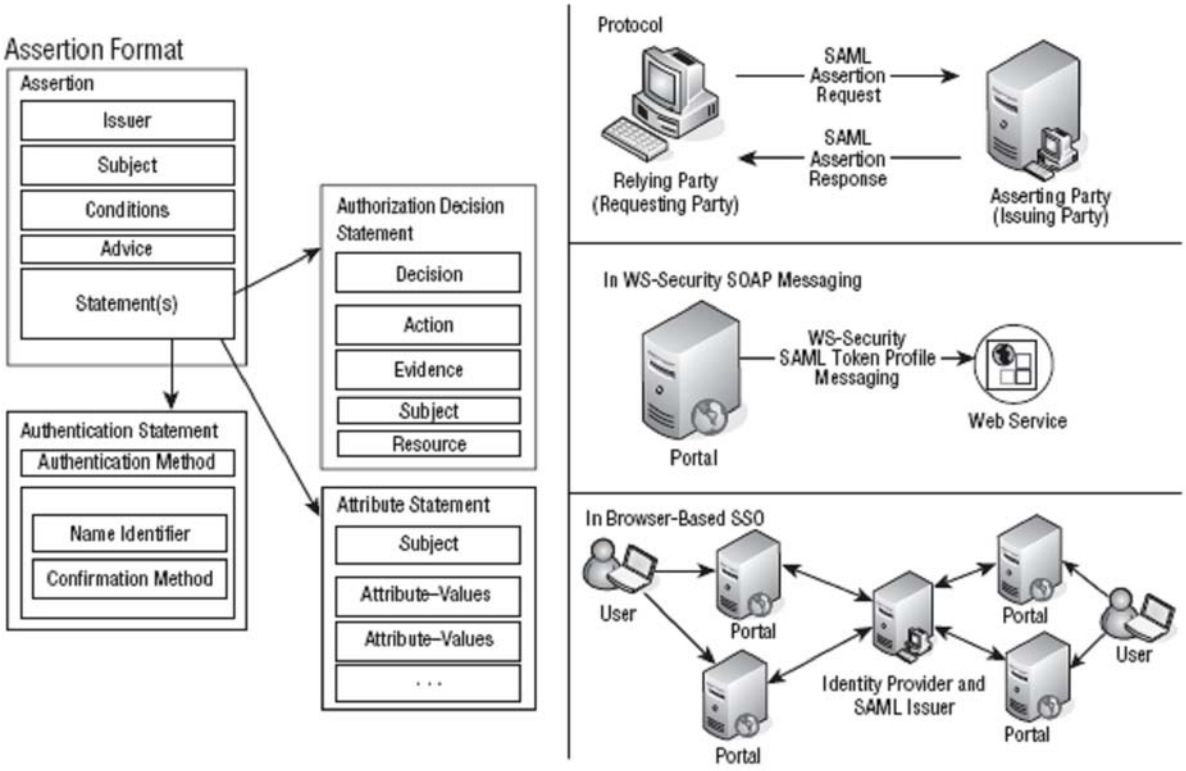
### 5.7.5. *WS-SecurityPolicy* i *WS-Policy Framework*

*WS-Policy* tj. *Web Services Policy Framework* je XML jezik koji se koristi za predstavljanje mogućnosti, ograničenja i zahteva veb servisa. Ove mogućnosti i zahtevi su izraženi kao polise. Zbog nemogućnosti WSDL-a da izrazi i definiše mogućnosti i zahteve veb servisa, kao što su kvalitet usluge-QoS(*Quality of service*) i bezbednost poruka, dopunjen je *WS-Policy* jezikom. *WS-SecurityPolicy* je "podmodel" *WS-policy* standarda koji pruža skup iskaza polisa. Polise opisuju zahteve, koje treba postići po pitanju sigurnosti poruka koje se koriste za komunikaciju veb servisa. Koristi se u saradnji sa *WS-Security SOAP Messaging*, *WS-Trust*, i *WS-SecureConversation* standardima.

## 5.7.6.*SAML*

SAML(*The Security Assertion Markup Language*) je okruženje bazirano na XML-u i koristi se za uspostavljanje autentifikacije i naziva subjekta kao i atributa informacije. Koristi se u razne svrhe i obično u kombinaciji sa drugim standardima uključujući i *WS-Security SOAP Messaging, WS-Trust, WS-Federation, XACML* i *ID-WSF(Liberty Identity Web Services Framework)*. Ovaj standard se razvija oko koncepta tvrdnje(*assertion*), koji je predstavlja izjavu o subjektu. Može biti tvrdnja o autentifikaciji subjekta(identitet subjekta), spisku autorizacijskih prava subjekta ili izraz autorizacijske odluke koja daje pristup nekom resursu. Poverenje u SAML tvrdnju je bazirano na poverenju u entitet koji zahteva tu istu tvrdnju. Od usvajanja 2002 godine ovaj standard je značajno uznapredovao i sa verzijom 2.0 pruža podršku za federativni identitet.

Slika 12 je primer korišćenja SAML-a kao protokola za zahtev/odgovor (*request/response)* u potražnji tvrdnji za odluke o pravima pristupa. Pokazuje korišćenje SAML novčića u *WS-Security* i konceptualnu upotrebu SAML tvrdnji u *browser-based* SSO okruženju.




Slika 12: Upotreba SAML-a u SOA okruženju (Rosen, M., et al, 2008, slika 11-9)

Vidimo tri tipa izjave u tvrdnji:

- **Izjava autentifikacije** - Pruža informacije o tome kako, kada i od strane koga je izvršena autentifikacija subjekta
- **Izjava atributa** - Prikazuje spisak atributa koji se odnose na subjekat
- **Izjava autorizacijske odluke** - Koristi se za prikaz prava subjekta da pristupi određenom resursu

SAML takođe definiše zahtev/odgovor (*request/response)* protokol za tražnju tvrdnji, obično se koristi za autorizaciju.

SAML novčići se mogu koristiti za slanje poruka u SOA okruženjima, obično u saradnji sa *WS-Security* ili drugim protokolima.

SAML 2.0 pruža podršku za SSO. Pojedini tipovi SSO-a podržani od strane SAML-a us:

- **Identitetni SSO –** Identitet korisnika je registrovan u oba domena
- **Atributni SSO** – Kontrolu prava pristupa resursima izvršavaju autorizacijski atributi

Anonimni SSO se postiže prosleđivanjem SAML izjava o atributima, takođe postoji mogućnost korišćenja pseudonima radi veće privatnosti.

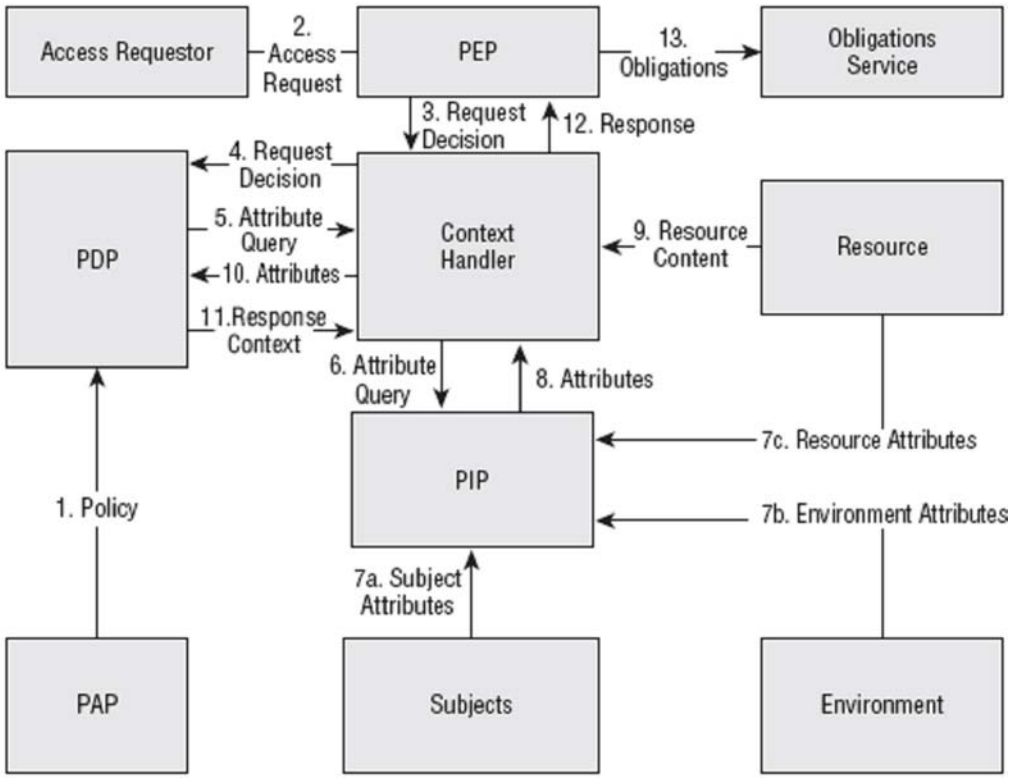
SAML *Web browser* SSO profil određuje kako se tvrdnje o autentifikaciji pregovaraju između pružaoca identiteta i pružaoca usluga.

## 5.7.7.XACML

XACML(*eXtensible Access Control Markup Language*) je ekspresivni i fleksibilni jezik zasnovan na XML-u. Koristi se za dostavljanje polisa o kontroli prava pristupa resurisma i uključuje zahtev/odgovor (*request/response)* protokol za slanje upita o pravima pristupa. On opisuje zahteve kontrole prava pristupa i može se proširivati definisajem novih funkcija, tipova podataka i logikom.

Na slici 13 prikazan je protok XACML podataka. PAP kreira XACML polise za resurse i čini ih dotupnim PDP-u. Kada subjekat zatraži pristup resursu PEP šalje zahtev o pristupu upravljaču konteksta(*context handler*), koji prenosi zahtev PDP-u. PDP traži pristup atributima subjekta, resursa i okruženja da bi mogao da donese odluku o pravima pristupa. Zahtev za atributima se šalje PIP-u i on ih vraća. Kada PDP dobije informaciju vraća autorizacijsku odluku,

kroz upravljač konteksta, PEP-u. PEP donosi odluke na osnovu obaveza koje se dobijaju od strane servisa obaveza(*obligation service*).




Slika 13: Dijagram protoka podataka upotrebom XACML-a (Rosen, M., et al, 2008, slika 11-10)

Iako XACML pruža zahtev/odgovor (*request/response)* protokol za mnoge od interakcija na slici 13, velika količina informacija može biti prosleđena i SAML novčićima i mnoge razmene mogu koristiti SAML protokol. XACML se prevashodno fokusira na mehanizme koji omogućavaju pristizanje autorizacijskih odluka.

## 5.7.8.XML Potpis

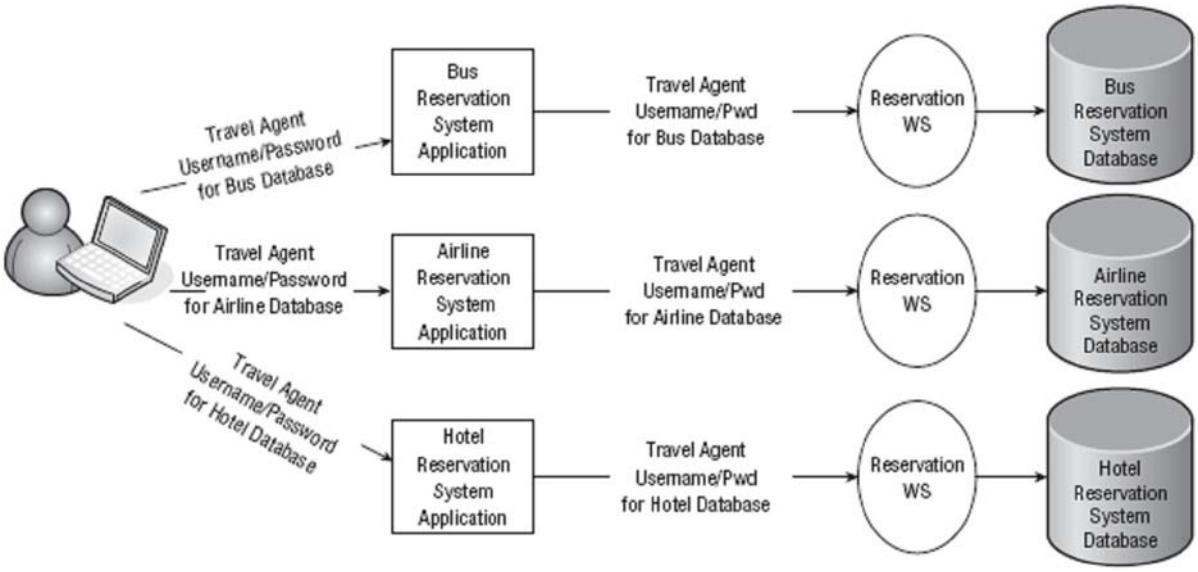
Standard propisan od strane W3C-a koji služi da omogući porukama integritet i nemogućnost poricanja XML dokumenta ili elementa dokumenta. Moguće je naići na oznake XML-SIG ili XML-DSIG koje takođe predstavljaju XML potpis. Oslanja se na tehnmologiju javnog ključa, gde je *hash* deo ili skup XML elemenata potpisan privatnim ključem pošiljaoca i može se kriptografski dešifrovati samo odgovarajućim javnim ključem pošiljaoca.

Poruka može imati elemente koji su digitalno potpisani od strane više različitih strana, i potpis može kriptografski spojiti elemente. Moguće je spojiti identitet, autorizaciju i vreme iz

*WS-security* naslova i SOAP zahtev, ovim sprečavamo napade ponovnim korišćenjem istih poruka.

## 5.8.Greške i rešenja u praksi implementacije sistema bezbednosti-Studija slučaja: turističke agencije

Ovo je studija slučaja fiktivne turističke agencije koja posluje dugi niz godina i integrisala je veliki broj kompanija nasledivši njihove sisteme rada. Zaposleni , u agenciji, imaju 20 naloga na nasleđenim sistemima, ovi sistemi služe za rezervaciju hotela, regulaciju troškova leta i drugih vrsta transporta. Par godina ranije urađena je revizija celog sistema ali su problemi ostali i pored implementacije novih tehnologija. Slika 14 prikazuje trenutno stanje poslovanja agencije, prikazano je 3 od 20 sistema koji se koriste.




Slika 14: Trenutni sistem rada agencije, prikaz 3 od 20 aktuelnih sistema (Rosen, M., et al, 2008, slika 11-23)

Kao što vidimo na slici, zaposleni moraju raditi na 20 aplikacija sa još tolikim brojem kombniacija korisničkog imena/šifre, ovo stvara velike probleme i pokazatelj je loše implementiranog sistema bezbednosti. Falinka ovog sistema ne predstavlja nizak nivo bezbednosti, već nepromišljenost pri projektovanju i nerazumevanje da dobar sistem bezbednosti mora biti što jednostavniji za korisnika, a da pri tome ne narušava sigurnost celokupnog sistema.

Ovo takođe predstavlja problem za administratore baza podataka, koji vode računa o 20 različitih baza, zbog konstantnog bavljenja greškama unosa korisničkog imena/šifre i menjanjem

prava pristupa podacima. Kada se nekom menjaju prava pristupa to se mora uraditi na svih 20 baza podataka.

Nakon kraće studije trenutnog rešenja i starije dokumentacije vidimo da revizija sistema ima velikih mana koje se tiču bezbednosti ali da je donela i neka poboljšanja. Napisano je 20 novih aplikacija, po uzoru na stare monlitne, koje se baziraju na klijent/server arhitekturi, razdvojen je sloj prezentacije od poslovne logike, dizajneri veb servisa su kreirali dobru šemu rezervacija koja se može koristiti za sve aplikacije, nezavisno od tipa rezervacije. Unos u baze podataka se izvršava preko jedinstvenog veb servis interfejsa.

Pokušaj kompletne revizije celog sistema je polu uspešan i nedovršen. Dizajneri sistema su želeli da kreiraju interfejs zasnovan na tehnologiji portala, koji bi omogućio sve rezervacije preko jedinstvene aplikacije. Pitanje bezbednosti nije razmatrano do samog kraja projekta i, u nedostatku vremena, implementacija bezbednosnog sistema nije vezana za sam sistem, već je pravljena kao nezavisna. Takođe su napravljene loše odluke oko prenošenja(*propagation*) korisničkog imena/šifre između delova sistema. Za svaki sistem aplikacija prosleđuje prava pristupa korisnika veb servisu, koji ih prenosi na bazu podataka. Kao rezultat ovoga svaki korisnik mora se posebno ulogovati za svaki sistem, ovo dovodi do odustajanje od ideje jedinstvenog interfejsa.

Gorenavedeno je primer dobre namere ali sa izostavljenjem pitanja, implementacije sistema bezbednosti, do samog kraja, dovodi do loših rezultata. Problemi oko komplikovane upotrebe i održavanja ostaju. Pitanje bezbednosti mora pratiti ceo projekat i mora se paralelno razvijati sa svakim delom sistema, tj. ono mora biti deo sistema od samog početka projektovanja. Naravno, ovo komplikuje projektovanje ali je krajnji rezultat daleko bolji, daleko je lakše korišćenje i održavanje sistema.

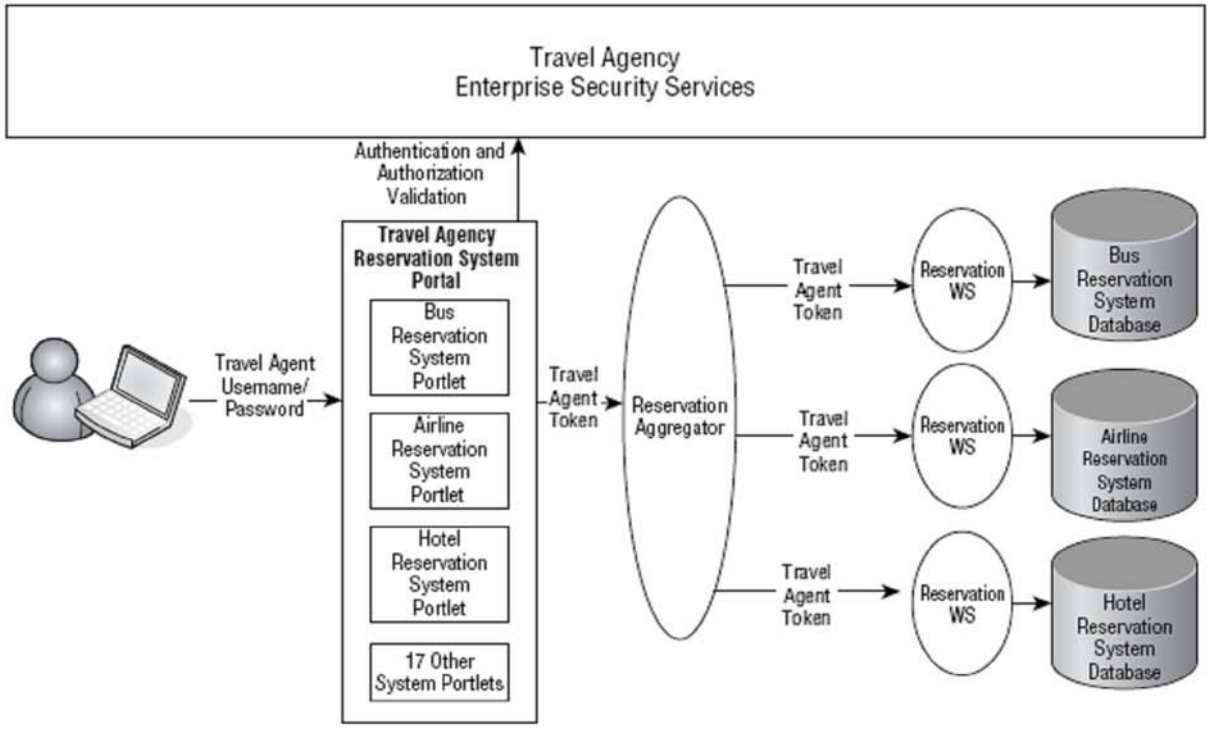
Koren problema, dizajana sistema, ove turističke agencije je jaka sprega(*tight coupling*) delova sistema. Pitanje bezbednosti je posebno vezano za svaku od 20 baza podataka. Ovo ne vezuje samo podatke, iz baze opdataka, sa sigurnosnom polisom prava pristupa, već i aplikaciju sa bazom podataka. Uzrok ovoga je posebna autentifikacija za svaki od ovih sistema. Problem je u odsustvu SSO sistema koji bi omogućio daleko efikasnije poslovanje i smanjio zahteve sistema administracije.

Uviđamo da je neophodno uraditi dve ključne stvari u sistemu:

- **Uspostaviti servise bezbednosti**, na nivou celokupnog poslovanja, koji bi objedinili polise prava za sve zaposlene. Ovim se relocira pitanje bezbednosti sa baza podataka na posebne servise i pružena je mogućnost SSO modela.

- **Neophodna je definisati metodu prenosa identiteta** da bi se izbeglo višestruko proveravanje između delova sistema, što bi i povećalo efikasnost i performanse celokupnog sistema. Fleksibilnost bi obezbedilo korišćenje nekog sistema sa novčićem(*token*) za prenos identiteta.

Smeštanjem sigurnosnih servisa u sistem rada i pružanjem SSO mogućnosti dobijamo sledeći dijagram, slika 15. Stvoren je portal, koji radi sa više *portlet*-a odjednom, koji korisnicima pruža mogućnost pristupa svim sistemima preko jedinstvenog interfejsa. Takođe je neophodan i servis koji radi kao agregator rezevacija(servis agregacije), on obavlja sve transakcije rezervacija, za jednog korisnika, u jednom zahtevu. Novčić, koji predstavlja identitet korisnika, se prenosi kroz sve delove sistema.




Slika 15: Željeni način rada servisa agencije (Rosen, M., et al, 2008, slika 11-24)

U praksi to izgleda ovako:

- **Korisnik se uloguje na portal** – Portal za svaki sistem poseduje po *portlet* koji je vezan za taj sistem.
- **Portal šalje zahtev sa novčićem korisnika**
- **Zahtevi se agregatorom rezervacija grupišu i šalju u odgovarajuću bazu**

### 5.8.1. Uspostavljanje sigurnosnih servisa

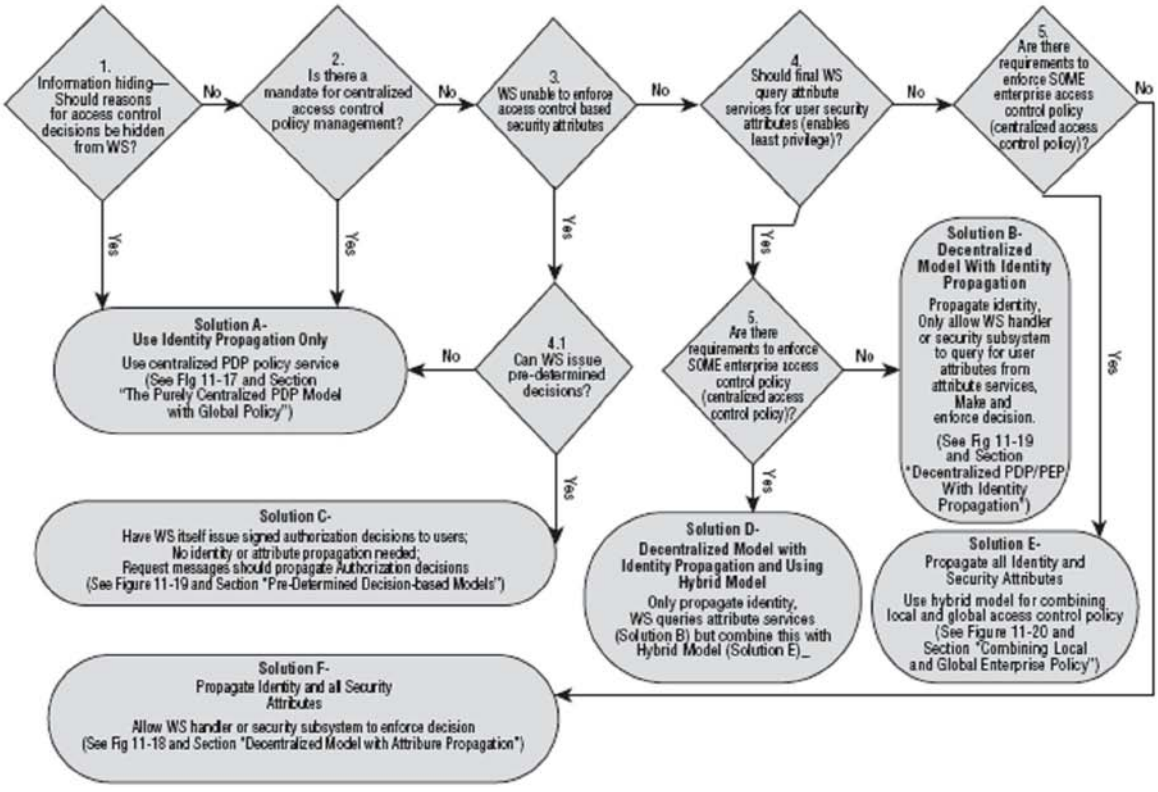
Postoji još bitnih odluka koje treba doneti. Najpre je neophodno definisati servise bezbednosti. Na raspolaganju nam stoji širok spektar opcija za kontrolu prava pristupa, odabir se vrši na osnovu primenjenih bezbednosnih servisa. Fleksibilnost se postiže korišćenjem bezbednosnih servisa, koji pokrivaju sve alternative primena polisa, kontrole prava pristupa. Ovo daje za mogućnost budućim aplikacijama, sa drugačijim zahtevima, korišćenje više različitih opcija. Iz ovih razloga neophodni su nam:

- **Servis autentifikacije –** Identifikuje korisnika i/ili aplikacije
- **Servis atributa –** Izlaže autorizacijska prava korisnika. Trenutne uloge zaposlenih su: odobravalac_rezervacija, rezervator_putovanja, odobravalac_kupovine. Ovaj servis omogućava ove uloge i ostavlja mogućnost proširenja i daje im fleksibilnost.
- **Servis upravljanja polisama –** Koristi se od strane administratora polisama za kreiranje, menjanje i upravljanje polisama.
- **Servis za pronalaženje polisa –** Izlaže polise resursa. Polise se mogu izraziti pomoću XACML jezika za bilo koji resurs, bez obzira na njegov izvor. Na ovaj način se baze podataka oslobađaju od pojedinačnog rukovanja polisama.
- **Servis za autorizacijske odluke –** Odlučuje o pravu pristupa, subjekta nekom resursu, na osnovu polisa. Da bi doneo autorizacijsku odluku mora znati:
  - Identitet subjekta koji potražuje resurs - postiže se sigurnosnim novčićem.
  - Autorizacijske mogućnosti subjekta – postiže se pomoću servisa atributa

- Polise koje se tiču potraženog resursa i polise poslovanja – postiže se pomoću servisa za pronalaženje polisa

- **STS –** Izdaje novčiće koji su vezani za identitet i atribute bezbednsti korisnika kao i za autorizacijske odluke. Ovaj seris identifikuje korisnike i tako omogućava komunikaciju sa drugim servisima. Pri kreiranju novčića za identitet STS komunicira sa servisom autentifikacije radi utvrđivanja identiteta korisnika. Pri kreiranju novčića za identitet i atribute STS komunicira sa servisom autentifikacije i servisom atributa. Pri kreiranju novčića za donošenje autorizacijskih odluka STS mora komunicirati sa svim servisima.

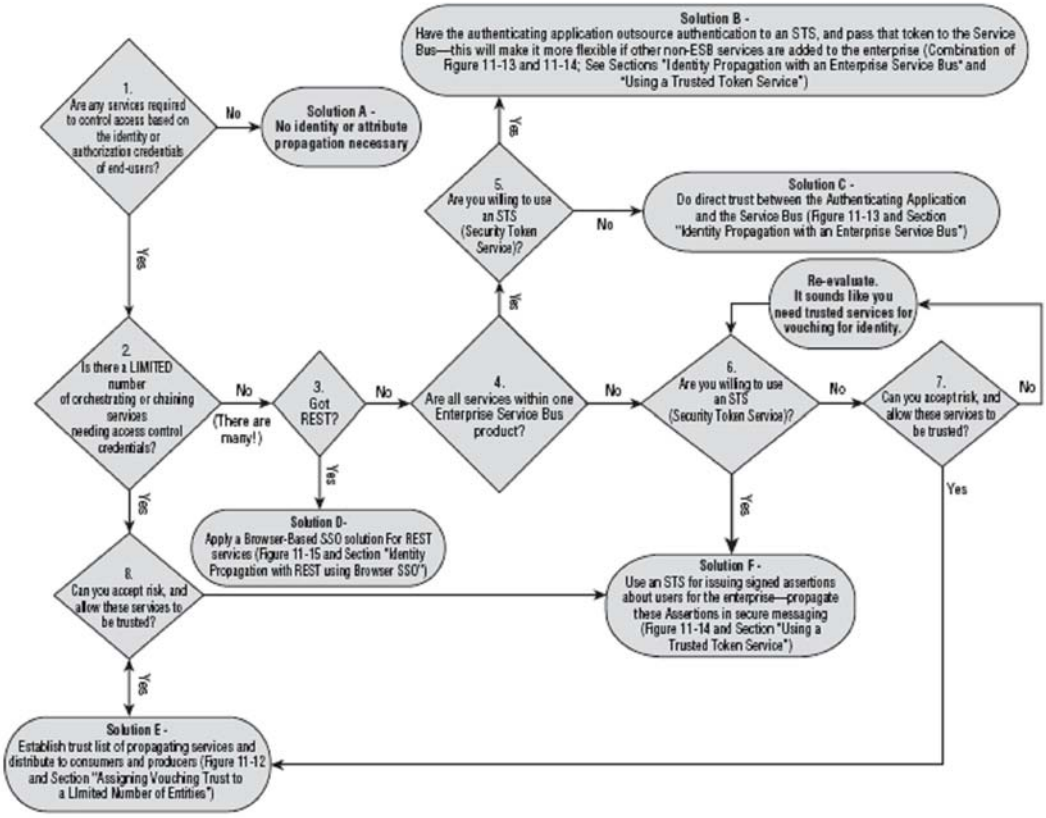
Ovi servisi odgovaraju šemi koja je prikazana na slici 16. Ovo je dijagram odluka o načinu kontrole prava pristupa sa 6 potencijalnih netoda. Odvajanjem pitanja bezbednosti od ostatka sistema i davanje te uloge ovim servisima dobijamo fleksibilnost arhitekture. Ova fleksibilnost se ogleda u tome da su promene u sistemu bezbednosti lako izvodljive i ne moraju se odraziti na celokupni sistema.




Slika 16: Dijagram odluka o načinu (Rosen, M., et al, 2008, slika 11-22)

## 5.8.2.Definisanje prenosa identiteta i kontrole prava pruistupa

Pošto smo odlučili da koristimo prenos pomoću novčića, u sistemu ove turističke agencije, ovo nam ostavlja otvorene mogućnosti po pitanju konkretnog tipa autentifikacije. Sada je bitno pogledati dijagrame odluka o načinu kontrole prava pristupa, slika 16 i dijagram odluka o načinu prenosa identiteta, slika 17. Ovo je važno zato što tpi kontrole prava pristupa određuje vrstu koja vrsta novčića se koristi, identiteta, atributa ili oba.

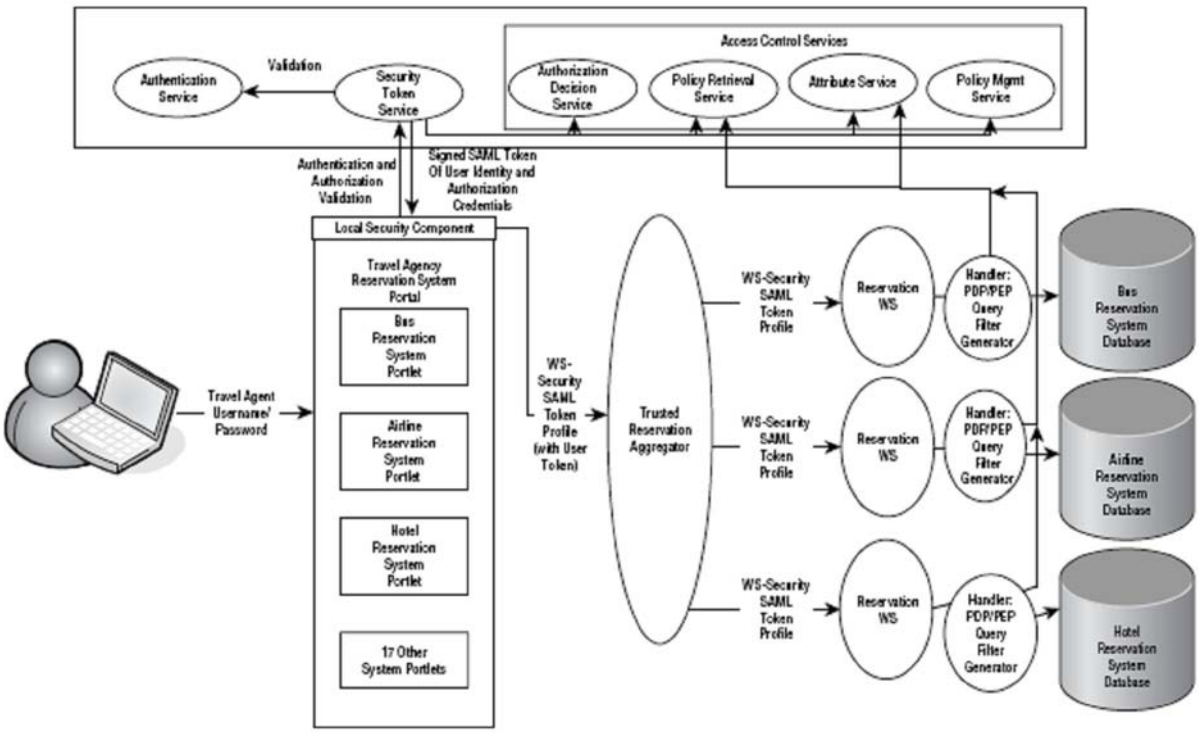



Slika 17: Dijagram odluka o načinu prenosa identiteta (Rosen, M., et al, 2008, slika 11-16)

Ranije, smo takođe odlučili da uspostavimo STS i pošto je jedini posrednički servis, između portala i veb servisa rezervacija , servis agregacije usvajamo STS pristup u prenosu identiteta, primer F na slici 17. Bitno je da, iz sigurnosnih razloga , se kontrola prava pristupa izvršava centralno ali nametne daljinski. Ovim se postiže kombinacija lokalne i centralne polise, primer D na slici 16. Na ovaj način svaka odluka o kontroli prava pristupa donosi poređenjem centralne i lokalne polise sa atributima korisnika.

Slika 18 prikazuje arhitekturu bezbednosti koja se bazira na prethodnim odlukama. Korisnička aplikacija, portal, može autentifikovati korisnika i tako inicirati komunikaciju sa

STS-om, koji preuzima novčić koji predstavlja identitet korisnika. Za bezbedno slanje poruka se ,zbog rasprostranjenosti i dobre prakse u korišćenju, odabiraju SAML novčići i *WS-Security* SAML profil novčića. Bazirajući se na uspešnoj autentifikaciji, autorizacijskim pravima i polisi kontrole prava pristupa – pružaju je bezbednosni servisi, svaki veb servis može izvršavati filtrirane upite na svakoj bazi podataka.




Slika 18: Arhitektura bezbednosti korišćenjem sigurnosnih servisa (Rosen, M., et al, 2008, slika 11-25)

Na slici 18 vidimo kako se funkcionalnost bezbednosti dalje izdvaja iz svakog implementiranog veb servisa rezeravcije za svaku bazu podataka. logika sigurnosti je razdvojena na lokalnu komponentu bezbednosti, PDP/PEP generator filtriranih upita. Ova komponenta potvrđuje validnost SAML novčića koji je poslat veb servisu, vrši upit globalne polise od servisa za pronalaženje polisa, vrši upit servisa atributa da ustanovi korisnička prava i zahvajlujući ovim informacijama kreira filtriranu transakciju na bazu podataka. Zbog boljih performansi upit za globalnom polisom se ređe izvršava keširanjem atributa korisnika, na određeni vremenski period.

Na kraju, sigurnosni zahtevi diktiraju ostale aspekte slanja poruka. Ako se zahteva nemogućnost poricanja transakcija, svaki komponenta bi snosila odgovornost digitalnog potpisivanja odgovora veb servisa. Ako je neophodnoa privatnost svake transakcije koristićemo XML enkripciju. Ova studija slučaja predstavlja primer kako koristiti neke od strategija koje su iznete u ovom poglavlju.

# 6.SOA Upravljanje(*Governance*)

SOA upravljanje je ključno za uspešnu implementaciju i praktičnu primenu SOA sistema. Bez njega dobro napravljeni i definisani projekti na papiru propadaju kada treba da se primene u praksi. Potroši se puno novca na razvijanje servisa ali zbog lošeg planiranja ,na svakom koraku primene, se javljaju nepredviđene situacije koje dovode do haosa u sistemu. Menadžeri, arhitekte i projektanti konstantno moraju da prave ispravke i da rade na sistemu koji se već primenjuje. Ova situacija je dobar pokazatelj nemogućnosti uspešnog pravljenja SOA sistema bez SOA upravljanja.

## 6.1.SOA Upravljenje i SOA menadžment

Pojmovi SOA upravljenje i SOA menadžment su dosta isprepletani ali imaju svoje bitne razlike.

SOA upravljanje je kreiranje, komuniciranje, primena i prilagođavanje polisa koje se koriste da kontrolišu kreiranje i implementaciju životnog ciklusa servisa. To je administrativna alatka koja je neophodna i u radu(*run-time*) i pri dizajniranju(*design-time*). Jean Jacques Dubray je u svom članku, koji je izašao oktobra 2007 god. "*Establishing a Service Governance Organization*", pruža dobro opisan zadatak SOA upravljanja.

*The main objective of Service governance is to achieve the benefits of a Service Oriented Architecture by fostering the creation of reusable, enterprise class services. As a cross functional organization, service governance ensures the timely resolution of issues and conflicts due to the necessary tradeoffs that are made when shared requirements are defined.*

Glavni cilj servisnog upravljanja je da postigne prednosti SOA-e pomažući kreiranje ponovo-upotrebljivih servisa na nivou preduzetništva. Kao multi-funkcionalana organizacija servisno upravljanje pruža sigurnost u rešavanju konflikata i problema nastalih usled neophodnih kompromisa, koji su nastali usred definisanja zajedničkih zahteva.

Proces upravljanja u okviru organizacije je zasnovan na centralizovanim polisama(*policy-centric*). Ove polise uključuju i standarde, mogućnosti i ograničenja korišćenja servisa. Upravljanje daje mesto polisama i daje mehanizme njihovog priomenjivanja. Upravljanje nije proces koji je svojstven samo u okviru SOA sistema, može se primeniti na bilo koji deo poslovanja. SOA upravljanje ima neke jedinstvene karakteristike koje se razlikuju od uopštenog upravljanja koje se odnose na životni ciklus servisa.

SOA menadžment uspostavlja kontrolu nad SOE(Service-Oriented Enterprise) sistemom kreiranjem sveukupnog pogleda celog poslovanja a zatim i pružanjem mogućnosti za kontrolom,

nadgledanjem i merenjem informacija o servisima i ostalim komponentama poslovanja. Ovom širom slikom SOA sistema bolje razumemo na koji način se koriste servisi i možemo ih bolje prilagođavati zahtevima poslovanja.

Informacije prikupljne u SOA mendžment procesu mogu direktno uticati na upravljanje. Npr. vreme odaziva koje je garantovano SLA-om može se menjati vremenom na osnovu prikupljenih podataka u radu servisa. Životni ciklus SOA upravljanja se obično odvija u četiri faze:

- **Upravljanje dizajniranjem** - Odnosi se na definisanje i kontrolu servisa i kreiranje polisa koje se koriste za kontrolu implementacije životnog ciklusa servisa. Ključni delovi upravljanja dizajnianjem su:
    - Kreiranje poslovnih polisa koje se koriste za upravljanje implementacijom životnog ciklusa servisa.
    - Kreiranje polisa rada servisa koje određuju ograničenja i mogućnosti servisa

  SOA arhitekra je u ovom delu odgovoran za definisanje i stvaranje polisa za usaglašavanje standarda, zahteve privatnosti, kontrole prava pristupa, pouzdanost, performanse, poruke i razvijanje SLA-a.

- **Upravljanje postavljanjem –** Uključuje procese testiranja i kontrole rada servisa u skladu sa polisama poslovanja da bi se mogli implementirati u SOA sistem.

- **Upravljanje radom –** Primenjivanje i nadgledanje polisa u toku rada servisa i praćenje performansi rada.

- **Upravljanje promenama –** Upravljanje servisima u ciklusu promene. U životnom ciklusu servisa interfejsi, polise i osobine mogu se menjati popriličan broj puta. Upravljanje može implementirati posrednike koji bi rutirali nadolazeće poruke koje su kompatibilne samo sa prethodnim verzijama, na taj način se postiže kontinuitet u radu i pored promena.

Sistem upravljanja se zasniva na polisama. Polise se stvaraju, menjaju i redefinišu kroz životni ciklus projekta. ovo su uobičajeni tipovi polisa koji se koriste u SOA upravljanju:

- **Bezbednost poruka –** Definiše koje se polise odnose na privatnost, integritet i nemogućnost poricanja za svaki servis. Koja će se autentifikacija koristiti, u

kom obliku su novčići koji se koriste i koji standardi se koriste da bi se postigli ovi mehanizmi.

- **Polisa kontrole prava pristupa –** Bavi se time koje su polise korišćene za kontrolu prava pristupa.

- **Prilagođenost poslovnom rečniku i šemi –** Za koje šeme je neophodna podrška da bi se servis postavio, da li postoji zajednički poslovni jezik koji je neophodno primenjivati i da li postoji referentrni model podataka?

- **Prilagođenost tehničkim standardima(WS-I, WSDL, WS-Security,WS-ReliableMessaging) –** Koji su standardi po kojima servis mora da se prilagodi da bi se primenio?

- **Proces primene –** Kakav je proces primene novih servisa?

- **Polisa verzija –** Koje se polise odnose na određivanje verzije servisa i uklanjanjae postojećeg servisa?

- **Polisa otkrivanja –** Koji je proces otkrivanja servisa i određivanja prava pristupa?

- **Propisi privatnosti –** Koji propisi privatnosti se moraju primeniti i na koji način?

- **Kvalitet servisa(QoS-Quality of Service) –** Da li postoji garantovani rok odgovora za svaki servis? Da li postoje različiti nivoi odaziva servisa i prioriteti po klasifikaciji korisnika? Da li se prioriteti primenjuju u odnosu na tip korisnika ili individualno? Da li je moguć dogovor oko nivoa rada servisa ,u toku rada, u odnosu na zahteve potražioca?

- **Pouzdanost –** Da li postoje polise koje se odnose na isporuku po narudžbini, bar jednom isporuka i najviše jednom isporuka?

- **Zahtevi izveštavanja i pregleda –** Koji tip pregleda i izveštavanja je neophodno da svaki servis podrži?

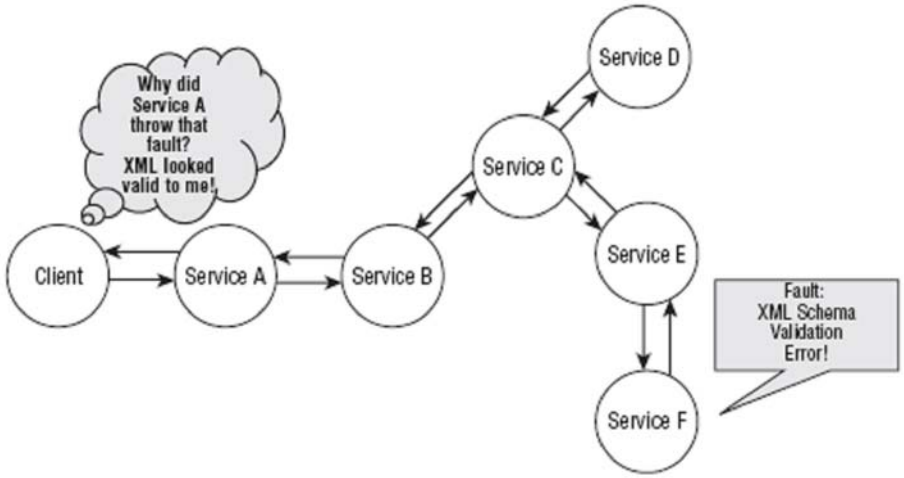
- **Dogovori o nivou servisa(SLAs-Service Level Agreements) –** Da li postoji formalni dogovor između korisnika i ponuđača servisa o vremenu odgovora servisa itd.?

Ove polise se ne moraju odnositi na svaki SOA sistem, svako poslovanje može imati drugačije zahteve. Bitno je napomenuti i to da polise SOA upravljanja ne moraju biti preplavljujuće i detaljne, mali broj dobro organizovanih polisa može puno toga uraditi. Ključno je to da zahtevi poslovanja diktiraju koji nivo upravljanja će se primenjivati. Neki su čak neophodni zato što tako nalaže zakonska regulativa. Usvajanje uspešne strategije SOA upravljanja znači mogućnost da se arhitektura poslovanja razvija planski a ne stihijski.

## 6.2.Neophodnost šire slike poslovanja

Najveća dobit implementacijom SOA sistema je da servisi postaju komponente koje se mogu ponovo koristiti i koje se kombinuju sa drugim servisima da bi se naravile funkcionalne aplikacije. Kao rezultat postoje mnoge permutacije načina korišćenja servisa. Ovo dovodi do rasta situacija u kojima se javlja nepravilnost u radu i zato je neophodno imati centralizovani uvid u široku sliku SOA sistema. Na osnovu ove široke slike lakše se detektuju i rešavaju problemi i usvajaju druga stanovišta rada servisa koja bolje odgovaraju realnim uslovima.

Jednostavan primer neophodnosti ovakvog pregleda sistema može biti lociranje greške kada se ona pojavi u radu. Kada servis javi grešku ona može biti poprilično neinformativna da bi nam ukazala zašto se javio problem. Da bi se rešio problem neophodan je detaljan pregled logova da bi se samo locirao uzrok problema. Slika 19 prikazuje scenario koji demonstrira ovaj primer.



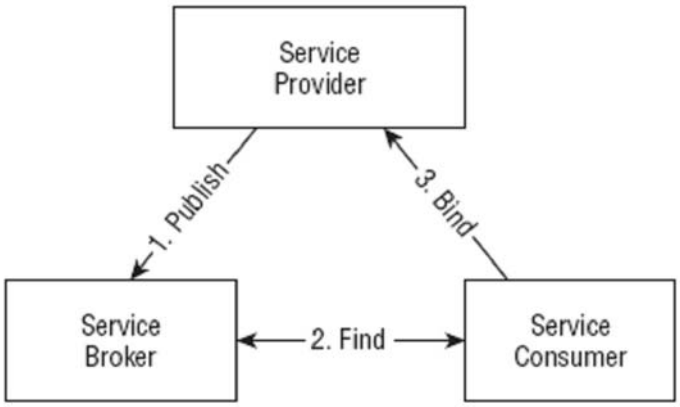

Slika 19: Primer scenarija koji pokazuje neophodnost centralnog pregleda SOA sistema (Rosen, M., et al, 2008, slika 12-1)

Klijent šalje SOAP zahtev servisu A, neophodo je rutiranje poruke sve do servisa F da bi se izvršio zahtev. Servis F javlja grešku zato što je servis E neispravno kreirao zahtev. U ovom primeru svaki servis vraća SOAP grešku onom ko ga je pozvao. U takvom okruženju, gde ne

postoji centralizovan pregled, je otkrivanje korena problema moguće samo pregledom log zapisa o radu servisa koji su uključeni u životni ciklus zahteva. Ovaj primer jasno pokazuje koliko je praktičnije imati centralizovan uvid u širu sliku celokupnog poslovanja.

## 6.3.Neophodnost primene polisa u radnom okruženju servisa

Na slici 20 je prikazan trougao "*publish-find-bind*", koji je dobra ilustracija kako veb servisi mogu da funkcionišu u saradnji sa UDDI i ostalim registrima. U praksi pružaoc servisa obično registruje svoj WSDL kod servis brokera, koji čuva informacije u registru. Korisnik servisa se registruje kod brokera da bi pronašao veb servis. Konačno korisnik preuzima WSDL servisa i dinamički se veže na veb servis.



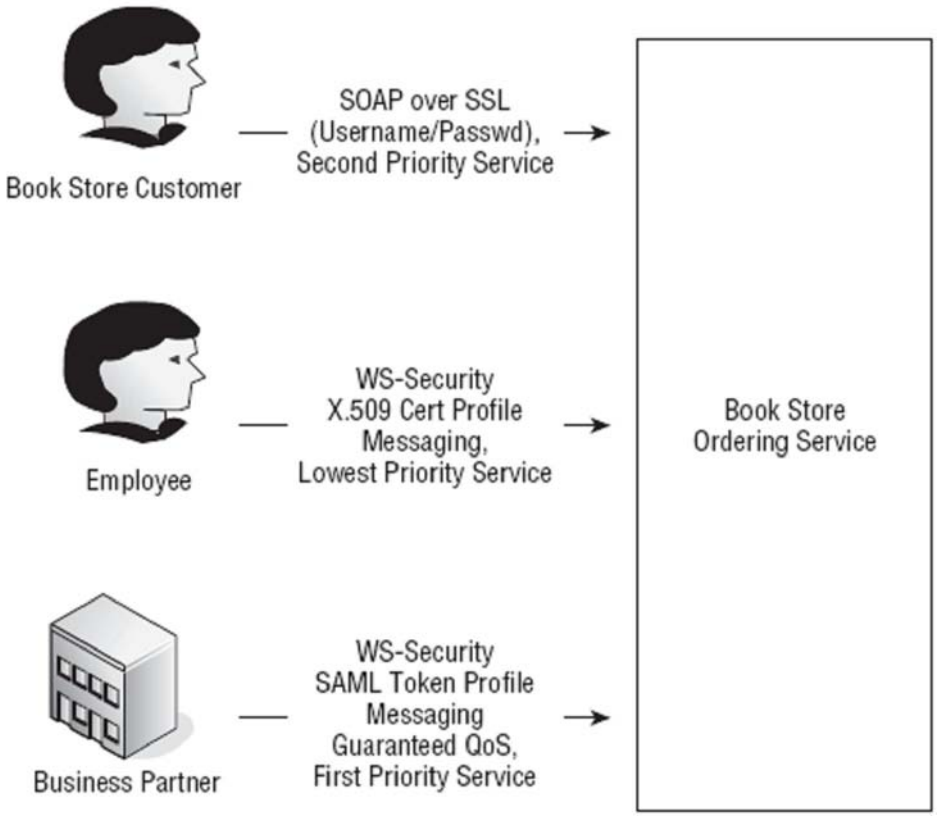

Slika 20: "*publish-find-bind*" trougao (Rosen, M., et al, 2008, slika 12-2)

Konceptualno ovaj apstraktni model je, na pri pogled, dobar ali zato je izuzetno manjkav i nedovoljan u praktičnoj primeni. U realnom okruženju je neophodno mnogo više za povezivanje sa servisom pored poznavanje sintakse interfejsa. U operacionom okruženju postoje razne polise sigurnosti, transporta, autorizacije i garancije nivoa rada servisa koje se baziraju na različitim mušterijama, partnerima, dobavljačima i zaposlenima koji stupaju u interakciju sa servisima. Korisnici servisa moraju razumeti ove polise kao i semantiku rezultata rada ovih servisa. Pružaoci servisa moraju da obezbede primenu ovih polisa u radu i obe strane se moraju prilagođavati promenama polisa i načina rada servisa, koje su neizbežne.

# 6.4.Neophodnost razdvajanja logike polisa od logike poslovanja

Postoje ljudi koji se u principu slažu sa SOA upravljanjem i veruju da je neophodno imati polise koje su vezane za upravljanje dizajniranjem ali da u vreme rada i promene one nisu neophodne. Smatra se da se ovo može postići dobrim kodiranjem i spajanjem logike rada i poslovne logike servisa. Ovo bi zahtevalo čestu promenu koda servisa.

Polise se vremenom menjaju i menjanje koda servisa u skladu sa polisama zvuči teoretski moguće ali u praksi je to teško izvodljivo. Menjanje koda bi trebalo da se izvrši svaki put pri promeni polise a svaka promena servisa bi zahtevala promenu kod klijenata. Dodatna komplikacija se javlja kada shvatimo da postoje različite polise za za raličite ljude u jednom istom servisu. Slika 21 prikazuje kako jednostavan servis može imati različite polise u zavisnosti od tipa mušterije koji se koristi servisom. Knjižara poseduje servis za naručivanje knjiga, njime se mogu koristiti mušterije, poslovni partneri i zaposleni.



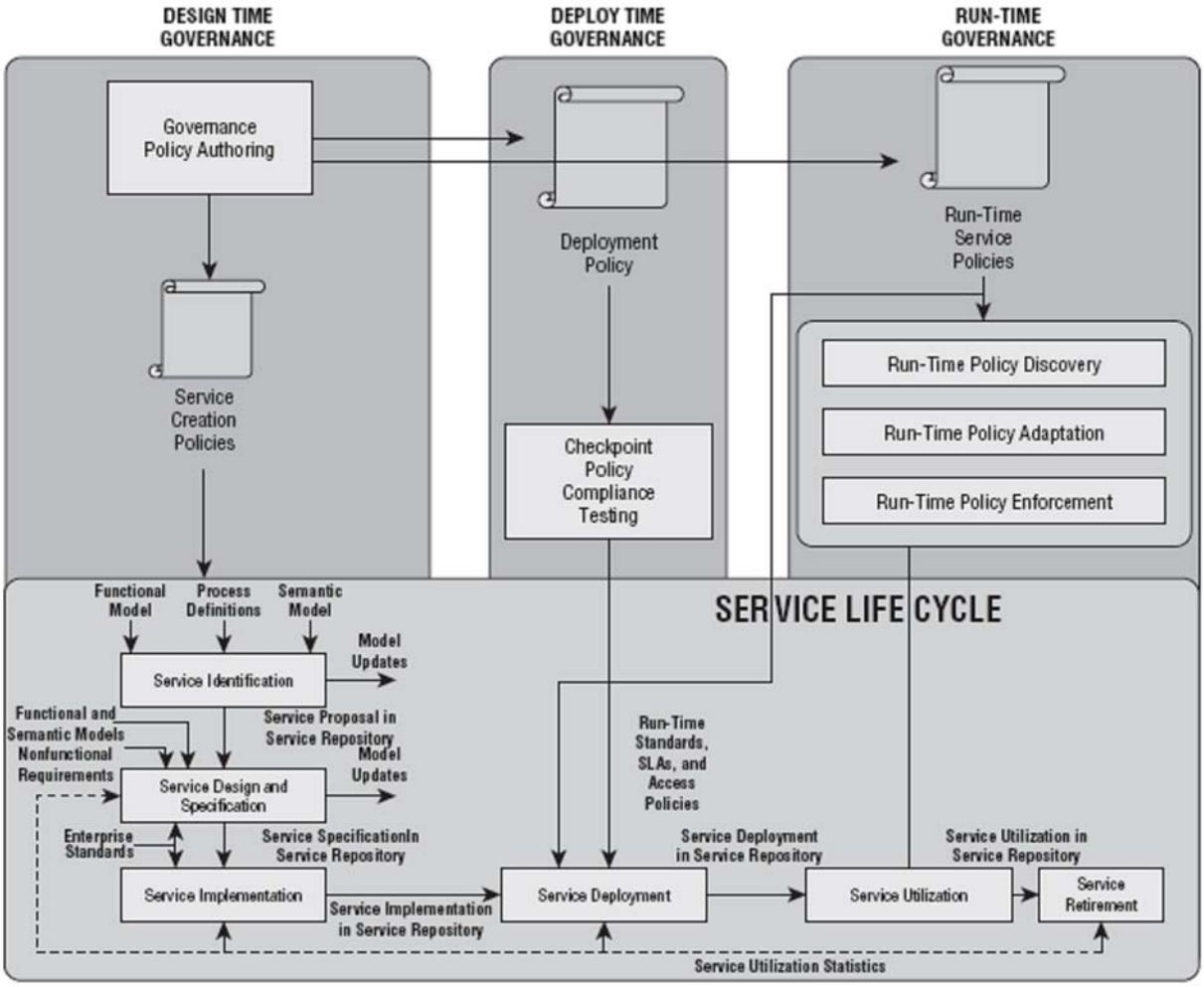

Slika 21: Prikaz upotrebe različitih polisa za različite mušterije (Rosen, M., et al, 2008, slika 12-3)

Svaka mušterija zahteva drugačije sigurnosne zahteve i dobija različite prioritete korišćenja servisa. Poslovni partneri imaji zagarantovan kvalitet usluge servisa, najviši prioritet i koriste *WS-Security* SAML profil sa novčićem za autentifikaciju. Mušterije koriste obične SOAP poruke preko SSL-a sa korisničkim imenom i šifrom, prioritet im je ispod poslovnih partnera. Zaposleni koriste WS-Security X.509 profil sertifikata preko SSL-a i imaju najniži prioritet. Kada servis dobije red zahteva on odgovara na zahteve na osnovu prioriteta i mora primeniti odgovarajući sigurnosni mehanizam za svaki zahtev.

Ovo predstavlja jednostavan primer primene različitih polisa. Ali i pored jednostavnosti vidimo da je poprilično teško uraditi kodiranje ako su logika polisa i poslovna logika rada servisa spojeni. Efektivan način da se ovo izbegne je da se polise naprave u formatu koji je razumljiv za softver i korišćenjem mehanizama upravljanja u radu, da bi se primenjivale polise na strani servera i omogućilo otkrivanje polisa na strani klijenta.

## 6.5.SOA upravljanje u životnom ciklusu servisa

Životni ciklus SOA upravlanja i životni ciklus servisa su prikazani na slici 22 kao dva paralelena procesa, SOA upravljanje vodi servis kroz ceo njegov životni ciklus.




Slika 22: SOA upravljanje u životnom ciklusu servisa (Rosen, M., et al, 2008, slika 12-4)

Tim ljudi koji se bave SOA upravljanjem diktira procese koji se javljaju u životnom ciklusu servisa kao i uloge i odgovornosti ljudi koji rade na SOA sistemu. Procesi i uloge ljudi se razlikuju od organizacije do organizacije ali postoje sličnosti i tipične uloge i procesi koje je teško izostaviti u bilo kojoj organizaciji. SOA upravljanje se odražava na celu organizaciju i jedna osoba može izvršavati više uloga istovremeno. Uobičajene uloge su:

- **Vođa rešenja –** Poslovni arhitekta zadužen za određeni poslovni problem.

- **Arhitekta funkcionalnosti –** Arhitekta koji održava i poboljšava funkcionalni model poslovanja. Održava rečnik poruka koji obezbeđuje interoperabilnost između servisa i stara se o usklađivanju servisa sa funkcionalnim modelom.

- **Arhitekta portfolia –** Upravlja portfoliom servisa. Održava kontakte sa poslovnim jedinicama kao i sa drugim portfolijima. Ekspert za neku oblast servisa u okviru portfolia.

- **Preduzetnički arhitekta –** Igra pivotirajuću ulogu u identifikaciji i dizajnu servisa. Stara se o tome da se svaki servis uklapa u celokupni predzetnički kontekst. Odgovorn je za pristupe u integraciji i dizajn adaptera. Bira odgovarajuće platforme implementacije servisa i odgovoran je za nefunkcionalne zahteve servisa i njihove implementacije.

- **Arhitekta poslovnih procesa –** Upravlja poslovnim procesima i njihovom evolucijom. Blisko sarađuje sa arhitektom funkcionalnost, starajući se o usklađenosti servisa sa trenutnim i budućim poslovnim procesima.

- **Bibliotekar servisa –** Osoba odgovorna za održavanje registra servisa, centralni deo upravljanja servisima. Stara se o tome da su sve informacije o servisima u standardizovanoj formi, tačne, ažurne i pravilno klasifikovane.

- **Arhitekta SOA prometa –** Odgovoran za održavanje i poboljšanje rada servisa, uključujući standardizaciju API-a(Application programming interface).

- **Autor aplikacija –** Osoba koja piše i održava poslovnu funkcionalnost osnovnih servisa.

- **Asembler servisa –** Održava poslovnu funkcionalnost kompozitnih servisa tj. poslovnih procesa.

- **Tester servisa –** Odgovoran je za kvalitet i testiranje rada servisa.

- **Specijalista za infrastrukturu servisa –** Inžinjer infrastrukture odgovoran za kreiranje dijagrama i topologija smeštanja servisa i održavanja registra servisa. Takođe prati korišćenje servisa i pridržavanje SLA rada servisa.

SOA upravljanje direktno utiče na ljude iz drugih delova organizacije. Polise, procesi i procedure, koje se razviajju SOA upravljanjem, moraju biti razumljive i propratiti servis kroz ceo njegov životni ciklus.

# 7.Zaključak

Pojavom XML-a je stvorena mogućnost interkomunikacije sa, do tada, nespojivim aplikacijama i sistemima koje su radile kao nezavisne usluge. Ove aplikacije su donele velike promene u radu bilo koje oblasti, bilo da je u pitanju knjigovodstvo, računovodstvo, projektovanje, proračunavanje, pisanje, prezentovanje itd. U kombinaciji sa bazama podataka su ubrzale rad višestruko i tako omogućile efikasnije poslovanje. Rastom poslovanja zahtevi su postajali veći, informacioni sistem sve glomazniji i teži za održavanje. Neophodne investicije su postajale veće i to samo da bi se održala trenutna funkcionalnost sistema. Ulaganje u inforacioni sistem plako počinje da biva opterećenje za budžet a ne smanjenje troškova. Do ove situacije je dovelo odsustvo dobrog planiranja i sve veća zavisnost poslovanja od IT sektora. XML, kao i jezici koji se baziraju na njemu, je doneo mogućnost interkomunikacije u sistemu i mogućnost spajanja i boljeg praćenja sistema kao celine.

SOA nudi rešenja i smernice za proces integracije, separacije i nezavisnoti poslovnih procesa od ograničenja IT-a, neophodno povećanje sigurnosti i centralizovanim upravljanjem. Centralizovanim upravljanjem se dobija bolji uvid u širu sliku. Ovom jasnijom i širom slikom celokupnog sistema može se bolje planirati, lakše intervenisati i lakše postići nezavisnost poslovnih procesa od tehnoloških procesa. Bitno je napomenuti da se u SOA rešenje ne kreće od celine, nego se radi parcijalna integracija. Pažljivim planiranjem se deo po deo sistema integriše i postepeno se implementiraju nove koponente sistema.

I pored toga, što ideja nije nova i činjenice da SOA predstavlja samo proširenje ideje veb servisa, začuđujuće je to kako se retko, gotovo nikad ne srećemo sa ovim terminom. Moje lično mišljenje je da ona predstavlja tihu revoluciju koja uopšte ne dotiče korisnike, na način kao bilo koja druga revolucija. Korisnici će primeteti praktičniji rad, bolje usluge, veću stabilnost, fleksibilnost, ređe kvarove i generalno bolju funkcionalnost željene usluge. Naravno, malo njih će se zapitati šta se promenilo u radu i šta je dovelo do svega toga. Tako da je moj lični utisak da dobar deo, ne samo javnosti nego i stručne javnost, nije svestan šta je to SOA, šta donosi, koje mogućnosti pruža i kako može uticati na kvalitet i način celokupnog poslovanja.

Uprkos visokim zahtevima, velikim ulaganjima, dužim planiranjem, pažljivijim projektovanjem, neophodnosti angažmana celokupnog menadžmenta i potrebom za dokazivanjem da je isplativa invensticija, SOA je tehnologija budućnosti. Pitanje sutrašnjice neće biti da li nam je potrebna SOA, nego kada i kako početi rad na implementaciji.

# LITERATURA:

## Knjige:


1. Hurwitz, Judith, Robin Bloor, Carol Baraudi i Maricia Kaufman (2007): *Service oriented Architecture for Dummies*, Wiley Publishing, Inc., Indianapolis, Indiana
2. Rosen, Mike, Boris Lublinsky, Kevin T. Smith i Marc J. Balcer (2008): *Applied SOA: Service-Oriented Architecture and Design Strategies*, Wiley Publishing, Inc., Indianapolis, Indiana.
3. Carter, Sandy (2007): *The New Language of Business SOA & Web 2.0*, Pearson plc Publishing as IBM Press.
4. Juric, Matijaz B., Poornachandra Sarang, Ramesh Loganathan i Frank Jennings (2007): *SOA Approach to Integration: XML, Web services, ESB, and BPEL in real-world SOA projects*, Packt Publishing Ltd.32 Lincoln RoadOltonBirmingham, B27 6PA, UK.
5. Erl, Thomas (2008): *SOA: Principles of Service Design,* Pearson Education, Inc. Rights and Contracts Department 501 Boylston Street, Suite 900 Boston.
6. Marks, Eric.A. i Michael Bell (2006): *Executive's guide to service-oriented architecture*, John Wiley & Sons, Inc., Hoboken, New Jersey.
7. Erl, Thomas (2005): *SOA Design Patterns*, Prentice Hall/PearsonPTR.


## Internet adrese:


1. http://en.wikipedia.org
2. http://bobbreedlove.com
3. http://blog.objectmentor.com
4. http://www.w3.org
5. http://www.oreillynet.com
6. http://www.soauniverse.com
7. http://www.workflowgen.com
8. http://www.jboss.org
9. http://www.opensymphony.com
10. http://ode.apache.org
11. http://www.w4.eu
12. http://www.workflowcommunity.com
13. http://www.webopedia.com
14. http://www.infoq.com

# BESPLATNI GOTOVI SEMINARSKI, DIPLOMSKI I MATURSKI RAD.

**RADOVI IZ SVIH OBLASTI, POWERPOINT PREZENTACIJE I DRUGI EDUKATIVNI MATERIJALI.**

## WWW.SEMINARSKIRAD.ORG

## WWW.MATURSKIRADOVI.NET

## WWW.MATURSKI.NET

## WWW.SEMINARSKIRAD.INFO

## WWW.MATURSKI.ORG

## WWW.ESSAYSX.COM

## WWW.FACEBOOK.COM/DIPLOMSKIRADOVI

NA NAŠIM SAJTOVIMA MOŽETE PRONAĆI SVE, BILO DA JE TO SEMINARSKI, DIPLOMSKI ILI MATURSKI RAD, POWERPOINT PREZENTACIJA I DRUGI EDUKATIVNI MATERIJAL. ZA RAZLIKU OD OSTALIH MI VAM PRUŽAMO DA POGLEDATE SVAKI RAD, NjEGOV SADRŽAJ I PRVE TRI STRANE TAKO DA MOŽETE TAČNO DA ODABERETE ONO ŠTO VAM U POTPUNOSTI ODGOVARA. U BAZI SE NALAZE GOTOVI SEMINARSKI, DIPLOMSKI I MATURSKI RADOVI KOJE MOŽETE SKINUTI I UZ NJIHOVU POMOĆ NAPRAVITI JEDINSTVEN I UNIKATAN RAD. AKO U BAZI NE NAĐETE RAD KOJI VAM JE POTREBAN, U SVAKOM MOMENTU MOŽETE NARUČITI DA VAM SE IZRADI NOVI, UNIKATAN SEMINARSKI ILI NEKI DRUGI RAD RAD NA LINKU IZRADA RADOVA. PITANjA I ODGOVORE MOŽETE DOBITI NA NAŠEM FORUMU ILI NA MATURSKIRADOVI.NET@GMAIL.COM