UNIVERSITATEA TEHNICĂ CLUJ-NAPOCA

Facultatea de Automatica si Calculatoare
Catedra de calculatoare

Planul National de Cercetare, Dezvoltare si Inovare - PN II

Program Idei, 2007

Contract CNCSIS Nr. 333

Maestro: Compunerea automată a serviciilor Web folosind ontologii - Sintază etapă 2009

Cuprins

Director proiect: **prof. dr. Ing. Ioan Salomie**

Septembrie 2009

1. Introducere

Serviciile Web furnizează o modalitate standard de a asigura interoperabilitatea între diferite aplicații software care rulează pe platform diferite. Indiferent de aplicația în sine, serviciile Web sunt folosite pentru a integra în mod flexibil sisteme slab cuplate care pot fi descompuse şi recompose pentru a reflecta natura dinamica a business-ului. Serviciile Web tind astfel să transforme Web-ul dintr-o colecție statică de documente într-o biblioteca vastă de programe [1]. Acesta este motivul pentru care cercetătorii din domeniul industriei cât şi cei din domeniul academic manifestă un interes considerabil în ceea ce priveste serviciile Web. Tehnologiile standard existente la ora actuală pentru servicii Web (*WSDL*, *UDDI*) furnizează descrieri la nivel sintactice ale acestora, neexistând o descriere formală a semanticii lor. Lipsa semanticii care poate fi interpretată de calculatoare duce la necesitatea intervenției umane în procesele de descoperire şi compunere a serviciilor Web, îngreunând utilizarea serviciilor Web în procese de business complexe în care automatizarea este obligatorie. Descrierile semantice ale serviciilor Web sunt necesare pentru a putea compune în mod automat servicii.

Proiectul Maestro are ca obiectiv principal "*studiul şi elaborarea unei teorii şi a unui cadru unitar de compunere automată a serviciilor Web folosind ontologii*". În această etapă a proiectului am urmarit să: (*i*) elaborăm algoritmi de selecție a soluție optime de compunere, (*ii*) elaborăm metrici de evaluare calitative/cantitative, (*iii*) implementăm, testăm, studiem eficiența algoritmilor propuşi, să efectuăm acțiuni corectoare. Criteriile care au stat la baza selecției soluției optime de compunere au fost: preferințele utilizatorului cu privire la atributele de QoS precum şi calitatea gradului de potrivire semantic între serviciile Web implicate într-o solutie de compunere. O descriere succintă a obiectivelor propuse, precum şi a performanțelor obținute în cadrul acestei etape a proiectului va fi făcuta în cele ce urmează.

2. Proiectarea şi elaborarea algoritmilor de selecție

În aceasta etapă a proiectului am urmărit dezvoltarea unui framework ce oferă suport pentru întreg ciclul de compunere automată a serviciilor Web: (*i*) *publicarea serviciilor Web* într-un registru îmbogățit semantic, (*ii*) *descoperirea automată a serviciilor Web semantice, (iii) compunerea automată a serviciilor Web*, şi *(iv) selectarea soluției optime de compunere*. O particularitate a abordării propuse o constituie tehnicile utilizate pentru compunerea automată a serviciilor Web şi selectarea soluției optime de compunere. Pentru a compune automat servicii Web am adaptat tehnici inspirate din domeniul planificarii din inteligența artificială (IA), iar pentru selecția solutiei optime de compunere am dezvoltat un algoritm de selecție care se bazează pe principiile de funcționare ale sistemului imunitare.

2.1.1 Modelul de graf de planificare îmbogătit

Din punct de vedere matematic, un graf de planificare este un graf orientat organizat pe nivele în care arcele sunt orientat de la un nivel la nivelul imediat următor. Nodurile nivelului zero corespund unui set de literali L_0 reprezentând starea inițiala a problemei de planificare. Nivelul 1 constă dintr-un set de acțiuni, A_1, şi un set de literari

L_1. A_1 este setul de acțiuni pentru care precondițiile sunt noduri din L_0, şi L_1 este reuniunea dintre L_0 şi setul de efecte pozitive ale acțiunilor din A_1. Un nod de acțiuni din A_1 este conectat prin arce de intrare la precondiții din L_0, şi prin arce de ieşire la efecte pozitive sau negative din L_1. Astfel, graful de planficare se extinde iterativ cu fiecare nivel. Procesul de expansiune continuă până când graful ajunge la un nivel în care setul de literali conține toți literalii obiectiv sau se atinge un punct fix în expansiunea grafului. Spunem ca am ajuns într-un punct fix în expansiunea grafului atunci când setul de acțiuni şi setul de literali de pe două nivele consecutive sunt identici.

Pentru a adresa problema compunerii serviciilor Web am definit o structură de graf de planificare îmbogățit. Graful de planificare îmbogățit extinde graful de planificare clasic cu conceptul de *cluster de servicii* şi *legătură de similaritate semantică*. Înainte de a prezenta modelul de graf de planificare îmbogățit, menționăm că în continuare prin operația unui serviciu, vom înțelege un concept ontologic care adnotează operația unui serviciu, iar prin parametrii de intrare/ieşire vom înțelege concepte ontologice care adnotează set-ul de parametrii de intrare/ieşire ai operației unui serviciu.

Problema compunerii serviciilor Web a fost mapată la graful de planificare îmbogățit astfel:

- O operația a unui serviciu Web reprezintă o acțiune în graful de planificare. Setul de parametrii de intrare corespunzători operației serviciului se mapează pe precondiții acțiunii, iar parametrii de ieşiri pe efecte acțiunii.
- Cererea de compunere poate fi văzută ca şi un serviciu virtual care nu are intrări, doar ieşiri ce reprezintă parametrii de intrare pentru serviciul compus dați de utilizator. Aceşti parametrii sunt mapați ca şi set-ul de literali L_0.
- În fiecare nivel $n \geq 0$, A_i este reprezentat printr-un set de *clustere de servicii*, parametrii de intrare ai acestor servicii sunt noduri în L_{i-1}. Un *cluster de servicii* grupează servicii care au aceeaşi funcționalitate, precum şi acelaşi set de parametrii de intrare si care sunt adnotate cu concepte care sunt in relatie *isA*. L_i este reuniunea dintre L_{i-1} şi setul de ieşiri ale serviciilor din A_i. Parametrii din L_i sunt de asemenea grupați, dar în *clusteri de literali*. Un cluster de literali grupează literali care sunt în relație *isA*, şi care aparțin serviciilor din A_i care fac parte din acelaşi cluster de servicii.
- Obiectivele compunerii sunt mapate pe un set de literali cu rol de obiective, L_g. Setul de literali L_g reprezintă parametrii de ieşire pentru serviciul compus dați de către utilizator.

Graful de planificare este extins până când toate obiective compunerii sunt conținute în setul de literali L_n de pe ultimul nivel, sau până cand se ajunge la un punct fix.

Construirea grafului de planificare se face iterativ. La fiecare iterație un nou nivel se adaugă grafului. Serviciile implicate în compunere sunt primite de la modulul de descoperire care găseşte serviciile corespunzătoare pe baza gradului de potrivire semantică între intrările serviciilor şi setul de literali corespunzător nivelului anterior. Figura 1 ilustrează un exemplu de graf de planificare îmbogățit. Graful prezentat are 3 nivele cu câte două *clustere de servicii* ($s_1 \ldots s_8$) pe fiecare nivel. Clusterii de literali sunt reprezentate prin acolade. Spre exemplu, clusterul de literali {*F, FF, FFF*} este legat la al doilea cluster de servicii de pe primul nivel, deoarece serviciile din acest cluster s_1, s_2 şi s_3, au *F*, *FF* şi, *FFF* ca şi ieşiri. O soluție de compunere pentru graful de planificare

îmbogățit din Figura 1 constă dintr-un sub-set de servicii: {[s_1 s_2], [s_6, s_7]}. În fiecare subset al soluției doar un sigur serviciu din fiecare cluster poate fi selectat.

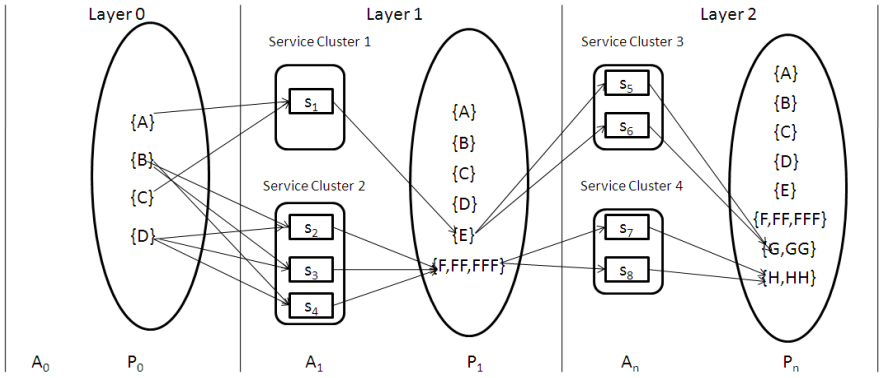

Figure 1: Exemplu de graf de planificare îmbogățit

2.1.2. Matrice de legături semantice

Matricea de legături semnatice (MSL) păstrează legături de similiaritate semantică stabilite între servicii implicate în compunere (serviciile din graful de planificare îmbogățit). Spunem că între două servicii se stabileşte o legătură de similaritate semantică dacă există un grad de potrivire semantic (*DoM*) >0 între setul de parametrii de ieşire ai unui serviciu şi set-ul de parametrii de intrare ai celuilalt. Un serviciu Web poate fi legat de unul sau mai multe servicii prin astfel de legături de similaritate semantică. Matricea de legături semnatice (MSL) se contruieşte iterativ în procesul de compunere odată cu expansiune grafului de planificare îmbogățit. Liniile şi coloanele matricii sunt etichetate cu servicii din graful de planificare. O coloană şi o linie cu acelaşi index va fi etichetată cu acelaşi serviciu. Un element al matricii de pe linia *i*, etichetată cu *service_i*, şi coloana *j*, etichetată cu *service_j*, din matrice este definit formal astfel:$msl_{ij} = (Out\_In, score)$, unde:

- $Out\_In = \{(out_i, in_j) | DoM(out_i, in_j) > 0\}$. *DoM* evaluează gradul de potrivire semantică între conceptul out_i care adnotează parametrul de ieşire al unui serviciu s_i, şi conceptul in_j care adnotează parametrul de intrare al serviciului s_j.
- *score* reprezintă o valuare asociată serviciului s_i de pe linia *i* şi este calculată pe baza unui scor inițial ($score_{init}$) şi a unui scor intermediar ($score_{inter}$) folosind versiuni adaptate ale metricilor din information retrieval (IR) cum ar fi *recall*, *precision*, *F-Measure*.

2.1.3. Selecția soluției optime de compunere

Următorul pas după construierea *EPG* şi *MSL* este determinarea soluție de compunere a serviciilor Web. Această soluție se obține pe baza *EPG* şi *MSL*. Prin urmare structura unei soluții de compunere va fi similară structurii grafului. Astfel va conține o secvență de nivele, fiecare nivel având câte un set de clustere, fiecare conținând câte un singur

serviciul Web selectat din cluster-ul echivalent. Deoarece poate să existe mai mult de o singură soluție compusă, intenția noastră este aceea de a determina soluția care se potriveşte cel mai bine cerințelor utilizatorului. În acest sens propunem ordonarea soluțiilor obținute după valoare unei funcții dependente de valorile QoS si de DoM stabilit între serviciile implicate într-o soluție de compunere. Pentru a atinge acest obiectiv se va folosi o abordare inspirată din principile de funcționare ale sistemului imunitar.

2.1.3.1. Maparea principiului de selecție a clonelor pe probleme selecției serviciilor Web

În compunerea serviciilor Web, selecția clonelor se mapează după cum urmează: (i) o celulă B (sau anticorp) este reprezentată de o soluție de compunere a serviciilor Web; (ii) un antigen este reprezentat printr-o funcție *f* care evaluează setul de soluții compuse în vederea determinării soluției optime în termeni de atribute *QoS*; (iii) afinitatea dintre un anticorp şi un antigen este reprezentată de valoarea funcției *f* pentru o soluție compusă. În abordarea de față, antigenul este reprezentat print-o funcție *QF* care evaluează valoarea *QoS* şi scorul de similaritate semantică a unei soluții de compunerere *sol*. Această funcție este:

$$QF\ (sol) = \frac{w_{QoS} * \mathrm{QoS}(sol) + w_{Sem} * \mathrm{Sem}(sol)}{w_{QoS} + w_{Sem}}$$

unde,

- $Qos(sol) = \sum \mathrm{QoS}(s) = \sum \frac{\sum w_i * qos_i}{\sum w_i}$
- $Sem(sol) = \sum \mathrm{SimS}(s_{ijk}.o, s_{pqr}.i)$
- w_{Qos}, w_{sem} sunt ponderile asociate *QoS*-ului şi scorului de similaritate semantică
- w_i sunt ponderile asociate qos_i

Obiectivul aplicării principiului de selecție a clonelor este acela de a obține soluția optimă sol_{opt} care maximizează funcția *QF*. Astfel, afinitatea între un anticorp şi antigen este considerată ca fiind valoare funcției *QF* pentru o soluție *sol*. Se va folosi un alfabet binar pentru codificarea unui anticorp (a unei soluții *sol*). Codificare păstrează structura unei soluții şi conține o secvență de nivele, fiecare nivel având un set de clusteri. Din fiecare cluster al grafului *EPG* este selectat un singur serviciu Web. Un anticorp va fi reprezentat ca un set de forma $\{[s_{01},...,s_{0n_i}],\ldots,[[s_{m1},...,s_{mn_i}]]\}$, unde s_{ij} este un serviciu Web din clusterul *j* al nivelului *i* de servicii şi *m* este numărul de nivele din *EPG*. Fiecare serviciu Web este codificat ca şi un sir de numere binare reprezentând numărul de identificare al serviciului în cadrul clusterului (Figura 2)

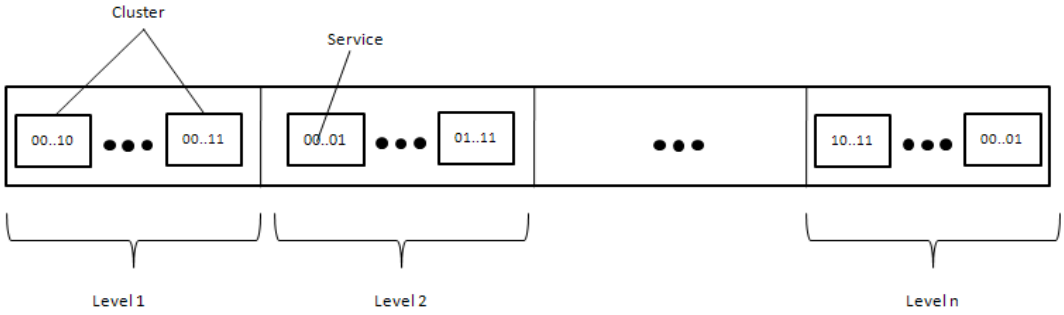

Figure 2: Reprezentarea unei solutii de compunere ca un sir de numere binare

2.1.3.2 Algoritmul de selecție

Algoritmul de selecție propus determină soluția optimă sol_{opt} pe baza funcției *QF*. Algoritmul va fi aplicat pentru tripletul: (*EPG, MSL, QF*). În vederea realizării acestui lucru, următoarele funcții trebuie stabilite: (i) o funcție pentru determinarea numărului n_c de clone care vor fi generate pentru fiecare soluție; (ii) funcție care va calcula rata de mutație pe baza valori afinității. Pentru calcularea numărului de clone care vor fi generate pentru fiecare soluție se va folosi formula (1), unde β este un factor de multiplicitate şi *N* este numărul total de soluții.

$$n_c = \text{round}(\beta \cdot N) \quad (1)$$

Rata de mutație va fi definită ca şi în formula (2), unde α este un factor de multiplictitate, *sol.length* se referă la numărul de servicii Web conținute în soluția *sol*, *N* este numărul de soluții, iar *i* este indexul corespunzător fiecărei soluții (i=1 corepunde soluției cu cea mai mare afinitate).

$$m_r(sol) = \text{round}\left(\frac{\alpha \cdot sol.length}{N - i + 1}\right) \quad (2)$$

Algoritmul iterează peste un set *B* de soluții de compunere servicii, până când se ajunge la o stare statisfăcătoare. Inițial, setul *B* conține o singură pereche (*sol, QF(sol)*), unde *sol* este o soluție validă generată aleator pe baza *EPG* şi *SLM*. La fiecare iterație, pentru fiecare soluție sol_i aparținând setului *B*, se realizează următoarele acțiuni: (i) se evaluează afinitatea soluției sol_i față de funcția *QF*; (ii) se creează un set C_{soli} ce conține n_c clone ale soluției sol_i; (iii) setul C_{soli} este supus procesului de maturizare a afinității (affinitz maturation) şi evaluat conform funcției *QF*. După procesarea tuturor soluțiilor din *B*, setul *B* va fi populat cu clonele supuse procesului de maturizare a afinității asociate fiecărei soluții. Apoi se generează o nouă soluție aleatoare şi se adaugă la setul *B*. Algoritmul va returna întreg setul de soluții valide ordonate descrescător după afinitatea față de funcția *QF*, prima soluție fiind soluția optimă, sol_{opt}. Procesul de maturizarea a afinității implică o serire de etape de mutație prin care se: (i) identifică serviciile Web ale soluției şi care minimizează afinitatea acesteia; (ii) înlocuiesc serviciile Web identificate cu alte servicii din același cluster.

Un serviciu Web *s* minimizează afinitatea soluției din care face parte dacă afinitatea serviciului *s* calclulată ca şi media ponderată dintre *QoS(s)* şi *SimS(s)*, folosind ponderile w_{Qos} şi w_{sem} este mică în comparație cu afinitatea celorlator servicii din soluție.

Maturizarea afinității est einvers proporțională cu afinitatea soluției, ceea ce înseamnă că numărul de servicii Web ce trebuie înlocuite este mai mic dacă afinitatea soluției este mai mare. Prin maturizarea afinității se obțin noi soluții care lărgesc spațiul de căutare al problemei. Din cauza faptului că înlocuirea se face în mod aleator, est eesențial să se verifice dacă noile soluții sunt valide, adică dacă soluțiile conținute pot fi compuse, înainte ca acestea să fie integrate în setul *B*. Pentru a verifica dacă două servicii Web pot fi compuse se va folosi matricea *MSL*.

3. Implementare prototip experimental

Pentru a demonstra abordarea propusă a fost implementat un prototip experimental. Arhitectura prototipului este prezentată mai jos:

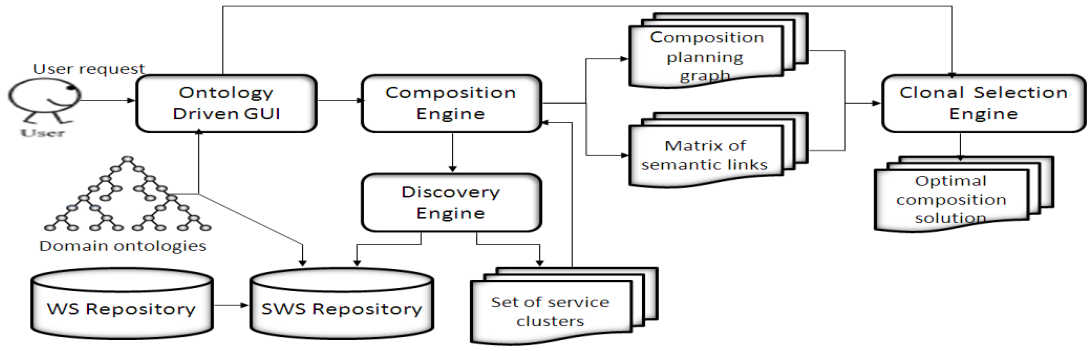

Figure 3: Arhitectura prototipului experimental

Atât furnizorul de servicii cât şi solicitantul de servicii interactionează cu prototipul experimental prin intermediul unei interfețe utilizator ghidată ontologic. Prin furnizarea unui limbaj controlat care utilizează concepte din ontologie, interfața ghidează utilizatorul în procesul de căutare şi compunere a serviciilor Web. Prin intermediul interfeței, utilizatorului poate să specifice: (i) criteriile pe baza cărora să se facă căutare unui serviciu atomic, (ii) o cererea de serviciu compus. Cererea de serviciu compus poate fi exprimată în termeni de cerințe functionale/non-functionale.

SWS Repository păstrează servicii adnotate semantic. Serviciile sunt adnotate cu concepte dintr-o ontologie de domeniu pe baza standardului *SAWSDL* [11].

Ontologia de domeniu a fost dezvoltată manual în editorul de ontologii Protégé. Pentru a putea interoga în mod eficient ontologia am folosit scheme de etichetare bazate pe intervale care asignează fiecărul concept ontologic un interval de forma [*start*, *end*].

Discovery Engine poate primi o cerere de la un solicitant de serviciu sau de la *Composition Engine*. În cazul în care cererea provine de la un solicitant de servicii *Discovery Engine* returnează un set ordonat de servicii care se potrivesc cu cererea. Dacă cererea provine de la *Composition Engine*, returnează un set de servicii atomice organizate în clusteri.

Composition Engine este componenta care este responsabilă cu construirea grafului de planificare îmbogățit şi a matricii de legături semantice. Pe parcursul construcției grafului de planificare *Composition Engine* interacționează cu *Discovery Engine*. Pe baza

clusterilor de servicii primiți de la modulul de descoperire, composer-ul construieşte graful de planificare împreună cu matricea de legături semantice.

Cele două structuri de date (graful de planificare îmbogățit şi matricea de legături semantice) sunt folosite de către *Clonal selection engine* în procesul de selecție. Rezultatul procesului de selecție este un set ordonat de soluții de compunere care satisfac preferințele utilizatorului. Prima soluție din listă corespunde soluției optime de compunere.

Soluția de compunere selectată este executată de către *Execution Engine*.

4. Studii de caz, rezultate experimentale

Prototipul a fost testat pe două studii de caz: trip planning and concert reservation. Rezultatele obținute au fost statisfăcătoare. În această secțiune sunt prezentate rezultatele experimentale obținute prin aplicarea metodei propuse pentru compunerea serviciilor Web din domeniul trip planning. Se ia în considerare următorul studiu de caz: *utilizatorul este interesant în planificarea unei vancanțe (rezervare bilete de avion, rezervare hotel*). Resursele dezvoltate pentru a putea aplica metodă propusă au fost: (i)o ontologie pentru domeniul planificării unei călătorii; (ii) un set de servicii Webdin domeniul trip planning. Figurile 3, 4, 5 ilustrează un fragment din graful de planificare îmbogățit, un fragment din matricea de legături semantice precum, şi o parte din soluțiile de compunere obținute ordonate după valoare lui *QF* pentru scenariul trip planning.

Level	Service Set			
Level 0	Cluster 1	**UserRequest** *Inputs:* *Outputs:* country destinationcity sourcecity startdate numberofdays numberofperson numberofrooms hotel singleroom		
Level 1	Cluster 1	**SearchHotel** *Inputs:* DestinationCity Hotel *Outputs:* HotelName	**SearchExoticHotel** *Inputs:* EuropeanExoticDestinationCity Accommodation *Outputs:* MediterraneanHotelName	**SearchEuropeanExoticHotel** *Inputs:* EuropeanDestinationCity Hotel *Outputs:* LuxuryMediterraneanHotelName
	Cluster 2	**CheckFlight** *Inputs:* Country SourceCity DestinationCity StartDate *Outputs:* FlightIdentifier	**LookupFlight** *Inputs:* Country SourceCity DestinationCity StartDate *Outputs:* Identifier	
Level 2	Cluster 1	**BookHotel** *Inputs:* HotelName SingleRoom StartDate NumberOfDays NumberOfRooms *Outputs:* AccommodationConfirmation	**BookExoticHotel** *Inputs:* MediterraneanHotelName DeluxSingleRoom Date NumberOfDays NumberOfRooms *Outputs:* ReservationConfirmation	**BookEuropeanExoticHotel** *Inputs:* LuxuryMediterraneanHotelName SingleRoom StartDate NumberOfDays NumberOfRooms *Outputs:* AccommodationConfirmation
	Cluster 2	**FlightReservation** *Inputs:* FlightIdentifier NumberOfPerson StartDate *Outputs:* FlightPrice FlightConfirmation	**FlightReservationOnline** *Inputs:* Identifier NumberOfPerson Date *Outputs:* FlightPrice Confirmation	

Figure 3: Un fragment din EPG

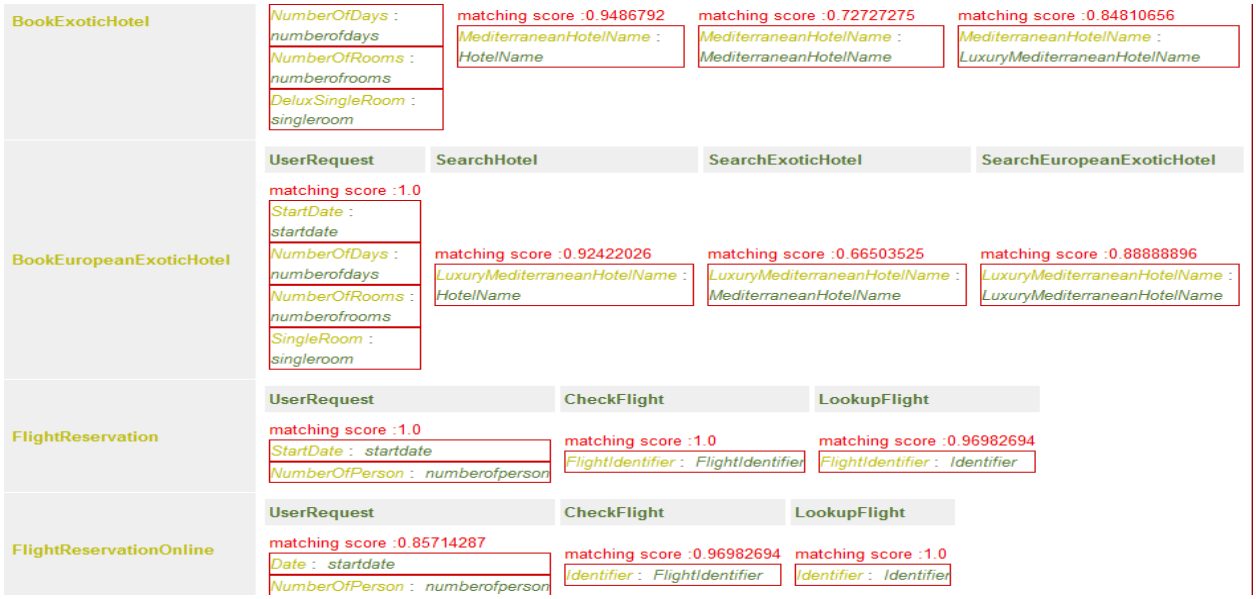

Figure 4: Un fragment din MSL

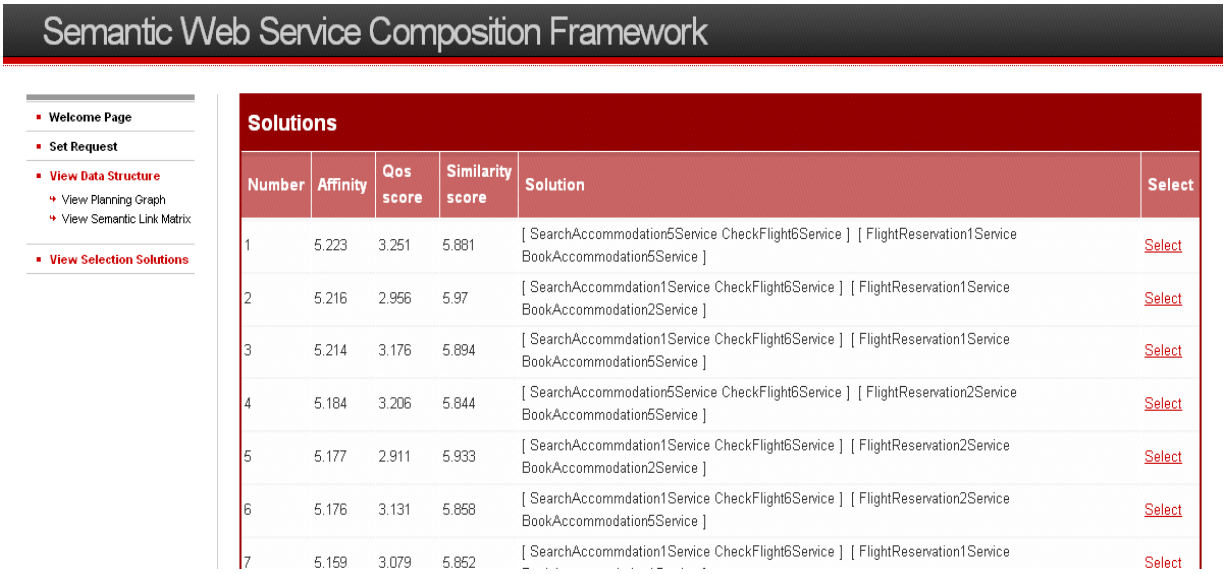

Figure 4: O parte din soluțiile obținute pentru trip planning

Referințe

1. Tanasescu, V., Domingue, J.: Toward User Oriented Semantic Geographical Information Systems. In: 2nd AKT Doctoral Symposium, Aberdeen Univeristy, UK (2006)
2. Zheng, X., Yan, Y.: An Efficient Syntactic Web Service Composition Algorithm Based on the Planning Graph Model. In: Proceedings of the IEEE International Conference on Web Services, pp. 691-699 (2008)
3. Gao, Y., et al.: Immune Algorithm for Selecting Optimum Services in Web Service Composition. In: Wuhan University Journal of Natural Sciences, Wuhan University Journals Press, Volume 11, Number 1, pp. 221-225 (2006)
4. Xu, J., Reiff-Marganiec, S.: Towards Heuristic Web Services Composition Using Immune Algorithm. In: Proceedings of the 2008 IEEE International Conference on Web Services, Beijing, China, pp. 238-245 (2008)