

PODRŠKA IZVRŠENJU MANUELNIH AKTIVNOSTI U ORKESTRACIJI SERVISNO-ORIJENTISANE ARHITEKTURE¹

SUPPORT FOR EXECUTION OF MANUAL ACTIVITIES IN ORCHESTRATION OF SERVICE-ORIENTED ARCHITECTURE

Milan Zdravković¹, prof. dr Miroslav Trajanović¹, Nikola Vitković¹
¹*Mašinski fakultet Univerziteta u Nišu*

¹ Rad predstavlja jedan od rezultata istraživanja projekta TR6215 - "Planiranje, terminiranje i adaptibilno upravljanje proizvodnim procesima", finansiranog od strane Ministarstva za nauku i zaštitu životne sredine, u okviru programa istraživanja u oblasti tehnološkog razvoja

Sadržaj – U radu je predstavljen projekat razvoja generičkog WSDL klijenta, koji treba da omogući distribuciju funkcionalnog korisničkog interfejsa, generisanog na osnovu semantičkog modela interoperabilnosti jednog web servisa. Korisnički interfejs je namenjen izvršenju manuelnih aktivnosti BPEL orkestracije.

Abstract – *This paper presents a project of development of generic WSDL client, which main goal is to enable distribution of functional user interface, generated on basis of semantic interoperability model of web service. User interface is intended to be use for manual activities in BPEL orchestration.*

1. UVOD

U servisno orijentisanoj arhitekturi, web servisi predstavljaju strukturni sloj koji primarno obezbeđuje integraciju korporativnog informacionog sistema, unutar granica poslovnog sistema i van njih. BPEL orkestracijom web servisa u poslovne procese, ovaj primarni cilj se formalno ostvaruje i to, u uslovima u kojima web servisi međusobno komuniciraju – svaki od njih može da igra ulogu i provajdera i korisnika servisa.

Sa stanovišta realizacije servisno orijentisanog sloja postavljenog iznad konvencionalnog informacionog sistema, ova međusobna komunikacija se inicira i izvršava automatski, pri čemu odgovornost za manuelne aktivnosti u podršci odvijanju poslovnih procesa, uobičajeno pripada aplikacionom sloju konvencionalnog informacionog sistema, odnosno njegovom korisničkom interfejsu.

U okolnostima razvoja korporativnog informacionog sistema, u kojima je primarni cilj - integrisanje poslovnih procesa, veoma često se u procesu projektovanja ustanovi da postojeće aplikaciono okruženje ne podržava neku funkciju, predviđenu projektom servisno orijentisane arhitekture. Razvoj

nove funkcije podrazumeva reviziju konvencionalnog informacionog sistema.

Ovaj zadatak može da bude izuzetno složen, naročito ukoliko se radi o zastareloj (*legacy*) aplikaciji, i da značajno uspori implementaciju SOA. Dalje, ukoliko je nova funkcija - podrška nekoj manualnoj aktivnosti poslovnog procesa, postupak razvoja se dodatno usložnjava, jer je, u tom slučaju, potrebno razviti i korisnički interfejs i integrisati ga sa razvijenom poslovnom logikom.

Opšti cilj projekta razvoja generičkog WSDL klijenta je generalna podrška manualnim aktivnostima u orkestraciji servisno orijentisane arhitekture. On obuhvata rezultate koji treba da demonstriraju mogućnosti korišćenja generičkih metoda za projektovanje web servisa, generisanje korisničkog interfejsa za pozivanje web servisa i odgovarajući prikaz odgovora, autentifikaciju poziva servisa, odnosno, *role-based* sistem za upravljanje njegovim korisnicima.

Specifični cilj razvoja generičkog WSDL klijenta u trenutnoj fazi projekta je generisanje HTML forme za pozivanje bilo kog web servisa iz internog registra i bazična podrška aktivnostima održavanja ovog registra.

Osnovni principi razvoja generičkog WSDL klijenta su:

1. Intuitivan korisnički interfejs koji "skriva" složene elemente logike servisno orijentisane arhitekture, odnosno, web servisa;
2. Funkcionalan korisnički interfejs koji omogućava efikasnu komunikaciju manuelnih aktivnosti poslovnih procesa sa web servisima, validacijom unetih podataka prilikom konstruisanja poruka za poziv web servisa;
3. Analiza meta podataka servisa - WSDL i XSD elemenata strukture, u cilju generisanja korisničkog interfejsa za pozivanje web servisa;

4. Podrška jednodimenzionalnim nizovima elemenata strukture i složenim strukturama (tipovima podataka) u dva nivoa;
5. Korišćenje web arhitekture i otvorenih standarda;
6. Modularnost arhitekture generičkog WSDL klijenta u cilju ostvarivanja funkcije višestrukog korišćenja generisanih formi u različitim poslovnim okolnostima;
7. Autentifikacija korisnika web servisa.

- Generisanje HTML forme za unos parametara (*input* poruke) za poziv izabrane operacije web servisa;
- Generisanje SOAP zahteva na osnovu kreirane input poruke;
- Slanje SOAP poruke, prijem SOAP odgovora i prikaz u odgovarajućem formatu;
- Praćenje i kontrola saobraćaja poziva distribuiranih formi i SOAP poruka;
- Upravljanje internim registrom servisa;
- Upravljanje internim registrom korisnika servisa;
- Autentifikacija pristupa administraciji generičkog WSDL klijenta;
- Autentifikacija poziva distribuirane forme.

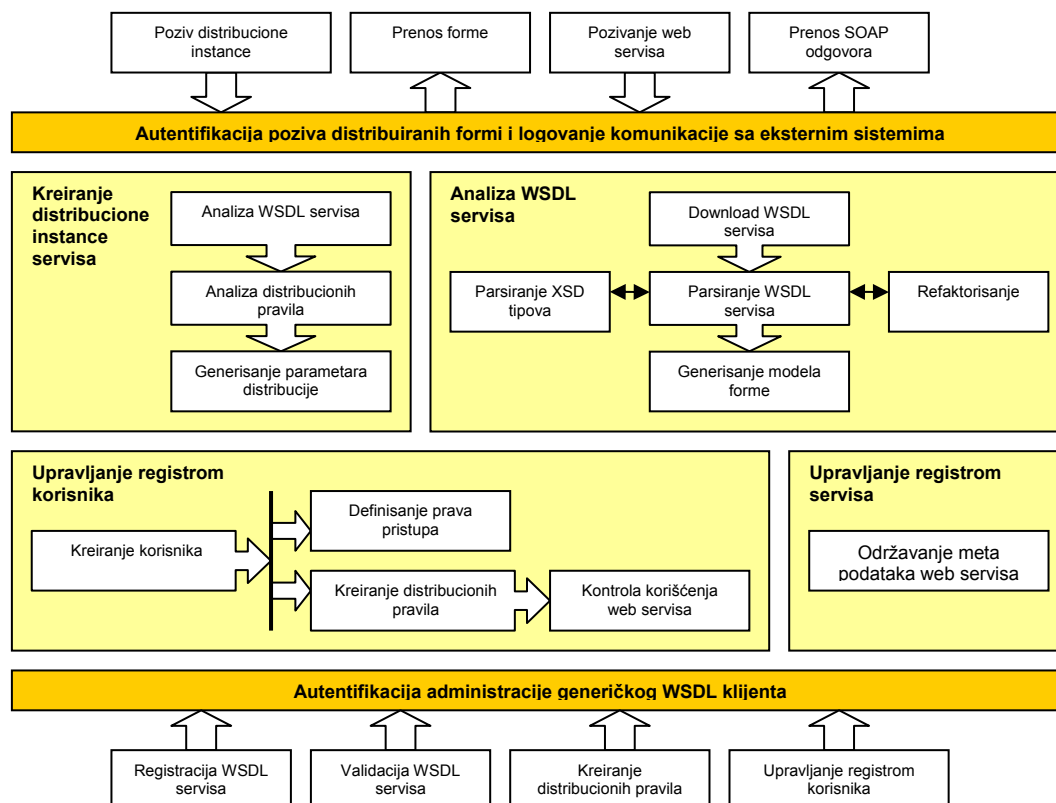
2. FUNKCIONALNA ARHITEKTURA GENERIČKOG WSDL KLIJENTA

Osnovnu aktivnost projektovanja generičkog WSDL klijenta predstavlja njegova funkcionalna analiza. Ona se vrši na osnovu definisanih opštih i specifičnih ciljeva, odnosno principa za njihovo ostvarivanje.

Funkcije WSDL klijenta, u trenutnoj fazi razvoja su:

- Analiza (parsiranje) WSDL interfejsa web servisa prema definisanom postupku;
- Analiza (parsiranje) XSD šeme, referencirane iz WSDL interfejsa;

Na osnovu definisanih osnovnih principa razvoja generičkog WSDL klijenta, projektovana je i realizovana njegova osnovna funkcionalna arhitektura, prikazana na slici 1.



Slika 1. Funkcionalna arhitektura generičkog WSDL klijenta

Za realizaciju prikazane funkcionalne arhitekture, neophodno je obezbediti skladištenje određenog skupa podataka u okviru svake instance klijenta i njegovu slabu vezu sa semantičkim modelom interoperabilnosti

web servisa. U relacionoj bazi podataka klijenta se čuvaju informacije iz sledećih domena:

1. internog registra korisnika, koji obuhvata osnovne autorizacione profile korisnika i modele distribucije pojedinačnih operacija

servisa. Modele distribucije, koji pripadaju jednom korisniku generičkog WSDL klijenta, čine prava pristupa i generički skup distribucionih pravila.

2. internog registra servisa – meta podataka web servisa;
3. praćenja i kontrole saobraćaja, koji obuhvata *log* korišćenja klijenta, arhivu razmenjenih poruka, generisanih grešaka i aktiviranih kompenzacionih mehanizama.

Skladištenje podataka generičkog WSDL klijenta je realizovano primenom MySQL relacione baze podataka.

3. IZBOR TEHNOLOGIJE RAZVOJA GENERIČKOG WSDL KLIJENTA

Integrisanje kompletnog poslovnog okruženja u servisno orijentisanu arhitekturu i njeno orkestriranje u okvirima jasno definisanih poslovnih procesa, pored direktnih ušteda, ostvarenih u segmentima ubrzanja komunikacije, povećanja njenog kvaliteta i dostupnosti podataka, podrazumeva i generalnu obavezu za formalnim uređenjem poslovnog sistema. Revizijom poslovnog sistema, sa stanovišta procesnog uređenja, ostvaruju se uslovi za dodatne uštede, kroz povećanje efikasnosti i produktivnosti poslovanja.

Sa druge strane, razvoj B2C i B2B poslovanja nameće potrebu za sve veći stepen web orijentacije poslovnih informacionih sistema. Temelj web komunikacije unutar granica poslovnog sistema, između njegovih udaljenih segmenta, odnosno web komunikacije sa partnerima poslovnog sistema i samim klijentima, predstavljaju savremeni sistemi za upravljanje sadržajem (*Content Management System-i*) i web kolaboracioni alati. Oni su osnovno sredstvo za realizaciju intraneta preduzeća – tehničkog okvira za upravljanje i izvršenje poslovnih procesa, kao i internu i eksternu komunikaciju.

Razvoj *open-source* tehnologija, ali i tržišta aplikacija otvorenog koda, preduzećima predstavlja izuzetan motiv da investiraju sredstva i resurse u osavremenjavanje poslovnog informacionog sistema. On je, pre svega, značajan izazov segmentu malih i srednjih preduzeća, kojeg uobičajeno karakteriše nizak nivo informatizacije, jer obezbeđuje dostupnost visokotehnoloških sredstava za uređenje poslovanja i web kolaboraciju, uz mnogo manje potrebne investicije u odnosu na korišćenje komercijalnih IT alata. Najprisutnija tehnologija na tržištu web aplikacija za upravljanje sadržajem i web baziranih kolaboracionih alata je LAMP (Linux-Apache-MySQL-PHP) model. Veliki broj *open-source* CMS-ova, zasnovanih na ovoj tehnologiji, se danas koristi za organizaciju intraneta, čak i u velikim poslovnim sistemima.

Navedene okolnosti su determinisale i izbor tehnologije za realizaciju generičkog WSDL klijenta. S obzirom na to da je on namenjen posredovanju između korporativnih servisa, realizovanih *enterprise*

tehnologijama i izvršioca manualnih aktivnosti poslovnih procesa, čije se sprovođenje vrši u okviru intraneta preduzeća, za njegov razvoj je korišćen PHP jezik i MySQL baza podataka.

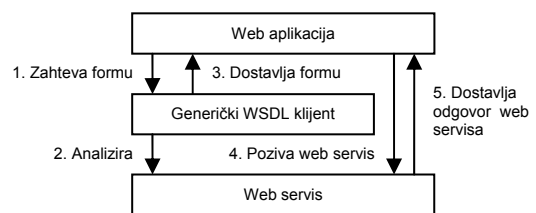
Primenom LAMP tehnologije za razvoj generičkog WSDL klijenta, ostvaruje se:

- Brzi razvoj prototipa, čime se obezbeđuju uslovi za ranu validaciju korisnosti, upotrebljivosti i pouzdanosti klijenta;
- Visoke performanse procesa distribucije formi, praćenja i kontrole saobraćaja između direktnih korisnika servisa i njegovog provajdera;
- Brzo i jednostavno integrisanje u sisteme za upravljanje sadržajem i kolaboracione alate koji čine intranet preduzeća;
- Tehnološku unificiranost i standardizovanost intraneta preduzeća, što obezbeđuje njegovu jednostavnije i lakše održavanje.

4. DISTRIBUCIJA FORMI ZA KOMUNIKACIJU SA WEB SERVISIMA

Iako je specifični cilj razvoja generičkog WSDL klijenta – generisanje HTML forme za pozivanje bilo kog web servisa, on se koristi i kao posrednik u ostvarivanju komunikacije različitih elemenata poslovnog okruženja sa samim servisom.

Naime, jedna od njegovih uloga je i distribucija klijentskih formi u web aplikacije, koje čine informacionu infrastrukturu preduzeća, kao što su javni web sajt, intranet web portal, web CRM sistem, itd. U tom smislu, može se zaključiti da generički WSDL klijent predstavlja poseban sloj komunikacije u servisno orijentisanoj arhitekturi.



Slika 2. Pozicija generičkog WSDL klijenta u servisno orijentisanoj arhitekturi

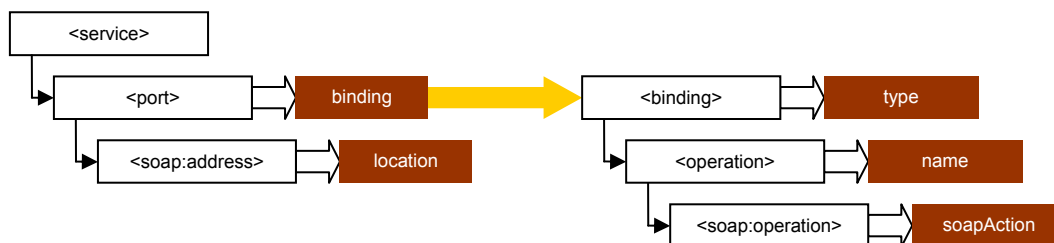
Očigledno je da generički WSDL klijent ima ulogu korisnika servisa u procesu njegove analize, dok web aplikacija kojoj se distribuiraju forme predstavlja njenog krajnjeg korisnika. Osnovna uloga generičkog WSDL klijenta u ovim uslovima je *dodavanje vrednosti atributa upotrebljivosti web servisa*, i to u različitim okruženjima njihove standardizacije.

Ostvarivanjem ove uloge, omogućava se višestruko korišćenje jedinstvene implementacije poslovne

funkcije, odnosno, web servisa za obavljanje iste manuelnih aktivnosti u različitim uslovima.

5. POSTUPAK ANALIZE WSDL INTERFEJSA I GENERISANJE FORMI

Informacija o servisu je u WSDL interfejsu predstavljena `<service>` strukturom. Postupak

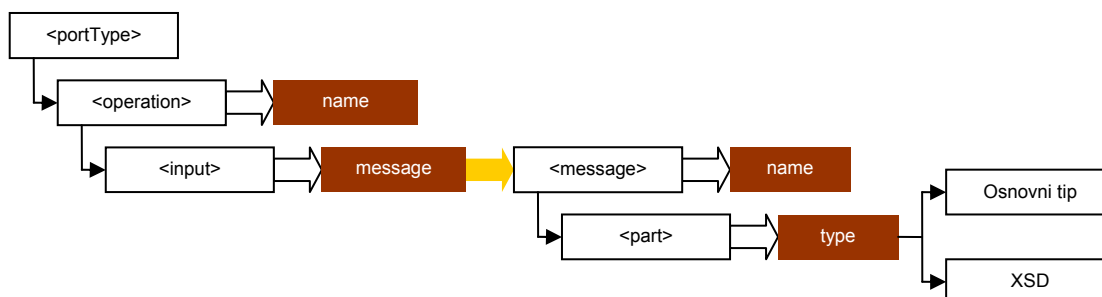


Slika 3. Analiza tipova mrežnih završetaka i operacija jednog WSDL interfejsa

Na osnovu *binding* atributa `<port>` strukture, parsiraju se `<binding>` strukture WSDL interfejsa, na osnovu čijih se *type* parametara identifikuju tipovi mrežnih završetaka. Operacije koje podržava svaki od mrežnih završetaka se identifikuju izdvajanjem *name* atributa `<operation>` elemenata. U procesu parsiranja

`<binding>` strukture se identifikuju i *soapAction* atributi `<soap:operation>` elemenata.

Nakon identifikacije tipova mrežnih završetaka i operacija servisa, vrši se analiza meta strukture WSDL interfejsa u cilju identifikacije tipova poruka.



Slika 4. Analiza strukture parametara za poziv operacija servisa

Postupak analize strukture parametara za poziv operacija servisa je prikazan na slici 4. Struktura pozivne poruke web servisa je definisana u samom WSDL interfejsu. Njeni delovi (`<part>` element) mogu biti prostih tipova, ili složenih, čija definicija se može nalaziti u XSD šemi, definisanoj `<import>` deklaracijom u WSDL interfejsu.

Da bi se ostvarili uslovi za razmenu složenih poruka, interfejs omogućava privremenu serijalizaciju pripremljene pozivne poruke u sesiji web browsera. Ukoliko je u okviru neke poruke definisan kardinalni element, prostog ili složenog tipa, njegova vrednost se definiše nakon definisanja vrednosti ostalih elemenata i snimanja pozivne poruke u sesiju.

XSD šema se analizira u cilju identifikovanja prostih (`<xs:simpleType>`) i složenih (`<xs:complexType>`) tipova. Nakon inicijalne analize strukture XSD šeme, vrši se njeno iterativno refaktorisanje sve do trenutka kada su svi elementi prostih tipova. Ukoliko je definisana kardinalnost (broj mogućih pojavljivanja u poruci) nekog elementa, primenom njegovih *minOccurs* i *maxOccurs* atributa, vrši se generisanje složene forme, koja sadrži podformu za definisanje njegovih vrednosti.

Korisnik web servisa ima na raspolaganju alate, koji mu omogućavaju da u sesiji sačuva neograničeni broj pozivnih poruka, različitih operacija istog ili različitih web servisa. Sačuvane pozivne poruke se mogu poslati u bilo kom trenutku rada sa generičkim WSDL klijentom. Složena forma koja demonstrira ovakav princip rada je prikazana na slici 5.

WSDL Registry breakdown	WSDL Operation	Show Message
WSDL:http://localhost/xsli/OrderProcessing.wsdl		
PortType:InvoiceCallbackPT		
PortType:PurchaseOrderPT		
Operation:receivePO [1 message]		
Message 1 (2006-12-24 12:44:19) <i>Multiple:POItem</i> Message 1 (2006-12-24 12:44:31) Message 2 (2006-12-24 12:44:40)		
Operation:receivePOItem		
Operation:requestAccountCreditRating		
Operation:requestAccountRegularityRating		
Operation:requestPOProcessing		
Operation:sendInvoice		
Operation:sendShippingConfirmation		
Operation:sendShippingSchedule		
PortType:RiskAnalysisPT		
PortType:ShippingCallbackPT		
WSDL:http://localhost/xsli/FinancialManagement.wsdl		
WSDL:http://localhost/xsli/ProductionPlanning.wsdl		
WSDL:http://localhost/xsli/ProductCatalog.wsdl		
WSDL:http://localhost/xsli/StrategicPlanning.wsdl		

receivePO (POMessage) - Message 1		
SingleDelivery	<input type="checkbox"/> No	(boolean)
SpecialDelivery	<input checked="" type="checkbox"/> Yes	(boolean)
POAccountID	<input type="text" value="12"/>	(int)
DocumentReferenceID	<input type="text" value="23"/>	(int)
Multiple: POItem <i>minOccurs: 0</i> <i>maxOccurs: unbounded</i>		
[ADD NEW POItem]		
Units	<input type="text" value="12"/>	(float)
ProductID	<input type="text" value="23"/>	(int)
<input type="button" value="CLEAR MESSAGE"/> <input type="button" value="SAVE MESSAGE"/>		

Slika 5. Generisana složena forma

6. VALIDACIJA FORMI

Funkcionalnost web servisa je u velikoj meri određena sprovođenjem strategije čvrstog definisanja atributa pozivnih poruka. Naime, validacija njihovih elemenata utiče na smanjenje rizika od nepravilnog sprovođenja procesa – zaustavljanja procesa usled pojave grešaka i aktiviranja kompenzacionih mehanizama.

U okolnostima sprovođenja manuelnih aktivnosti poslovnih procesa, ovaj rizik je izuzetno veliki. Zato, neophodno je da generički WSDL klijent obuhvati i automatsku implementaciju validacionih mehanizama, i to na osnovu atributa XSD šeme.

Principi na osnovu kojih se vrši validacija elemenata formi su:

- Vrednosti prostih tipova predstavljenih strukturom `<xs:element>` čiji je atribut *nillable* podešen na vrednost *false* moraju biti definisane;
- Nad vrednostima prostih tipova (`<xs:simpleType>`), sa `<xs:restriction>` podelementom strukture koji sadrži `<xs:maxLength>` podelement se vrši validacija broja unetih karaktera i proverava da li je on veći od zadate vrednosti;
- Nad vrednostima prostih tipova (`<xs:simpleType>`), sa `<xs:restriction>` podelementom strukture koji sadrži `<xs:pattern>` podelement se vrši validacija na osnovu regularnog izrara (*regular expression*) koji predstavlja *value* atribut ovog podelementa;

7. ZAKLJUČAK

Iako su čvrsti standardi iz oblasti web servisa i njihove orkestracije (BPEL) definisani relativno davno, tržište SOA aplikacija je još uvek u stvaranju. Osnovni razlog za to je nedovoljna informisanost potencijalnih korisnika o mogućnostima organizacije poslovnog okruženja i velikom potencijalu mogućih ušteda kroz povećanje efikasnosti rada, sistematizovanu komunikaciju i izveštavanje i mikro-menadžment preduzeća kroz neposrednu kontrolu i praćenje njegovog poslovanja.

Pored ovoga, problem slabo dostupnog tržišta korisnika proističe i iz činjenice da su dostupni alati uobičajeno relativno teški za korišćenje, što utiče na to da su potrebna velika ulaganja u obuku korisnika, ali i određeni nivo reorganizacije preduzeća da bi se stekli uslovi za ostvarivanje navedenih ušteda. Iz tog, ali i drugih razloga, SOA oblast je u ovom trenutku skoro nedostupna korisnicima iz malih i srednjih preduzeća.

Generički WSDL klijent predstavlja jednu od osnova za izgradnju razvojne infrastrukture servisno orijentisane arhitekture, namenjene upravo ovom sektoru.

Orijentisanost ka tehnologijama i aplikacijama otvorenog koda, dekompozicija procesa razvoja servisno orijentisane arhitekture i njihova apstrakcija do opšte razumljivog nivoa, su osnovni principi funkcionalne tehnološke infrastrukture koja treba da učini realnim implementaciju SOA u sektoru malih i srednjih preduzeća.

Uloga generičkog WSDL klijenta je da, implementacijom novog sloja servisno orijentisane arhitekture koji treba da doda vrednost atributima upotrebljivosti web servisa, ali i ostvari veću upravljivost procesa razvoja SOA u budućnosti, podrži ostvarenje navedenih principa.

8. LITERATURA

[1] *"Towards a Normative SOA Model"*, Gregory Kohring, C&C Research Laboratories, NEC Europe Ltd., OASIS SOA-RM, March 2005

[2] *"SOA Enterprise Patterns—Services, Orchestration and Beyond"*, Dragos Manolescu and Boris Lublinsky, to be published by Morgan-Kaufmann Publishers, 2007

[3] *„Enterprise SOA: Service-Oriented Architecture Best Practices“*, Dirk Krafzig, Karl Banke, Dirk Slama, Prentice Hall PTR, 2004

[4] *„Service-oriented architecture and orchestration patterns“*, Dragos Manolescu, Boris Lublinsky, Draft, 2004-2006

[5] *„OASIS SOA Reference Model TC“*, OASIS (Organization for the Advancement of Structured Information Standards) konzorcijum, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm

[6] *„Web Services Description Language (WSDL) 1.1“*, W3C (The World Wide Web Consortium), <http://www.w3.org/TR/wsdl>

[7] *“Designing a WSDL client, Discovering Web Services”*, Billal Siddiqui, <http://www.devx.com/javaSR/articles/siddiqui2/siddiqui2-1.asp>