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 automata a serviciilor Web folosind ontologii

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

Decembrie 2007

Cuprins

Introducere

Serviciile Web au aparut ca o nevoie naturala de extindere a tehnologiei informatiei. Ele au la baza experienta acumulata de la inceputul erei informationale, experienta ce s-a concretizat, in diverse etape, prin tehnologii si metodologii noi. La ora actuala tot mai multe companii ce isi desfasoara activitatea cu ajutorul aplicatiilor Internet isi indreapta atentia catre aceasta ramura a tehnologiei. *Serviciile Web* fac posibila realizarea interoperabilitatii Business-to-Business (B2B) prin interconectarea serviciilor oferite de catre multipli parteneri de business pe baza unor procese de business. Aceasta interconectare a *serviciilor Web* pentru a satisface un process de business oarecare este numita *compunere de servicii Web*. Compunerea poate fi vazuta ca o agregare de *servicii Web* elementare sau compuse. Compunerea serviciilor Web permite definirea de aplicatii care cresc in complexitate prin agregarea in mod progresiv de noi componente la un nivel inalt de abstractizare. Odata cu cresterea rapida a numarului de servicii Web disponibile, a devenit practic imposibil pentru un utilizator uman sa analizeze toate aceste servicii si sa genereze manual un plan de compunere. Aparea astfel **nevoia de automatizare a procesului de compunere** a serviciilor Web. Automatizarea completa a procesului de compunere a *serviciilor Web* implica automatizarea urmatorilor pasi: descoperirea automata a serviciilor Web, compunerea automata a serviciilor Web si executia automata a serviciului Web rezultat in urma compunerii. Tehnologiile existente la ora actuala (*WSDL, UDDI*), furnizeaza doar descrieri la nivel sintactic ale serviciilor Web. Lipsa unor semantici intrepretabile de masina necesita intreventia umana pentru descoperirea si compunerea serviciilor, ceea ce impiedica folosirea serviciilor Web in contexte de business complexe in care e necesara automatizarea acestor procese. **Serviciile Web Semantice** relaxeaza aceasta restrictie prin adnotarea serviciilor cu descrieri formale bogate ale capabilitatilor lor, astfel incat acestea sa poata fi utilizate de catre alte aplicatii sau servicii fara asistenta umana.

2. Abordari existente in compunerea automata a serviciilor Web

In acesta sectiune vom face o scurta prezentare a abordarilor existente privind compunere a serviciilor Web.

2.1. Compunerea serviciilor Web folosind tehnica workflow-urilor. In compunerea serviciilor Web folosind tehnica workflow-urilor se considera ca un serviciu compus este similar cu un workflow in care se specifica fluxul componentelor de lucru.

Eflow [5] este o platforma pentru specificarea, imbogatirea si managemetul serviciilor compuse. Eflow foloseste o metoda statica de generare a workflow-ului. Un serviciu compus este modelat prin intermediul unui graf care defineste ordinea de executie a nodurilor in proces. Graf-ul este creat manual, dar poate fi actualizat in mod dinamic.

Argos [2] este o arhitectura pentru compunerea serviciilor Web bazata pe descrieri semantice ale serviciilor Web. Autorii definesc o ontologie de domeniu care furnizeaza semanticile formale necesare pentru e descriei operatiile si sursele disponibile. Sistemul poate genera in mod automat workflow-uri computationale care raspund interogarilor utilizatorului. Workflow-ul astfel generat este translatat in BPEL4WS pentru executie. Aceasta va reduce considerabil cantitatea de timp necesara pentru a genera un workflow.

Deoarece utilizatorii nu au control asupra procesului de executie al serviciului compus, pot exista executii nedorite de servicii compuse care nu intrunesc asteptarile acestora.

2.2. Compunerea serviciilor Web bazata pe tehnici din domeniul AI planning. La ora actuala exista foarte multe abordari in domeniul compunerii automate a serviciilor Web care trateaza compunerea de servicii ca si o problema de AI planning. In cele ce urmeaza vom face o scurta prezentare a abordarilor privind compunerea automata a serviciilor Web bazate pe tehnici din AI planning. Vom clasifica aceste tehnici de compunere in cinci categorii si anume: *calculul situatiilor*, *Planning Domain Definition Language (PDDL)*, *planificare bazata pe reguli*, *demonstratori de teoreme si altele*. Majoritatea metodelor descrise aici folosesc OWL-S ca si limbaj de descriere a serviciilor Web. Exista si cateva abordari care folosesc WSDL sau propriile lor limbaje.

2.2.1. Calculul situatiilor. McIlarith and **Son** [13] propun o abordare pentru construirea unei tehnologii bazate pe agenti pe baza notiunii de proceduri generice si constrangeri furnizate de utilizator. Ei argumenteaza ca o versiune imbogatita a limbajul de programare Golog furnizeaza un formalism natural pentru programarea *serviciilor Web*. Aceste contributii sunt realizate prin dezvoltarea unui interpretor ConGolog care comunica cu *serviciile Web* printr-un Open Agent Architecture (OAA). Principalul dezavantaj este ca servicile si procedurile sunt scrise in logica de ordinul intai.

2.2.2. PDDL este recunoscut pe scara larga ca si intrare standardizata pentru planificatori clasici. Mai mult, deoarece DAML-S a fost influentat puternic de limbajul PDDL, maparea dintr-o reprezentare in alta este usoara (atata timp cate este considerata doar informatie declarativa). Cand este necesara o planificare pentru o descriere de servicii, descrierile DAML-S pot fi translatate in format PDDL. Apoi diferiti planificatori pot fi exploatati pentru sinteza serviciilor. In [14] McDermott prezinta o metoda de compunerea a serviciilor Web bazata pe PDDL.

2.2.3. Planificarea bazata pe reguli. Medjahed [15] prezinta o tehnica pentru a genera un serviciu Web compus din descrieri declarative de nivel inalt. Metoda foloseste reguli de compozabilitate pentru a determina daca doua servicii sunt compozabile. Abordarea de compunere consta din patru faze. Prima, faza de specificare este faza in care este specificata compunerea dorita folosind un limbaj de descriere de nivel inalt numit Composite Service Specification Language (CSSL). A doua, faza de matchmaking foloseste reguli de composabilitate pentru a genera planuri de compunere care se conformeaza specificatiilor furnizate de solicitantul de service. Faza a treia este faza de selectie. In cazul in care sunt generate mai multe planuri, in faza de selectie, solicitantul de servicii selecteaza un plan pe baza parametrilor de QoS ai compunerii (ex:cost, ordonare, etc.). Faza finala este faza de generare in care este generata si prezentata o descriere detaliata a serviciului compus, solicitantului de serviciu. Principala contributie a acestei abordari o reprezinta regulile de compunere, deoarece acestea definesc atribute ale serviciilor Web care pot fi folosite in compunere. In final putem sublinea faptul ca autorii introduc notiunea de **sablon de compunere** care defineste dependente intre tipuri diferite de servicii.

SWORD [18] este un alt toolkit pentru constructia de service Web compuse care foloseste tehnici de planificare bazate pe reguli. SWORD nu este dezvoltat pe standarde

de descriere a serviciilor Web existente cum ar fi WSDL, DAML-S. In schimb foloseste modelul Entitate-Relatie (ER) pentru a specifica servicii Web. In SWORD un serviciu este modelat prin preconditii si postconditii. Ele sunt specificate intr-un model al lumii care consta din entitati si relatii intre entitati. Un serviciu Web este reprezentat ca si o regula Horn care denota postconditia care este realizata in cazul in care preconditia este adevarata. Pentru a crea un serviciu compus, e necesar ca solicitantul de serviciu sa specifice starea initiala si finala a serviciului compus. Pe baza acestor specificatii, folosind un sistem expert bazat pe reguli, se va genera un plan.

2.2.4. Alte tehnici din domeniul planificarii AI. In [30] planificatorul SHOP2 este aplicat pentru compunerea automata a serviciilor Web, care sunt furnizate cu descrieri DAML-S. SHOP2 este un planificator bazat pe Hierarchical Task Network(HTN). Autorii cred ca conceptul de descompunere a task-ului din planificare HTN este foarte similar cu conceptul de descompunere a proceselor compuse intr-o ontologie de procese DAML-S. Un alt argument in favoarea planificarii HTN adus de catre autori este acela legat de faptul ca planificarea HTN este mult mai eficienta decat alte limbaje de planificare, cum ar fi Golog. Autorii dau o descriere foarte detaliata a procesului de translatare din DAML-S in SHOP2.

Sirin et la [21] prezinta o metoda semiautomata pentru compunerea servicilor Web. De fiecare data cand un utilizator doreste sa selecteze un serviciu, sunt prezentate acestuia toate serviciile posibile care se potrivesc cu serviciul selectat. Alegerea unui serviciu posibil se bazeaza atat pe atribute functionale cat si non-functionale. Functionalitatile (parametrii) sunt reprezentate prin clase OWL, iar pentru a gasi potriviri intre servicii se foloseste un rationator OWL. O potrivire este definita intre doua servici daca un parametru de iesire al unui serviciu este aceeasi clasa OWL sau subclasa a unui parametru de intrare al altui serviciu. Motorul de inferenta OWL poate ordona potriviri intre servicii astfel: prioritatea potrivirii este mai scazuta cand distanta intre doua tipuri in arborele ontologic creste. In cazul in care sunt gasite mai multe potriviri, sistemul filtreaza serviciile pe baza atributelor non - functionale care sunt specificate de catre utilizator ca si constrangeri. Ideea compunerii semi-automate este deosebit de interesanta deoarece in scenari de business reale este deosebitde greu sa se compuna automat servicii. Desi metoda propusa este simpla, ea indica modul in care un utilizator uman poate lucra impreuna cu un planificator HTN pentru a genera un serviciu compus care satisface cerintele acestuia.

2.2.5. Demonstratori de teoreme. Waldinger [30] elaboreaza o idee pentru sinteza serviciilor folosind demonstratori de teoreme. Abordarea este bazata pe deductie automata si sinteza programelor si isi are originile in cercetarile lui anterioare. Serviciile Web disponibile initial si cerintele utilizatorului sunt descrise intr-un limbaj de ordinul intai, iar apoi sunt generate demonstratii constructive cu ajutorul demonstratorul de teoreme Snark. In final sunt extrase descrierile de compuneri de servicii dintr-o demonstratie particulara.

Lämmermann [31] aplica Structural Synthesis of Program (SSP) pentru compunerea automata a serviciilor. SSP este o abordare deductiva pentru a sintetiza programe din specificatii. Specificarea serviciilor include doar proprietati structurale (ex: informatie de intrare/iesire). SSP foloseste variabile propozitionale ca si identificatori pentru parametrii de intrare/iesire si logica propozitionala intuitiva pentru rezolvarea problemei

compunerii. Autorii considera ca problema compunerii serviciilor Web este echivalenta cu problema cautarii demontratiei. Ei iau in considerare si avantajul disjunctiilor din logica clasica pentru a descrie exceptiile care pot aparea in timpul invocarii serviciilor.

Rao et. al. [33] introduce o metoda pentru compunerea automata a serviciilor Web semantice folosind demonstratori de teoreme bazati pe logica liniara. Metoda foloseste DAML-S pentru reprezentarea externa a serviciilor. Intern serviciile sunt prezentate prin axiome extralogice si demontratii in logica liniara. Logica liniara permite oamenilor sa defineasca intr-o maniera formala atributele unui serviciu Web. Demonstratorii de teoreme bazati pe logica liniara pot trata atat specificarea serviciilor cat si informatia Web semantica. Spre deosebire de alte metode care folosesc atributele non-functionale doar pentru a filtra planul generat, autorii considera atributele non-functionale direct in procesul de demonstrare al teoremei. Atat atributele functionale cat si cele non functionale sunt translatate in axiome logice, iar distinctia intre atributele functionale si cele non-functionale se face prin reguli de inferenta.

1.3.2.3. Tehnici de compunere declarative. In compunerea declarativa, serviciile compuse sunt generate dintr-o descriere declarativa de nivel inalt. Tehnicile folosesc reguli de compunere pentru a determina daca doua servicii sunt compozabile. De cele mai multe ori aceste reguli actioneaza ca si constrangeri care trebuie sa fie satifacute pentru a compune un serviciu. Regulile sunt folosite pentru a genera planuri care se conformeaza cerintelor specificate de solicitantului de serviciu. Tehnicile care se incadreaza in aceasta categorie tind de obicei sa gaseasca o solutie optima (ex: in funtie de cost, timp) pe baza modelului matematic descris. Cel mai adesea, optimalitatea poate fi realizata prin maparea regulilor la constrangeri si incercarea de a le rezolva folosind metode de cautare a operatiilor.

SELF-SERV [3] este o unealta cadru pentru furnizarea dinamica si peer-to-peer a serviciilor Web. In SELF-SERV, serviciile Web sunt compuse in mod declarativ, iar serviciile compuse rezultate se executa de o maniera descentralizata intr-un mediu dinamic. Unealta cadru utilizeaza si adapteaza diagramele de stare ca limbaj declarativ vizual. Avantajul semnificativ al SELF-SERV este modelul peer-to-peer de executie a serviciilor, in care responsabilitatea coordonarii executiei unui serviciu compus este distribuita pe mai multe componente software echivalente numite coordonatori. Totusi, acest sistem nu ofera nici o metoda pentru crearea unei compuneri de servicii in timpul rularii. De asemenea, sistemul nu ia in considerare semantica serviciilor Web atunci cand ia deciziile privind compunerea. In plus, aceasta tehnica impune unele cerinte nerealiste care trebuie implementate de catre furnizorii de servicii.

FUSION este un sistem de infrastructura software care furnizeaza elementele de infrastructura necesare pentru a suporta portal-uri de service [23]. Data fiind o cerere de serviciu furnizata de utilizator, sistemul genereaza in mod automat un plan de executie optim si corect, apoi executa acest plan si verifica rezultatul. Cel mai important avantaj al acestui sistem este abilitatea sa de a genera un plan de executie optim in mod automat din cerintele abstracte furnizate de utilizator. Acest sistem de compunere verifica daca rezultatul planului de executie intruneaste cerintele utilizatorului.

1.3.2.4. Tehnici de compunere a serviciilor Web ghidate ontologic. Aceste tehnici faciliteaza compunerea dinamica a serviciilor Web semantice. Pentru a compune automat

si semi-automat servicii Web, sunt folosite descrieri ontologice ale serviciilor Web in termeni de concepte si relatii.
Un sistem de compunere a serviciilor Web in mod dinamic, unde un serviciu este solictitat si compus nu doar pe baza descrierii sintacticii, dar si a descrierii semanticii este propus de Fujii si Suda [10]. Pentru a satisface cererea pentru suport semantic, sistemul cuprinde trei sub-sisteme: Component Service Model with Semantic (CoSMoS), Component Runtime Environment (CoRE), si Semantic Graph based Service Composition (SeGSeC). CoSMoS integreaza informatia semantica si functionala intr-un graf semantic. CoRE frunizeaza o interfata unificata pentru a descoperi si accesa componente implementate folosind tehnologii diferite si a le face sa intercoopereze cu componentele CoSMoS. SeGSeC este un mecanism de compunere bazat pe semantica. El permite utilizatorilor sa solicite un serviciu folosind o propozitie exprimata in limbaj natural si sa genereze o cale de executie. Contributia majora a acestei abordari este ca se bazeaza pe semantica. Tehnica de compunere este cumva controlata de catre utilizatorul final. Dare setul de interogari care pot fi folosite pentru a testa functionalitatea sistemului este limitat (sistemul a fost proiecatat pentru un set de scenarii limitat). In final trebuie sa amintim ca sistemul nu dispune de un mecanism care sa monitorizeze executia serviciilor compuse.

In incheiere amintim ca a fost analizata si experimentata aplicabilitatea si a altor modele (automate finite, retele Petri, algebra proceselor, algoritmi din teoria grafurilor) in contextul compunerii automate a serviciilor Web.[7][8][22][25][29]

3. Proiecte similare

In acest subcapitol vom face o analiza/comparatie a trei proiecte: METEOR-S, WSMO, OWL-S. Scopul acestor proiecte a fost de a dezvolta unelte/limbaje pentru: a) a***dnotarea semantica*** a serviciilor Web, b) ***publicarea*** serviciilor Web semantice, c) ***descoperirea*** si ***selectia*** semantica a serviciilor Web publicate semantic, si d) **orchestrarea** si **compunerea** serviciilor Web semantice.

3.1. Meteor-S. Proiectul Meteor-S (laboratorul LSDI al Universitatii Georgia)[28] are drept scop extinderea standardelor WS existente cu tehnologiile Web-ului semantic. Meteor-S s-a focalizat pe adaugarea de semantica descrierilor WSDL si UDDI. In cadrul proiectului au fost implementate unelte specifice fiecarei faza referitore la ciclul de viata al serviciilor Web semantice: a) Adnotarea semantica a serviciilor Web; b) Publicarea semantica a serviciilor Web; c) Descoperirea semantica a serviciilor Web; d) Orchestrarea si compunerea serviciilor Web semantice. Contributia de cercetare majora privind dezvoltarea serviciilor Web semantice adusa de catre acest proiect a fost propunerea de a adnota fisiere WSDL cu concepte semantice furnizate de ontologii. Cercetarile lor s-au concretizat in cele doua standarde recomandate de catre W3C pentru a adauga semantica la descrierile serviciilor Web: WSDL-S/SAWSDL. Deoarece cele mai multe dintre conceptele din SAWSDL sunt bazate pe eforturi anterioare de dezvoltare a lui WSDL-S vom prezenta pe scurt doar abordarea WSDL-S.

Pornind de la premisa ca exista deja un model semantic al serviciilor Web, WSDL-S descrie un mecanism pentru a lega acest model semantic cu descrierea functionala sintactica capturata de WSDL. Folosind atribute de exensibilitate ale WSDL, un set de

adnotari pot fi create pentru a descrie semantic intrarile, iesirile si operatia (functionalitatea) unui serviciu Web. Prin aceasta modelul semantic este pastrat in afara WSDL-ului, facind abordarea independenta de limbajul de reprezentare ontologic. Avantajul unei astfel de abordari se datoreaza faptului ca este o abordare incrementala, construita in varful unor standarde deja existente, care se bazeaza pe instrumente si expertize deja existente. In plus, utilizatorul poate dezvolta in WSDL intr-o manierea compatibila atat aspecte la nivel operational cat si semantica serviciilor Web. WSDL-S s-a focalizeaza pe adnotarea semantica a definitilor abstracte ale unui serviciu pentru a permite descoperirea dinamica, compunerea si invocarea automata a serviciilor. WSDL-S furnizeaza cinci elemente de extensibilitate ce pot fi folosite in adnotarea intrarilor, iesirilor si operatiilor unui serviciu Web:

- un atribut de extindere, numit "modelReference" pentru a specifica asocierea intre o entitate WSDL si un concept dintr-un model semantic. El poate fi adaugat la un element de tip complex, la o operatie precum si ca elemente de extindere(preconditii si efecte).
- un atribut de extindere, numit "schemaMapping", care este adaugat la elemente XSD si tipuri complexe, pentru a trata diferentele structurale dintre elementele XML, schema a unui serviciu Web si conceptele modelului semantic corespunzator.
- doua noi elemente, numite "preconditii" si "efecte", care sunt specificate ca elemente "copil" ale elemenului "operation". Preconditiile si efectele sunt folosite in primul rand in descoperirea serviciilor, si nu sunt neaparat necesare pentru a invoca un serviciu dat.
- un atribut extensie la elemenul *interface*, numit "*category*". El consta din informatie de categorizare care va putea fi folosita cand publicam un serviciu intr-un registru de servicii Web cum ar fi UDDI.

Proiectul Meteor-S defineste un set amplu de semantici [17][24] (semantici de date, semantici functionale, semantici non-functionale si de executie) si ilustreaza aplicarea lor in diferite stagii ale ciclului de viata a unui process Web. Ceea ce deosebeste Meteor-S de abordarile WSMO si OWL-S este ca acesta se focalizeaza pe constructia proceselor abstracte predefinite in locul generarii automate a proceselor. In incheiere vom prezenta doua unelte dezvoltate in cadrul proiectului Meteor-S.

Radiant este o unealta care face parte din suita de unelte METEOR-S pentru crearea de servicii si procese Web semantice. Radiant a fost dezvoltat ca si un plugin pentru Eclipse care suporta crearea si publicarea de interfete de servicii SAWSDL/WSDL-S. Permite adaugarea de adnotari semantice la descrieri de servicii existente prin intermediul unei interfete grafice.

Lumina furnizeza o interfata utilizator bazata pe Eclipse pentru descoperirea serviciilor Web. Ofera suport pentru descoperire semantica (abordarea WSDL-S), permite proiectarea semiautomata a proceselor Web (folosind Saros), furnizeaza un registru de service imbogatit semantic bazat pe UDDI.

3.2. Abordarea WSMO. Initiativa WSMO, parte a SDK Cluster, este considerata ca fiind initiativa principala in aria serviciilor Web semantice din Europa, avand drept scop standardizarea unui cadru unificat pentru servicii Web semantice care furnizeaza suport pentru modelare conceptuala si reprezentarea formala a serviciilor, precum si pentru executia automata a acestora. Spre deosebire de OWL-S, proiectul WSMO planuieste sa

creeze nu doar o specificatie, ci si o arhitectura, precum si un set vast de unelte pentru a suporta specificatiile. WSMO defineste patru componente principale: ontologii, servicii Web, scopuri si mediatori:

- Ontologii. Ontologiile furnizeaza semantici formale pentru terminologia folosita in toate celelalte componente WSMO. WSMO [19] defineste propriul sau limbaj ontologic numit Web Service Modeling Language (WSML) [4], care este bazat pe F-logic. Exista 5 variate diferite ale WSML in functie de expresivitatea si scopului limbajului. Acestea sunt: WSML-Core (intersectie intre logici de descriere si logici horn), WSML-DL (extinde WSML-Core la logici de descriere expresive), WSML-Flight (extinde WSML-Core in directia programarii logice), *WSML-RULE* (extinde WSML-Flight) si WSML-Full (unifica toate variantele WSML)
- Servicii Web. WSMO furnizeaza o descriere a functionalitatii si comportamentului serviciilor (permitand o descriere a serviciilor care sunt solicitate de catre solicitantul de servicii, furnizate de catre furnizorul de servicii, precum si a acordului intre furnizorul si solicitantul de servicii). Serviciile Web sunt definite in WSMO prin intermediul preconditiilor, postconditiilor, supozitiilor si efectelor. Diferenta intre preconditii/postconditii si supozitii/efecte este ca preconditiile si postconditiile reprezinta conditii in spatiul de informatie inainte si dupa executia serviciului, in timp ce supozitiile si efectele sunt conditii pe stare lumii.
- Scopuri. Un scop specifica obiectivele pe care un client le poate avea atunci cand consulta un serviciu Web. Descriu aspecte legate de dorintele utilizatorului in ceea ce priveste functionalitatea si comportamentul acestuia. Ontologiile sunt folosite ca terminologie semantica definita pentru specificare scopului. Scopurile modeleaza vederea utilizatorului in procesul de utilizare a serviciilor Web si sunt definite in termeni ai capabilitatilor dorite folosind preconditii, postconditii, efecte si supozitii.
- Mediatori. Conceptul de mediator in WSMO adreseaza problema manipularii heterogeneitatilor aparute intre elemente care vor interopera prin rezolvarea problemelor care pot aparea atunci cand se folosesc terminologii diferite la nivel de date, la nivel de comunicare intre servicii (nivel protocol) si la nivel de proces de business. Mediatorii conecteaza elemente WSMO intr-o maniera slab cuplata si furnizeaza facilitati de mediere pentru rezolvarea problemelor care pot aparea in procesul de conectare a diferitelor elemente definite de WSMO.
- WSMX (Web Service Modelling Execution Environment) - este un mediu de executie care permite descoperire, selectia, medierea si invocarea serviciilor Web semantice [6]. WSMX este bazat pe un model conceptual furnizat de WSMO si demonstreaza fiabilitatea abordarii WSMO in realizarea interoperabilitatii dinamice a serviciilor Web. Componentele principale care au fost deja proiectate si implementate in WSMO sunt: Core Component, Resource Manager, Discovery, Selection, Data and Process Mediator, Communication Manager, Choreography Engine, Web Service Modelling Toolkit si Reasoner [].

Initiativa WSMO a supus abordarea WSMO si specificatiile WSMO, WSML, respectiv WSMX la W3C in 2005 pentru a deveni un cadrul formal si standard pentru servicii Web semantice.

3.3. Abordarea OWL-S. Coalitia OWL-S consta dintr-un grup de cercetatori de la Stanford, SRI, Maryland, College Park, Carnegie Mellon si alte institutii implicate in domeniul cercetarii Web-ului semantic. Scopul coalitiei OWL-S a fost de a defini o ontologie generica pentru servicii Web care sa permita descrierea proprietatilor si capabilitatilor serviciilor Web intr-o forma interpretabila de computer, neambigua. Ca si limbaj pentru reprezentarea ontologiei a fost ales OWL [26]. OWL-S permite automatizarea urmatoarelor task-uri: descoperirea serviciilor Web, compunerea, monitorizarea si executiei. OWL-S ofera un cadru generic pentru a descrie orice tip de serviciu Web. In OWL-S serviciile sunt clasificate in doua categorii: servicii primitive (servicii elementare); servicii complexe (servicii compuse din mai multe servicii primitive). Ontologia are trei sub-ontologii principale - *profile* (ce face serviciul), *process* (cum sa interactionezi cu serviciul) si *grounding* (cum sa invoci serviciul). Toate aceste sub-ontologii sunt legate la conceptul top-level, ***Service***. In OW-S *Service* ***presents*** a *Profile*, is **describedBy** *serviceModel* and **supports** *ServiceGrounding*.

4. Elaborarea si implementarea modelului de ontologiei de domeniu

Pentru dezvoltarea de servicii Web semantice am elaborat doua modele de ontologii de business: unul pentru adnotarea de servicii Web semantice din domeniul trasabilitatii produselor alimentare, iar altul pentru adnotarea unor servicii Web din domeniu „online book store". Urmatoarele criterii au fost luate in considerare in proiectarea acestor modele de ontologii de business: (i) ontologia trebuie sa ofere a vedere relevanta a domeniului; (ii) sa aiba o structura ce poate fi cu usurinta adaptata/extinsa la alte domenii similare; (iii) sa defineasca taxonomii si relatii bine structurate pentru a se evita redundanta datelor si a se putea face inferente; (iv) sa furnizeze informatia semantica necesara pentru descoperirea, selectia si compunerea automata a serviciilor Web. Vom prezenta unul dintre cele doua modele propuse, si anume cel din domeniul trasabilitatii.

Modelul de ontologie propus pentru adnotarea semantica a serviciilor Web din domeniul trasabilitatii consta dintr-o ontologie nucleu si doua categorii de arbori taxonomici Business Service Description trees si Business Product Description trees. Business Service Description trees reprezinta toate conceptele asociate cu un serviciu implicat in trasabilitatea lantului alimentar, iar Business Product Description trees contine concepte specifice asociate cu produse furnizate de un actor de business (numele produsului, UPC (Universal Product Code), pret, cantitate). Modelul nostru contine *clase*, *proprietati* si *restrictii*. *Clasele* reprezinta concepte din domeniul trasabilitatii produselor din carne organizate in structuri ierarhice. Proprietatile definesc relatii intre concepte sau restrictii. Relatiile existente in modelul nostru sunt relatii ierarhice/asociative. Relatiile ierahice sunt relatii de tip "is-A" si "part-Of". Modelul ontologic propus furnizeaza informatia semantica necesara pentru a descrie fiecare serviciu din punct de vedere al intrarilor, iesirilor si functionalitatii pe care o ofera.

4.1. Ontologia nucleu. Ontologia nucleu defineste sase concepte generice (fig.1): Business Actor, Service, Service Input, Service Output, Product and Feature. Dupa cum se poate vedea, in modelul nostru pot fi indentificate doua domenii:

- Business Service Descriptions (BSD) din care fac parte Business Actor, Service, Service Input, Service Output ;
- Business Product Descriptions (BPD) din care fac parte Product si Feature.

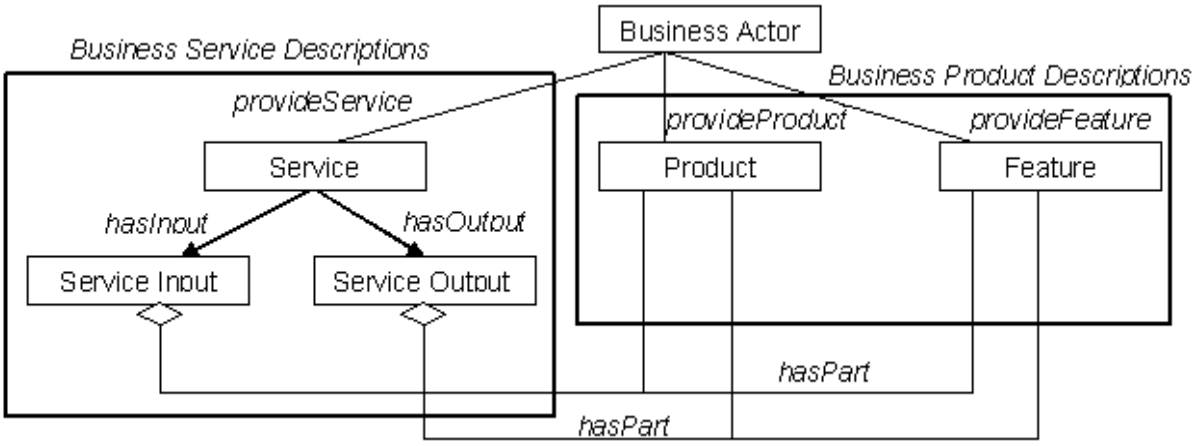

Figura 1: Ontologia Nucleu

O descriere detaliata a celor 6 concepte generice si a relatiilor dintre ele este prezentata in cele ce urmeaza:

Actorul de business reprezinta agentul economic implicat in trasabilitatea lantului alimentar (ex. Producator, Transporator, etc.). Fiecare dintre acestia ofera **Servicii**, **Produse** (disponibile prin intermediul unui serviciu Web) si informatii despre produsele pe care le ofera grupate sub conceptul generic **Caracteristici**. Relatiile dintre Actorul de Business si Serviciu, Produs, respectiv Caracteristici sunt relatii de asociere cu ordin de multiplicitate one–to-many (in abordarea noastra un Actor de Business ofera cel putin un serviciu, produs si caracteristici). **Serviciu** reprezinta o clasificarea ierahica a serviciilor furnizate de catre un Actor de Business implicat in trasabilitate lantului alimentar. Fiecare Serviciu este caracterizt prin **Intrari** si **Iesiri**. Relatiile dintre **Serviciu**, si **Service Input**, respectiv **Serviciu Output** sunt relatii functionale. **Produs** reprezinta o clasificare ierahica a produselor oferite de catre un actor de business, iar **Feature** o clasificare ierarhica a caracteristicilor produselor. Product si Feature sunt in relatie *hasPart* cu Service Input respectiv Service Output.

Adaptarea modelului de ontologie propus la un domeniu de business specific se face prin introducerea de sub-arbori de concepte specifici domeniului sub nodurile corespunzatoare din aceasta ontologie nucleu. Pornind de la ontologia nucleu propusa, a fost construita o ontologiei de business pentru trasabilitatea produselor din carne. Constructia ontologiei de domeniu a constat in dezvoltarea arborilor de concepte BSD si BPD conform cu constrangerile si regulile de business specificate. Arbori BSD au fost dezvoltati in editorul de ontologii Protege [15], iar arbori BPD au fost construiti automat din descrieri textuale furnizate de site-uri Web [22][23] ale companiilor producatoare de produse din carne din Romania. Ca si limbaj de specificare pentru ontologia dezvoltata s-a folosit OWL-DL, deoarece logicile permit verificarea consistentei ontologiei. Pentru constructia automata a arborilor BPD am experimentat doua abordari: una bazata pe o retea neuronala nesupervizata(abordarea statistica), iar alta bazata pe sabloane lexico-sintactice. Rezultatele obtinute au fost satisfacatoare in ambele cazuri.

Referinte:

[1] Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M., Sheth, A., & Verma, K., *Web Service Semantics – WSDL-S*. Technical note, 2005. Available at http://lsdis.cs.uga.edu/library/download/WSDL-S-V1.html.

[2]Ambite, J.L. and Weathers, M., *Automatic composition of aggregation workflows for transportation modeling*, Proceedings of the 2005 National Conference on Digital Government Research, Atlanta, GA, USA, May.

[3]Benatallah, B., Dumas, M., Sheng, Q. and Ngu, A, *Declarative composition and peer-to-peer provisioning of dynamic web service*', Proceedings of the International Conference on Data Engineering (ICDE), San Jose, CA, 2002.

[4] deBruijn, J, *The Web Service Modelling Language WSML*.WSMO Deliverable D16, WSMO Final Draft v0.2., 2005.

[5] Casati, F., Ilnicki, S., and Jin, L., *Adaptive and dynamic service composition in EFlow*. In Proceedings of 12th International Conference on Advanced Information Systems Engineering(CAiSE), Stockholm, Sweden, June 2000. Springer Verlag.

[6] Cimpian, E., Moran, M., Oren, E., Vitvar, T., & Zaremba, M., Overview and Scope of WSMX. Technical report, WSMX Working Draft, 2005.

[7] Dong,J., et al., *Verification of Computation Orchestration via Timed Automata*, in Proc. ICFEM'06, ser. LNCS, 2006.

[8] D´ıaz, G., et al., *Automatic Translation of WS-CDL Choreographies to Timed Automata*, in Proc. WS-FM'05, ser. LNCS, vol. 3670. Springer, 2005, pp. 230–242

[9] Fensel, D., & Bussler, C., The Web Service Modelling Framework WSMF. Electronic Commerce Research and Applications, 2002.

[10]Fujii, K. and Suda, T., *Dynamic service composition using semantic information*, Proceedings of the Second International Conference on Service Oriented Computing *(ICSOC'04)*, New York, USA, November, 2004.

[11] Lausen, H., Polleres, A., & Roman, D., *Web Service Modelling Ontology* (WSMO), 2005.

[12]Martin, D., Paolucci, M., and Wagner, M., *Towards Semantic Annotations of Web Services: OWL-S from the SAWSDL Perspective*, The 4th European Semantic Web Conference, 2007 (ESWC 2007)

[13]McIlraith, S., and Son, T.C., *Adapting Golog for composition of Semantic Web services*, In Proceedings of the 8th International Conference on Knowledge Representation and Reasoning(KR2002), Toulouse, France, April 2002

[14]McDermott, *Estimated-regression planning for interactions with Web services*, In Proceedings of the 6th International Conference on AI Planning and Scheduling, Toulouse, France, 2002. AAAI Press.

[15]B. Medjahed, A. Bouguettaya, and A. K. Elmagarmid, *Composing Web services on the semantic Web*, The VLDB Journal, 12(4), November 2003.

[16]Narayanan S. and McIlraith, S., *Simulation, Verification and Automated Composition of Web Services*," in WWW'02. ACM, 2002, pp. 77–88.

[17] Patil, A., Oundhakar, S., Sheth, A., Verma, K., *METEOR-S Web Service Annotation Framework*, Proceeding of the World Wide Web Conference, July 2004

[18]Ponnekanti, S.R., and Fox., A, *SWORD: A developer toolkit for Web service composition*, In Proceedings of the 11th World Wide Web Conference, Honolulu, HI, USA, 2002.

[19] Roman, D., Lausen, H., & Keller, U., *Web Service Modelling Ontology (WSMO)*. WSMO Working Draft D2v1.2, http://www.wsmo.org/, April 2005.

[20] Sabou, M., *Building Web Service Ontologies*, Ph. D., Thesis, Knowledge Representation and Reasoning Group of the Vrije Univeristeit, Amsterdam., 2005

[21]Sirin, E., Hendler, J., and Parsia, B., *Semi-automatic composition of Web services using semantic descriptions*. In Proceedings of Web Services: Modeling, Architecture and Infrastructure workshop in conjunction with ICEIS2003, 2002.
[22]Kona, S., Bansal, A., Gupta, G., *Automatic Composition of SemanticWeb Services*, ICWS2007
[23]VanderMeer, D.E., Datta, A., Dutta, K., Thomas, H.M., Ramamitham, K. and Navathe, S.B., *FUSION: a system allowing dynamic web service composition and automatic execution*, *Proceedings of IEEE International Conference on Electronic Commerce (CEC 2003)*, Athens, Greece, May.
[24] Verma, K., *Configuration and adaptation of Semantic Web Processes*. *Service Matching with Contextualised Ontologies*, Ph.D. Thesis, Department of Computer Science, The University of Georgia, August 2006
[25]Zhang, J., et al., *WS-Net: A Petri-net Based Specification Model for Web Services*, *Proceedings of IEEE International Conference on Web Services (ICWS'0)4*, 2004.
[26] The OWL Services Coalition, OWL-S 1.1. Available from http://www.daml.org/services/owl-s/1.1/, Retrieved on November 2004.
[27]http://www.w3.org/TR/sawsdl/
[28]http://lsdis.cs.uga.edu/projects/meteor-s/
[29]Bertoli, P., et. al, Integrating Discovery and Automated Composition: from Semantic Requirements to Executable Code, ICWS2007.
[30]Wu, D., Sirin, E., Hendler, J., Nau, D., and Parsia.B., *Automatic Web services composition using SHOP2*. In Workshop on Planning for Web Services, Trento, Italy, June 2003.
[31]Waldinger, R., *Web agents cooperating deductively*. In Proceedings of FAABS 2000, Greenbelt, MD, USA, April 5–7, 2000, volume 1871 of Lecture Notes in Computer Science, pages 250–262. Springer-Verlag, 2001.
[32]Lämmermann, S., *Runtime Service Composition via Logic-Based Program Synthesis*, PhD thesis, Department of Microelectronics and Information Technology, Royal Institute of Technology, June 2002
[33] J. Rao, P. K¨ungas, and M. Matskin, *Logic-based Web services composition: from service description to process model*. In Proceedings of the 2004 International Conference on Web Services, San Diego, USA, July 2004. IEEE