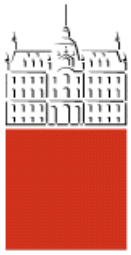
Univerza
v Ljubljani
Fakulteta
za gradbeništvo
in geodezijo

Jamova 2
1000 Ljubljana, Slovenija
telefon (01) 47 68 500
faks (01) 42 50 681
fgg@fgg.uni-lj.si



Univerzitetni program Gradbeništvo,
Hidrotehniška smer

Kandidat:
Vladimir Mijatović

Semantične spletne storitve

Diplomska naloga št.: 2831

Mentor:
prof. dr. Žiga Turk

Somentor:
Etiel Petrinja

Ljubljana, 29. 9. 2005

BIBLIOGRAFSKO – DOKUMENTACIJSKA STRAN IN IZVLEČEK

UDK: **004.42:004.738.5:624+69(043.2)**

Avtor: **Vladimir Mijatović**

Mentor: **prof. dr. Žiga Turk**

Naslov: **Semantične spletne storitve**

Obseg in oprema: **106 str., 38 sl., 8 pril.**

Ključne besede: **Spletna storitev, semantični splet, semantika, ontologija, WSDL, OWL-s, WSDL-s, WSMO**

Izvleček

Gradbena industrija zaradi svojih specifičnih lastnosti in delovanja ne izkorišča potenciala, ki ga trenutno ponuja informacijska tehnologija. Tako se še vedno išče rešitev, ki jo bo ta industrija lahko sprejela in ki bo zmanjšala negativne posledice nehomogenega razvoja informacijske tehnologije znotraj same panoge. Ena izmed možnosti se kaže v virtualni organizaciji, za katero se predvideva, da bo predstavljala model organizacije poslovanja v prihodnosti in bo verjetno prinesla opazne spremembe tudi v gradbeništvu. Eno izmed pomembnejših vlog v modelu virtualne organizacije imajo semantične spletne storitve, ki predstavljajo nadgradnjo standardnih spletnih storitev in so namenjene za uporabo znotraj virtualne organizacije. Glavni pridobitvi njihove uporabe sta izboljšanje procesa odkrivanja ustreznih storitev in avtomatično povezovanje več storitev v delovne procese.

Diplomska naloga na kratko opisuje osnovne koncepte in tehnologije spletnih storitev ter semantičnega spleta, iz katerega tudi izhaja ideja dodatnih semantičnih opisov. Jedro naloge predstavlja opis treh trenutno najbolj razvitih možnosti nadgradnje spletnih storitev v semantične spletne storitve. Ti obravnavani modeli so OWL-s, WSDL-s in WSMO. Narejen je tudi praktični primer na osnovi obstoječe spletne storitve, preko katerega je posredno predstavljen tudi del programske opreme, ki se trenutno uporablja za izdelavo semantičnega opisa, na koncu pa je opisan še scenarij uporabe takšnih storitev v gradbeniški virtualni organizaciji.

BIBLIOGRAPHIC – DOCUMENTALISTIC INFORMATION

UDC: **004.42:004.738.5:624+69(043.2)**
Author: **Vladimir Mijatović**
Supervisor: **prof. dr. Žiga Turk**
Title: **Semantic Web Services**
Notes: **106 p., 38 fig., 8 ann.**
Key words: **Web service, Semantic web, semantics, ontology, WSDL, OWL-s, WSDL-s, WSMO**

Abstract

Due to some of its specific characteristics the civil engineering industry is not using the potential that is currently available in information technology. The search for a suitable solution that would be acceptable for the industry and would be able to reduce the negative effects of different levels of the use of information technology among participants is still going on. A potential solution is seen in the Virtual Organisation, which is believed to be the future model of bussiness organisation and could bring noticeable changes even into the civil engineering industry. The semantic web services, which represent an upgrade of standard web services hold an important role in the Virtual Organisation where they are given in use to participants of Virtual Organisation. Main benefits of using them are improvement in discovery process of suitable services and a posibility of automatic composition of different services into workflows.

This diploma describes in a short way some basic elements and techologies of web services and semantic web, that started the idea of extra semantic descriptions. The main part gives a description of the three most sofisticated methods of upgrading web services into semantic web services. These models are the OWL-s, WSDL-s and WSMO. There is also an example based on an existent web service and through which some of the software for creating semantic web services is also being descibed. In the end an example of use in civil engineering virtual organisation is described.

ZAHVALA

Za pomoč in nasvete pri izdelavi diplomske naloge se iskreno zahvaljujem prof. dr. Žigi Turku in somentorju Etielu Petrinji.

Zahvalil bi se mojim domačim za ljubezen in podporo, ki so mi jo dajali skozi skozi vsa ta leta in mi ne glede na okoliščine vedno stali ob strani.

Na tem mestu bi se zahvalil tudi Damirju Mesariću. Skupne izkušnje iz študentskih let so v moj slovar poleg besede prijatelj vpisale *Mesko*.

Vladimir

.

KAZALO VSEBINE

KAZALO SLIK

KAZALO PRILOG

OKRAJŠAVE IN SIMBOLI

1) DAML – DARPA Agent Markup Language
2) HTTP – Hyper Text Transfer Protocol
3) HTML - Hyper Text Markup Language
4) IFC - Industry Foundation Classes
5) IOPE – input-output-precondition-effect
6) IT – Informacijska tehnologija
7) mySQL – mySQL podatkovna baza
8) OIL - Ontology Inference Layer
9) OWL – Ontology Web Language
10) OWL-s - Ontology Web Language Semantic
11) P2P – Peer to Peer
12) PHP – PHP Hxpertext Language
13) RDF - Resource Description Framework
14) RDF/S – Resource Description Framework Schema
15) SOAP – Simple Object Access Protocol
16) SQL – Standard Query Language
17) ŠIS – Študentski informacijski sistem
18) UDDI - Universal Description, Discovery and Integration
19) URI – Uniform Resource Identifier
20) VO – Virtualna Organizacija
21) W3C – World Wide Web Consortium
22) WSDL - Web Services Description Language
23) WSDL-s – Web Services Description Language with Semantic
24) WSMF – Web Services Modeling Framework
25) WSML – Web Services Modeling Language
26) WSMO – Web Services Modeling Ontology
27) WSMX – Web Services Execution Environment
28) XML – eXtensible Markup Language
29) XQuery – XML Query Language
30) XSD - eXtensible Markup Language Schema
31) XSLT – XML document transformation language

1 UVOD

Gradbena industrija trenutno izkorišča zelo malo potenciala, ki ga ponuja informacijska tehnologija. Razlogi za to so večinoma v specifičnosti panoge [11]:

- razdrobljenost
- brez dominantnega udeleženca, ki bi vsilil uporabo informacijske tehnologije
- komunikacija na projektu poteka pretežno med udeleženci na projektu, ponavadi brez končnega naročnika in tako ni pogodbeno kontrolirana
- večina udeležencev hkrati deluje na več drugih projektih
- panoga je projektno orientirana, kar pomeni, da bi morala biti investicija v informacijsko tehnologijo povrnjena do zaključka projekta (za večino ali kar za vse udeležence)
- začasno in ponavadi kratkotrajno poslovno sodelovanje med soudeleženci pri projektu, ki morda ne bodo nikoli več sodelovali

'Zaradi te raznoterosti udležencev se je informacijska tehnologija dobro uveljavila samo na določenih področjih, na drugih pa skoraj nič. Tak nehomogen razvoj ima vrsto negativnih posledic. Komunikacija znotraj panoge je daleč za potencialnimi možnostmi, posledica slabe komunikacije je časovna neučinkovitost, napake in druga neskladja v viziji dela celotne panoge.' [8] Druga težava ponavadi nastopi, tudi če udeleženci že imajo vpeljano kakšno IT (6) podporo in sicer zaradi nekompatibilnosti uporabljenih tehnologij.

Ena bolj pogosto omenjanih potencialnih rešitev je VO (20), za katero se predvideva, da bo v prihajajočem obdobju z razvojem informacijske tehnologije počasi izpodrinila industrijsko organizacijsko obliko in tako postala prevladujoča organizacijska oblika delovanja in sodelovanja podjetij [4]. Pa ne samo na področju gradbeništva, ampak kar na splošno. Za VO je predlaganih več definicij, eno izmed njih je napisal Pascal Sieber leta 1998: 'Bolj kot je družba ali skupina družb zmožna priskrbeti svoje izdelke in storitve neodvisno od položaja in od vsakršne časovne omejitve, uspešnejša je. Informacijska tehnologija je eden od najpomembnejših dejavnikov pri podpori prostorske in časovne neodvisnosti. Zato definiram virtualno organizacijo kot vsako institucionalizirano obliko poskusa zagotavljanja svojih izdelkov in storitev časovno in prostorsko neodvisneje od svojih tekmecev.' Vzroki za razvoj takšnega modela so:

- zmanjšanje števila stalno zaposlenih in zaposlovanje samo po potrebi za določen čas ali za določen projekt
- večja fleksibilnost delovnega časa in kraja npr. starš lahko dela tudi od doma, če je narava dela temu ustrezna
- povpraševanje po visoko specializiranih osebah je vse večje, te osebe pa ne morejo biti fizično prisotne na več krajih hkrati. Informacijska tehnologija danes nudi že veliko dobro razvitih možnosti za komunikacijo na daljavo, tako lahko posamezni strokovnjak sodeluje na več projektih hkrati, ne glede na prostorsko porazdeljenost le-teh.
- stalno padanje cen računalniške in telekomunikacijske tehnologije

Pojem VO je omenil Mowshowitz že leta 1986, vendar je šele z razvojem računalniške tehnologije postal operativno izvedljiv in učinkovit organizacijski model [4].
Tako se v zadnjem času precej intenzivno razvijajo ogrodja (*grid*), ki bodo predstavljala tehnološko infrastrukturo za delovanje VO. Ogrodja so sestavljena iz skupine med seboj povezanih računalnikov, kjer so za razliko od računalniških gruč posamezni računalniki in druga strojna oprema precej heterogeni, prav tako so različne tudi programske opreme posameznih komponent. Od P2P (11) omrežja, ki je v nekaterih pogledih tudi podoben ogrodju pa se slednja razlikujejo v nadzoru nad posameznimi komponentami, ki je lahko natančno opredeljen. Po definiciji je ogrodje 'poskus koordiniranja deljenja virov in reševanja nalog v dinamičnih in večustanovnih navideznih (virtualnih) podjetjih' [6]. Funkcijsko jih lahko razdelimo na računska, podatkovna, storitvena in avtonomna, od katerih so implementacije računskih in podatkovnih ogodij že na voljo, storitvena so v prototipni fazi, medtem ko so avtonomna, ki naj bi imela samonadzorne sposobnosti, še v idejni fazi.

Pomemben del VO je tudi tehnologija, ki se že sedaj vse bolj uveljavlja kot možnost sodelovanja med različnimi podjetji in sicer so to spletne storitve. 'Tehnološki napredek, ki poteka v zadnjem desetletju s pojavom spletnih storitev in z vse večjim povezovanjem udeležencev trga, zahteva ustrezno prilagajanje novostim vseh, ki želijo ohraniti ali celo izbošljati svojo konkurenčnost pred ostalimi subjekti trga.' [8] Uporaba spletnih storitev ima veliko pozitivnih faktorjev [7]:

- uporaba drage programske opreme, ki si je manjše podjetje drugače ne more privoščiti
- majhna začetna investicija

- uporabniku ni treba vzdrževati programov, zaradi centralne ponudbe je pa vzdrževanje za ponudnika storitve hkrati precej lažje in cenejše
- enostavne možnosti preizkušanja posameznih programov
- stroški za informacijsko tehnologijo so predvidljivi in enakomerni
- možnost uporabe aplikacij tudi z manj zmogljivimi napravami, saj tečejo na gostiteljevih virih
- za ponudnike boljše možnosti nadzora nad uporabo aplikacij in piratstvom

Če združimo tri zgoraj navedene komplementarne rešitve in obravnavamo problem panoge gradbeništva, se povezava spletnih storitev z organizacijskim modelom VO na infrastrukturi storitvenega ogrodja ponuja kot model informacijsko-tehnološke rešitve za delovanje in sodelovanje podjetij na področju gradbeništva. Zanimivo je, da pomembne koncepte VO, kot so stalna fleksibilnost pri menjavi partnerjev, projektno usmerjeno delovanje in obstoj možnosti enkratnega sodelovanja v celotni dobi delovanja podjetij, gradbena industrija uporablja že vrsto let, le da je IT podpora navadno minimalna ali pa je sploh ni. Zato lahko tukaj sklepamo, da ker VO uporablja podoben način delovanja, kot ga ima moderno gradbeništvo in ker so mnoge druge panoge že dokazale veliko uporabnost IT, je model VO za razvoj gradbeništva lahko velik korak naprej.

Poleg razvoja samega storitvenega ogrodja in modela ustanavljanja virtualnega podjetja, ki sta oba še v prototipni fazi, morajo tudi spletne storitve narediti korak naprej. Njihova zasnova interakcije je dobra, kar kaže hitro naraščanje njihove uporabe v podjetjih, potrebno pa je rešiti problem neusklajenosti med njimi. Vpeljava skupnih standardnih konceptov znotraj posameznih VO (npr. VO za področje gradbeništva), na osnovi katerih bodo zasnovane storitve, bo izjemno izboljšala njihovo interoperabilnost, možnosti povezovanja ter možnosti avtomatične računalniške obdelave. Tako se bo zmanjšala količina posredovanja in potrebnega angažiranja pri uporabi storitve s strani uprabnikov. Nadgradnja standardnih spletnih storitev, ki bo omogočala naštete možnosti so SEMANTIČNE SPLETNE STORITVE. Ideja nadgradnje je razvita iz semantičnega spleta, ki se kaže kot naslednja razvojna stopnja interneta.

V nadaljevanju te diplomske naloge sem opisal koncept spletnih storitev, osnove semantičnega spleta, kot glavni del naloge pa sem opisal prednosti semantičnih spletnih

storitev in tri trenutno najbolj razvite možnosti njihove izvedbe. V praktičnem primeru sem na dejanski spletni storitvi prikazal postopek izdelave dodatnega semantičnega opisa.

2 SPLETNE STORITVE

Vodilna ideja odbora W3C (21) organizacije za spletne storitve :
'Spletne storitve ponujajo standardizirana sredstva interoperabilnosti med različnimi aplikacijami, ki tečejo na različnih platformah in/ali ogrodjih. Spletne storitve so označene z odlično interoperabilnostjo in razširljivostjo zahvaljujoč uporabi XML (28), hkrati pa se jih lahko kombinira na različne načine z namenom doseganja rezultatov kompleksnih operacij. Programi, ki povezujejo preproste storitve lahko med seboj komunicirajo in tako ponudijo visoko razvite storitve z dodano vrednostjo'. [34]

2.1 Kaj so spletne storitve

Spletne storitve so standardiziran način ustvarjanja interoperabilnosti med različnimi aplikacijami, ki tečejo na različnih platformah in ogrodjih. Preko vmesnikov in navodil za interakcijo omogočajo združevanje namenskih in geografsko porazdeljenih programov. Spletna storitev ima glede na ostale storitve popolno avtonomijo s stališča programskih jezikov, operacijskih sistemov in platform, na katerih se izvajajo. Ta namen preseganja tehnoloških omejitev različnih, med seboj nezdružljivih sistemov je bil tudi eno glavnih vodil pri ustvarjanju standardov in ideje spletnih storitev. Dodatno je proces povezovanja in izvajanja storitev zaradi načina opisovanja in registriranja le-teh lahko popolnoma avtomatiziran.

Najbolj razširjena oblika spletne storitve je z XML podprta storitev, ki komunicira preko internet protokolov (najpogosteje HTTP (2)) ter pošilja in sprejema podatke v XML obliki. Kot semantično sredstvo za izmenjavo informacij ali vsebin med storitvami je vzet označevalni jezik XML, ki se vse bolj uveljavlja kot glavni standard za izmenjavo informacij med računalniškimi sistemi, pa tudi pa prenos ukazov in navodil v odprtih sistemih. Ta oblika spletne storitve je tudi obravnavana v nadaljevanju diplomske naloge.

2.2 Osnovne tehnologije XML za spletne storitve

Tri osnovne XML specifikacije za spletne storitve so [29] [24]:

- SOAP (15)
- WSDL (22)
- UDDI (18)

2.2.1 SOAP

SOAP je predmetna komunikacija, zasnovana okrog XML formata in
je namenjena izmenjavi strukturiranih informacij v decentraliziranem in geografsko porazdeljenem okolju. Protokol določa obliko in način izmenjave XML dokumentov ter navodila za obdelavo vsebine, izmenjava pa poteka preko HTTP ali kakšnega drugega internet protokola.

Sporočilo SOAP sestavljajo trije temeljni deli:

- *envelope* (ovojnica) - obvezna ovojnica določa začetek in konec sporočila
- *header* (zaglavje) - namenjeno je prenosu sporočilnih atributov, t.j. identifikatorjev, na osnovi katerih je mogoče razbrati razred ali kakovost storitve. Zaglavja ni obvezno vključiti v sporočilo.
- *body* (vsebina) - gre predvsem za navodila, kako naj se obravnava posamezno sporočilo ali vsebina, ko prispe na cilj (ali rokuje v vozlišču).

2.2.2 WSDL

WSDL je jezik za opis načina in oblike komunikacije oz. načina uporabe spletne storitve. Elementi tega jezika vsebujejo opis podatkov na osnovi XML shem, prav tako pa so vključena navodila za uporabo operacij nad podatki. S tem pošiljatelj ve, kako podatke pripraviti in jih poslati, prejemnik pa, kako jih najprej interpretirati in nato obdelati. Povezovalno sredstvo je SOAP, zato ponavadi WSDL opis vsebuje tudi povezavo s tem komunikacijskim protokolom. Če sta ponudnik in uporabnik storitve seznanjena z WSDL opisom, je lahko za samo storitvijo poljuben programski vmesnik.
Preslikavo med vmesnikom spletne storitve in namenskim programom v ozadju interpretirajo elementi WSDL. Temeljni elementi so [24]:

- definicije (*definitions*) – to je osnovni element vsakega WSDL dokumenta, definira pa ime storitve, domene (*namespaces*), ki se bodo uporabljali preko celega dokumenta, prav tako pa vsebuje tudi vse ostale elemente dokumenta
- vrste podatkov (*types*) – določajo, za kakšne vrste podatkov gre
- sporočilo (*message*) – opisuje enosmerno sporočilo, pa naj bo to zahteva ali odgovor. Definira tudi ime sporočila in vsebuje 'message part' elemente, ki se sklicujejo na parametre sporočila ali vrnjene vrednosti sporočila

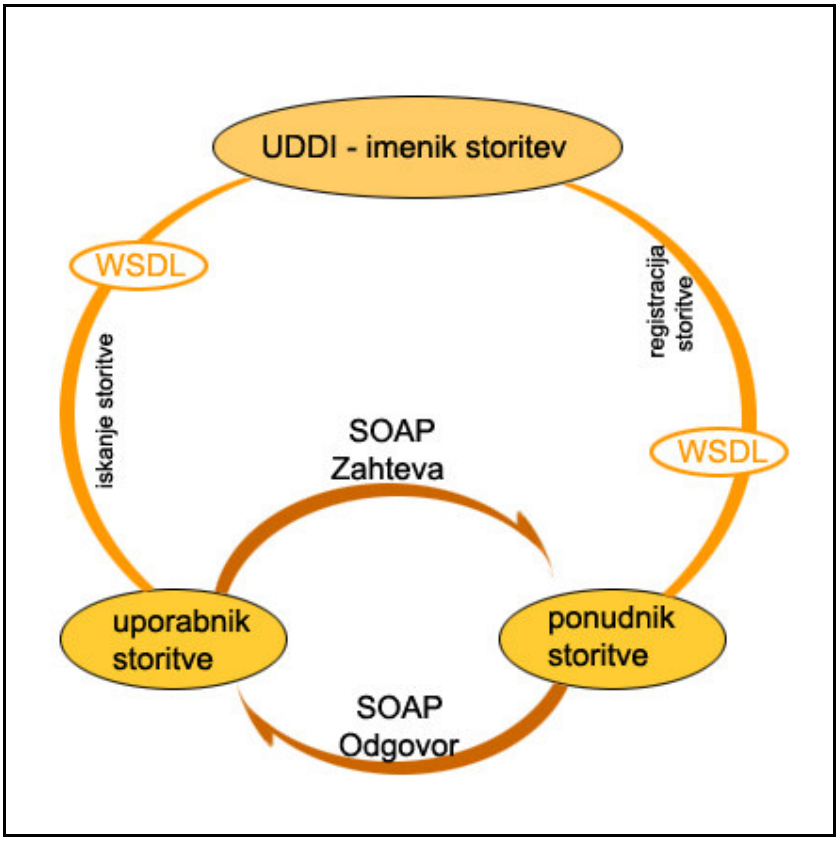
- oblika vhoda (*portType*) –vsebuje več message elementov, ki skupaj opisujejo enosmerno ali krožno operacijo. Primer: portType lahko kombinira eno zahtevo in en odgovor v eno zahteva+odgovor operacijo, pogosto pa definira več operacij.
- povezava (*binding*) – konkretno opisuje kako bo storitev izvedena.
- storitev (*service*) – vsebuje lokacijo, preko katere se bo izvedla obravnavana storitev. Pogosto je to naslov domene za sprožitev SOAP storitve.

2.2.3 UDDI

UDDI je imenik za objavljanje in iskanje spletnih storitev. Na tej točki si lahko priskrbimo temeljne podatke o storitvi, primerjamo storitve med seboj…
Imenik je podobno kot telefonski imenik razdeljen na bele, rumene in zelene strani. Bele strani hranijo ime, naslov, kontakte, spletni naslov in identifikacijsko oznako. Rumene strani osnovni nabor informacij razširjajo s področjem poslovanja, lokacijo in izdelki, vključno z davčnimi podatki. Tehnične napotke in informacije, kako komunicirati in uporabljati storitev, vključno z opisom WSDL pa najdemo med zelenimi stranmi.

2.3 Koncept delovanja spletnih storitev

Uporabnik se prijavi v UDDI imenik in sproži iskanje po storitvi na osnovi ključnih besed, imenik pa kot rezultat vrne podatke o prijavljenih storitvah, ki ustrezajo iskalnim pogojem, uporabnik pa lahko nato izbere eno storitev, ki ustreza njegovim zahtevam in analizira njen WSDL opis. Na osnovi opisa sestavi zahtevo, ki jo v 'SOAP/Zahteva' obliki pošlje npr. preko HTTP protokola ponudniku spletne storitve. SOAP program za storitvijo sprejme zahtevo in jo analizira, če je pravilno zastavljena. Iz zahteve se izvlečejo vhodni podatki in če je vse v redu, se izvrši operacija storitve na osnovi podatkov, ki jih je poslal oddaljeni uporabnik. Odgovor ponudnika storitve je lahko prav tako kot zahteva poslan preko HTTP protokola v 'SOAP/Odgovor' obliki. Uporabnik prejme odgovor in ga prebere s svojim SOAP programom. Če je vse potekalo po načrtu, je uporabnik na svojo zahtevo dobil odgovor, ki ga je iskal in ki ga lahko nadalje uporabi v svoji obdelavi podatkov.



Slika 1: Koncept delovanja spletnih storitev

3 SPLETNE STORITVE V SEMANTIČNEM SPLETU

3.1 Vzroki za nastanek in razvoj semantičnega spleta

Leta 1989 je Tim Barners-Lee, takrat kot programer v podjetju CERN predlagal, kako bi izboljšali način zapisa informacij o dokumentih, v katerih so ljudje dokumentirali in spravljali rezultate svojega dela. *"The aim would be to allow a place to be found for any information or reference which one felt was important, and a way of finding it afterwards.*" [23] Kar je pravzaprav hotel je bil internet. Izdelal je prvi brskalnik, strežnik in zasnoval protokol, ki je omogočal komunikacijo med računalniki. Danes je v internet omrežju več sto tisoč računalnikov in več milijard dokumentov, ki gradijo največje omrežje na svetu. Ob hitrem naraščanju števila dokumentov se je sprva predvidevalo, da bo prevelika množica dokumentov neobvladljiva za preiskovanje in da uporabniki omrežja v večini primerov ne bodo našli tistega dokumenta, ki bi vsebinsko ustrezal njihovim iskalnim pogojem. To bi pomenilo, da internet ni izpolnil svoje primarne naloge. Vendar je z razvojem računalništva ta velika količina dokumentov postala obvladljiva s pomočjo spletnih iskalnikov. Med več milijardami zapisov v svojih registrih nam na osnovi dobro izbranih ključnih besed vrnejo tako dobre rezultate, da navadno dobimo povezavo do vsebinsko ustreznega dokumenta že med prvimi 50-imi rezultati. Seveda če takšen dokument sploh obstaja in če je poleg tega še registriran v iskalniku.
Že pred takšnim razvojem iskalnikov (1999) pa je ustanovitelj interneta, takrat kot predsednik W3C organizacije, sprožil razvoj nadgradnje interneta. V enem svojih člankov je zapisal: *"For the documents in our lives, everything is simple and smooth. But for data, we are still pre-Web. "* [2] [31]

Kar si je pravzaprav zamislil je SEMANTIČNI SPLET ali 'splet podatkov' ali tudi 'računalniško razumljiv splet pametnih podatkov'. Velika večine vsebin na internetu je zasnovana tako, da je razumljiva ljudem, ampak računalnikom pa ne pomeni ničesar drugega kot skupina znakov, ki jo oni znajo prikazati. Računalniki so sposobni vsebino dokumenta razgraditi in do neke mere analizirati (npr. HTML (3) spletne strani razdelijo na glavo, sliko, povezavo na drugo stran…) ampak s tem načinom analize strukture dokumenta računalnik ne ve ničesar o semantiki (pomenu) teh elementov npr. 'To je domača stran Fakultete za gradbeništvo in geodezijo' ali 'Ta povezava kaže na stran predmeta Matematična analiza 1'.

Slaba stran računalnika (če pustimo ob strani umetno inteligenco, ki je še vedno v razvoju) je, da ne zna sam sklepati, dobra je pa to, da je izjemno hiter v procesiranju operacij in ta razlika z razvojem računalniške tehnologije v primerjavi s človekom samo še bolj narašča. Dober primer je npr. spletni iskalnik Google, ki v manj kot 1 s pregleda več milijard zapisov v svoji bazi podatkov. Takšno hitrost omogoča strukturiran način zapisa v bazi podatkov, ki so se izkazale kot izjemni napredek pri hranjenju, prikazovanju in analiziranju le-teh. Na tem hitrem računalniškem procesiranju dobro strukturiranih podatkov je zasnovan tudi del ideje semantičnega spleta. Drugi del ideje semantičnega spleta temelji na tem, da računalniki s pomočjo dodatnih semantičnih opisov in pravil za sklepanje razumejo podatke, ki jih obdelujejo. S tem bi izpolnili pogoje za avtomatično sklepanje računalnikov, kar bi omogočilo ogromno število procesiranih operacij, ki bi jih računalnik v primerjavi s človekom lahko opravil v izredno kratkem času. Tako bi 'semantični' iskalnik znal odgovoriti na vprašanje *'Ljubljana temperatura ?*', medtem ko danes za ta odgovor moramo najprej v iskalnik vnesti besede *'Ljubljana temperatura*', nato izbrati povezavo med rezultati, za katero mislimo, da bomo tam našli odgovor na svoje vprašanje in če smo dobro izbrali lahko med podatki na tej strani poiščemo podatek o temperaturi v Ljubljani. Rezultat obeh iskanj bi bil enak, potrebni čas in angažiranost uporabnika pa bi bila že v tem osnovnem primeru v semantičnem spletu precej manjša.

Iz zgoraj opisanega primera je razvidna uporabnost ideje, ampak kako jo realizirati?
Ključ do operativnega semantičnega spleta je po mnenju W3C organizacije zagotoviti jezik, ki opisuje tako podatke kot pravila sklepanja o podatkih in omogoča objavo baze znanja v splet iz kateregakoli sistema.

Dve za semantični splet pomembni tehnologiji sta že v uporabi: XML in RDF (13). Tretja pomembna tehnologija, ki dopolnjuje prvi dve pri omogočanju semantičnega spleta, je zbirka informacij imenovana ontologija.

3.2 Osnovni izrazi in tehnologije

3.2.1 XML

XML jezik se uveljavlja kot glavni standard za odprte sisteme, kar pomeni da je njegova glavna značilnost omogočanje deljenja informacij med posameznimi sistemi, ki so drugače med seboj

nekompatibilni. Vse tehnologije, na katerih bo temeljil semantični splet, bodo temeljile na XML, kar bo zagotavljalo osnovno stopnjo interoperabilnosti med računalniškimi sistemi.Omogoča pisanje svojih oznak (*tag)*, ki jih lahko aplikacije ali programi uporabijo za analizo. Vendar mora pred tem avtor programa, ki bo vršil analizo XML datoteke, poznati vsebinski namen uporabe vsake oznake posebej. Iz tega sledi, da lahko uporabniki z XML naredijo poljubno strukturo, ampak o tej strukturi pa ne povedo ničesar. Tukaj nastopi RDF, ki nadgradi XML v smislu izražanja pomena. [29]

Primer XML zapisa:

```
<author>
 <uri>http://www.w3c.org</uri>
 <name>Tim Barners-Lee</name>
</author>
```

3.2.2 Semantika

Semantika je izraz, ki se nahaja že v samem imenu semantičnega spleta. V lingvistiki je to nauk o pomenu besed, pomenoslovje. V računalništvu se izraz uporablja v kontekstu baze znanja, ki omogoča ustvarjanje bolj prilagodljivih, efektivnih in pametnih aplikacij v smislu, da so npr. sposobne na osnovi pomenov izrazov v bazi znanja poiskati skrita razmerja v kompleksnih predmetnih sistemih. Bolj preprosto povedano je semantični opis tista informacija, ki omogoča opis in definiranje predmetov, razredov… v jeziku, ki ga lahko računalniška aplikacija razume in povezuje naprej. Semantika se veliko uporablja tudi na področjih umetne inteligence, ekspertnih sistemov ipd. [31]

3.2.3 RDF

RDF omogoča izražanje pomena. Ta je napisan v obliki trojčkov – osebek (*subject)*, glagol (*verb),* predmet (*object)* – ki so lahko napisani v obliki XML oznak. V RDF jeziku dokument predvideva, da določene stvari (ljudje, Spletna stran…) imajo lastnosti (npr. 'je avtor od') z določenimi vrednostmi (druga oseba, druga spletna stran…). Izpostavilo se je, da je takšna struktura naraven način za opis velike večine podatkov, ki jih analizirajo računalniki. Osebek in predmet sta oba definirana z URI (19), prav tako tudi glagol, s čimer je vsakomur omogočeno, da

si za ta glagol določi svoj koncept in ga opiše na naslovu, podanem z URI lokacijo nekje na spletu.

Primer RDF:

<Aaron><pozna><Philip> - Aaron pozna Philipa.

<Philip><zaposlen><MIT> - Philip je zaposlen na MIT.

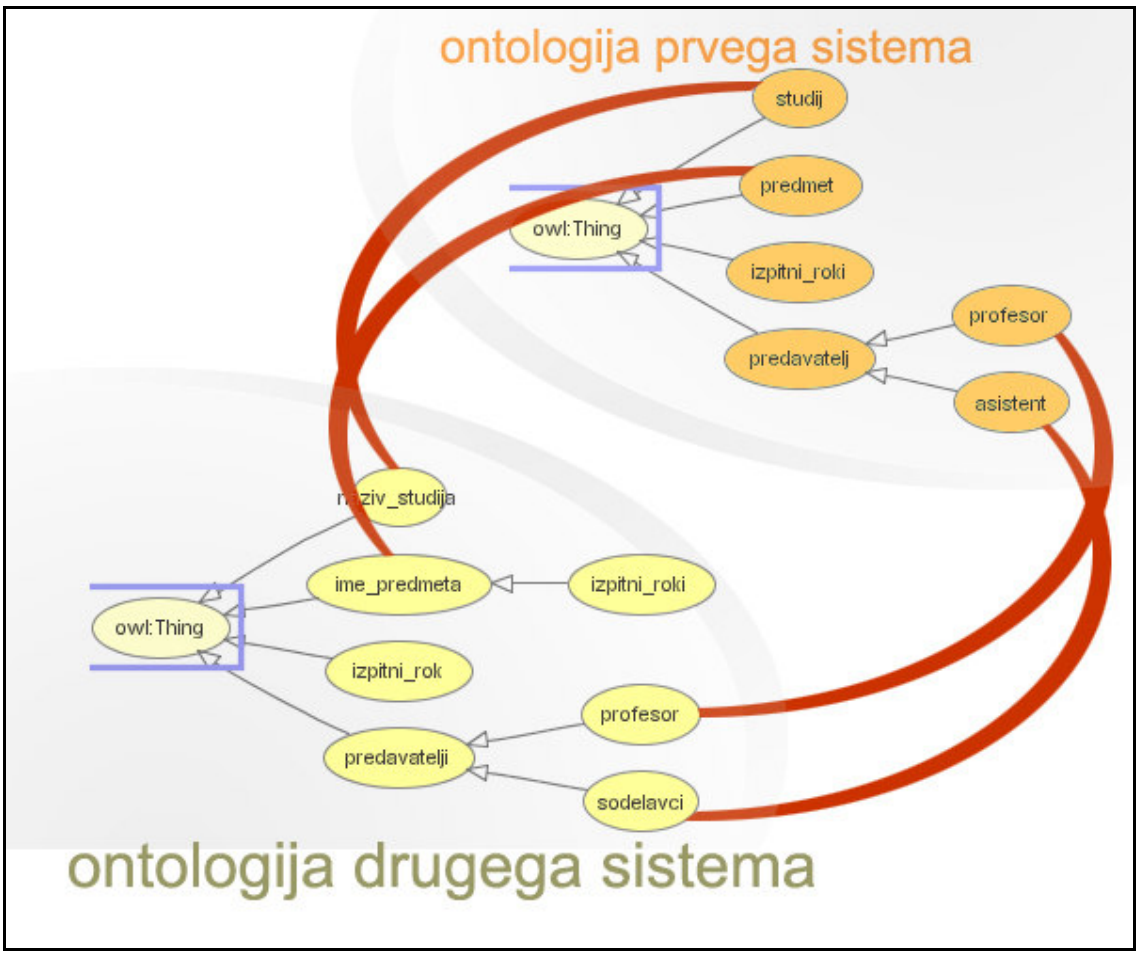
<MIT><spletna stran><http://web.mit.edu/> - MIT ima spletno stran http://www.mit.edu.

3.2.4 Ontologija

V filozofiji je ontologija veda, ki se ukvarja z strukturo obstoja, ki obravnava osnovo, vzroke in najsplošnejše lastnosti stvarnosti in se smatra za najbolj temeljno filozofsko disciplino. V računalništvu se je uveljavila kot izraz za dokument z zbirko informacij, ki definirajo izraze in razmerja med njimi ali kot pravi ena najbolj uporabljanih definicij je ontologija ' opis (kot formalna specifikacija programa) konceptov in njihovih razmerij ' [26]. Tipična spletna ontologija ima taksonomijo (stopenjsko razdelitev oz. načela) in zbirko razmerij (*inference rules*).Taksonomija definira razrede (*classes*) predmetov (*objects*) in relacije med njimi. Razredi, podrazredi in entitete so zelo močno orodje za uporabo v spletu. Razmerja v ontologiji dodatno pomagajo pri opisu strukture npr. ontologija lahko izrazi pravilo 'Če je poštna številka povezana z državno številko, in naslov uporablja to poštno številko, potem sledi, da je naslov povezan z državno številko.' Računalnik ne razume popolnoma, kaj pomenijo ti podatki, ampak s tem lahko manipulira z izrazi precej bolje v smislu da zagotavlja rezultate, ki imajo koristen pomen za uporabnika.

Ontologija omogoča tudi reševanje zelo pogostega problema, do katerega pride pri velikem številu ljudi in organizacij, ki delujejo na istem področju. Različni uporabniki lahko za iste koncepte (npr. poštna številka) uporabljajo različne oznake. Tukaj ontologija daje razlago, za kateri koncept je bil izraz uporabljen. Tako se lahko za program, ki išče po podatkih dveh baz definira dodatno razmerje, ki mu pove, da je pomen dveh različnih oznak (npr. 'poštna_številka' v eni bazi in 'postna_st' v drugi) enak in da to upošteva pri nadaljnji analizi podatkov ali povedano v računalniški obliki – semantične povezave med dvema bazama bi omogočile

transformacijo poizvedb (*queries*) po eni bazi v poizvedbe po drugi bazi. Podobna rešitev se omenja tudi v relacijskih bazah. Omenjene povezave kaže slika spodaj. [25]



Slika 2: Prikaz povezovanja enakih konceptov med ontologijami

Podatke v ontologiji bi lahko zelo dobro uporabili tudi iskalniki, ki bi izboljšali svojo natančnost pri iskanju, ko bi vključili tudi iskanje po konceptih, ne pa samo po korenu in primerjavi besed. Tako nekdo, ki npr. išče letovišče na Kajmanskih otokih ne bi dobil v izpisu rezultatov iskanja tudi povezav na živalske vrte, ki imajo v svojem terariju kajmane in krokodile.

3.2.5 OWL

OWL (9) je računalniški jezik, s katerim lahko eksplicitno predstavimo pomene izrazov in njihova medsebojna razmerja, kot smo jih zastavili z ontologijo. Namen OWL zapisa ontologije

je uporaba v primerih, ko bi morale informacije v dokumentih procesirati aplikacije. To je analogno primeru, ko bi bilo treba vsebino predstaviti samo človeku.
XML ima možnost predstaviti svoje oznake, RDF in RDF *Schema* ali RDF/S (14) sta zelo dobra za predstavljanje podatkov, ampak skupaj pa nimajo dovolj ekspresivne moči za izražanje pomena izrazov in semantike ali predstavitve terminologije, uporabljene v dokumentih na spletu. Zato je bil razvit OWL, ki ima večjo sposobnost izražanja računalniško berljive vsebine na spletu, nastal pa je kot združitev znanja in izkušenj pri razvoju dveh jezikov, ki sta se že ukvarjala z računalniškim prikazom ontologije: DAML (1) (DARPA Agent Markup Language) in OIL (8) (Ontology Inference Layer) web ontology language.
Obstajajo 3 različice OWL in sicer *OWL Lite*, *OWL DL* in *OWL Full*, ki se med seboj razlikujejo v svojih semantično ekspresivih sposobnostih in možnostih za računalniško obdelavo. Lastnosti sta obratno sorazmerni, kar pomeni da je OWL Full semantično najbolj bogat jezik, ne moremo pa zagotoviti, da ga bo znal računalnik v popolnosti pravilno interpretirati. Najbolj uporabljana je OWL DL različica. [38] Primer zapisa ontologije z OWL je podan kot priloga praktičnega primera.

Težava pri razvoju semantičnega spleta

Izraza semantika in ontologija sta po mnenju nekaterih raziskovalcev med največjimi krivci za počasnejše uvajanje semantičnega spleta, kot je bilo najprej predvideno in napovedano. Takšni izrazi zaradi svoje kompleksnosti in redke uporabe enostavno odvračajo veliko ljudi od nadaljnjega dela na tem področju. Ob prvi predstavitvi ali prebranih člankih jih prednosti semantičnega spleta pritegnejo k nadaljnjemu raziskovanju, ko se pa poglobijo v idejo in tehnologije pa sledi neizogibno srečanje s temi izrazi. Je pa potrebno poudariti, da je filozofija, ki stoji za izrazoma semantika in ontologija, zelo pomembna za razumevanje najosnovnejših konceptov semantičnega spleta in jo mora vsaj delno osvojiti vsak, ki namerava delati na tem področju.

Kot povzetek zgoraj opisanih izrazov in osnovnih tehnologij bi napisal ta stavek, ki zajema vse skupaj v opis končnega rezultata, ki ga bo predvidoma prinesel semantični splet. '*Če sta HTML in Internet prikazala vse dokumente kot eno ogromno knjigo, potem bodo RDF, shema in jeziki sklepanja (inference languages) prikazali vse podatke sveta kot eno ogromno bazo podatkov*.' [Tim Barners-Lee, Weaving the web,1999]

3.3 Spletne storitve s semantiko

Spletne storitve so bile primarno zasnovane za omogočanje interoperabilnosti med poslovnimi aplikacijami. Trenutne tehnologije, ki omogočajo uporabo spletnih storitev predvidevajo veliko poseganja in pomoči uporabnikov za uspešno delovanje storitve, kar je posledica dejstva, da integracija procesov zahteva poznavanje in razumevanje podatkov in funkcij vpletenih entitet. [29]
Semantične spletne tehnologije, podprte z jezikom OWL, pa lahko opišejo vsebino delovanja spletnih storitev v računalniku razumljivi obliki. Razumevanje vsebine opisa bi aplikacijam omogočalo boljše in lažje povezovanje spletnih storitev, rezultat bi pa bil precej manjše potrebe po človekovem poseganju v proces avtomatizacije spletnih storitev ter manjša količina potrebnega razumevanja podatkov in funkcij storitev.

Programi, ki razumejo vsebino semantičnih opisov spletnih storitev in ki lahko naredijo določene povezave med storitvami na osnovi njihovih semantičnih opisov se imenujejo agenti. Njihova naloga je pregled obstoječih zapisov v registru storitev in izločanje storitev, ki ustrezajo iskalnim pogojem uporabnika. Izločanje se lahko zaradi semantičnih opisov in ontologij, na katerih storitev temelji, vrši na osnovi konceptov in ne samo podobnosti imen. Tako lahko agenti izpostavijo najbolj ustrezne storitve in jih po potrebi povezujejo v sestavljene procese z nalogo reševanja kompleksnih problemov. Semantični splet ni 'učenje računalnikov da razumejo človeški jezik' ampak je to dobro zastavljen koncept opisa podatkov, kjer lahko agenti z dobro zastavljenimi vprašanji uporabniku vrnejo dobre odgovore. [15]

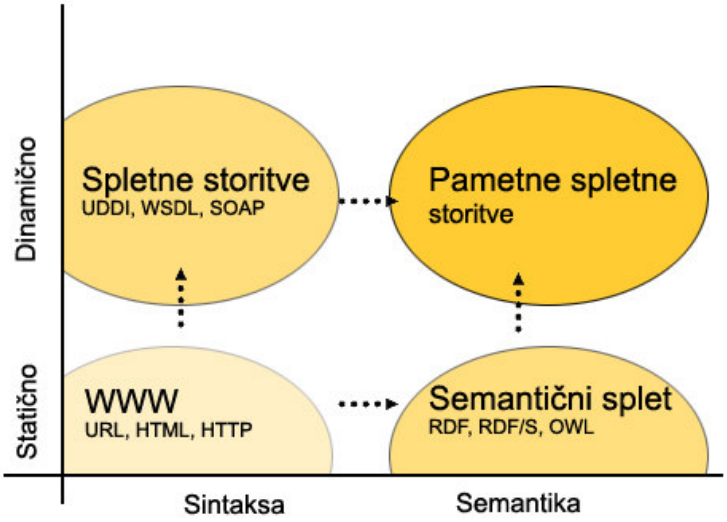
Semantične spletne storitve bodo s podporo tehnologij semantičnega spleta omogočale velik napredek pri treh pomembnih operacijah agentov [32]:

- Odkrivanje (*discovery*) spletne storitve – Program mora najprej biti sposoben avtomatično poiskati ustrezno spletno storitev. Niti WSDL niti UDDI ne omogočata agentu takšne razlage informacije o ponudbi spletne storitve, da bi jo on lahko razumel v smislu uporabljenih konceptov. Na osnovi dodatnega semantičnega opisa storitve pa bodo agenti sposobni takoj ugotoviti njihov namen.
- Sprožanje (*invocation*) spletne storitve – Agent mora biti zmožen tudi avtomatično ugotoviti, kako sprožiti uporabo storitve. Če storitev, da bi bila izpeljana do konca potrebuje več korakov interakcije, potem mora agent vedeti, kako od začetka do konca

poteka zaporedje korakov uporabe storitve. Semantična spletna storitev vsebuje seznam zahtev za agenta, da bi ta lahko sam sprožil uporabo. To je npr. lahko definiranje vhodnih in izhodnih podatkov procesa.

- Sestavljanje (*composition*) – agent mora biti sposoben (seveda ob zagotovljenih ustreznih podatkih, ki jih spet zagotavlja semantični opis) samostojno izbrati in združiti vrsto storitev, da bi izpolnil nalogo, ki mu je bila dana.

Iz zgoraj navedenih akcij bi primer uporabe semantičnih storitev bil videti kot zahteva '*Poišči storitev, ki prodaja letalske karte med Brnikom in Parizom in sprejema plačilo z Masters bančno kartico'* ali *'Opravi nakup letalske karte na spletni strani http://www.ryanair.com za let iz Londona v Barcelono dne 1.3.2006'*. Nobena od teh operacij z današnjimi spletnimi storitvami ni mogoča. Takšni ukazi bodo izvedljivi šele s semantičnimi spletnimi storitvami.



Slika 3: Položaj semantičnih spletnih storitev v razvoju interneta

4 TEHNIKE SEMANTIČNEGA ANOTIRANJA SPLETNIH STORITEV

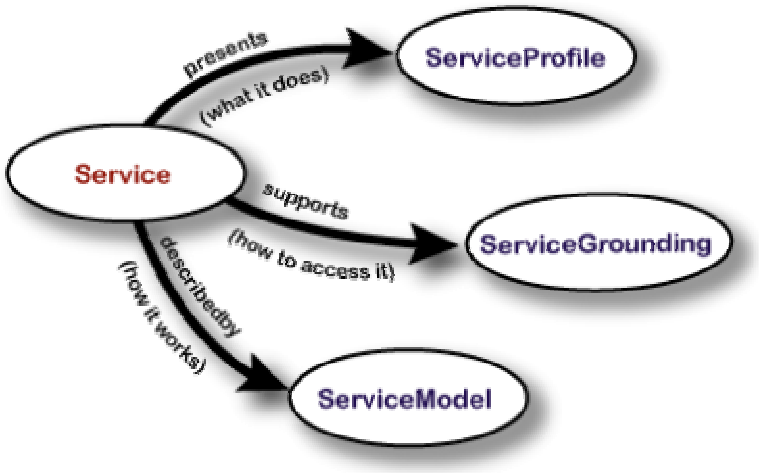
Osnovni način opisa spletne storitve je WSDL datoteka, ki za opis svojih podatkov v prvi vrsti uporablja XSD (30). Vendar XSD in WSDL sama ne zmoreta dovolj semantične sposobnosti za uspešno delovanje računalniških agentov, zato je potrebno osnovni način opisa spletnih storitev dopolniti s semantično anotacijo. Semantična anotacija pomeni zapis ontologije spletne storitve z OWL jezikom. S tem zagotovimo opis podatkov in operacij v obliki, ki jo agenti razumejo in lahko potem s temi podatki bolj konstruktivno manipulirajo in izpolnjujejo bolj kompleksne ukaze.
Za izvedbo anotacije bom opisal 3 tehnologije, ki se glede na razvojno stopnjo in podporo v skupnosti semantičnega spleta največkrat omenjajo. Te tehnologije so OWL-s (10), WSDL-s (23) in WSMO (26).

4.1 OWL-s

OWL-s izvira iz projekta DAML (http://www.daml.org). Projekt DAML se je uradno pričel Avgusta 2001, cilj pa je razviti jezik in orodja za podporo delovanja semantičnega spleta. Del programa, ki podpira semantične spletne storitve in skuša razviti metodo semantične anotacije spletne storitve se je najprej imenoval DAML-s. Ta je temeljil na DAML+OIL jeziku, ki je bil predhodnik OWL. Ob preimenovanju in nadgradnji DAML+OIL v OWL se je tudi jezik DAML-s preimenoval v OWL-s. Trenutna različica jezika je 1.1.

4.1.1 Zasnova modela

OWL-s uporablja pristop ustvarjanja dodatne plasti (*layer*) okrog opisa WSDL. Ta vsebuje semantično dobre opise konceptov, ki so uporabljeni v storitvi in s tem zadošča potrebam agentov za samodejno upravljanje s storitvijo. Ta plast je sestavljena iz vhodne referenčne točke za semantični opis spletne storitve, ki se imenuje *Service*. Service ima 3 lastnosti, ki so predstavljene kot razredi in ki vsebujejo informacije o storitvi: *ServiceProfile*, *ServiceModel* in *ServiceGrounding.* [35]



Slika 4: Glavna razdelitev ontologije OWL-s
Vir: [http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/]

4.1.2 OWL-s ontologija spletne storitve

ServiceProfile

ServiceProfile (*presents*) razred opisuje 'Kaj storitev naredi' in kot tak je namenjen oglaševanju storitve. Opis je napisan v takšni obliki, da lahko agent skozi opis preveri, če storitev ustreza iskalnim zahtevam (*matchmaking*). V *ServiceProfile* opisu so podane informacije o:

- Rezultatu storitve
- Omejitvah storitve in oceni njene uporabnosti (*quality of service*)
- Pogojih, ki jim mora uporabnik zadostiti za uspešno uporabo storitve

Za *ServiceProfile* opis ni posebnih pravil, kako ga narediti, je pa omogočeno z uporabo OWL podrazredov izdelati prav specializirane predstavitve storitev. OWL-s tukaj predlaga možnost predstavitve v obliki treh skupin:

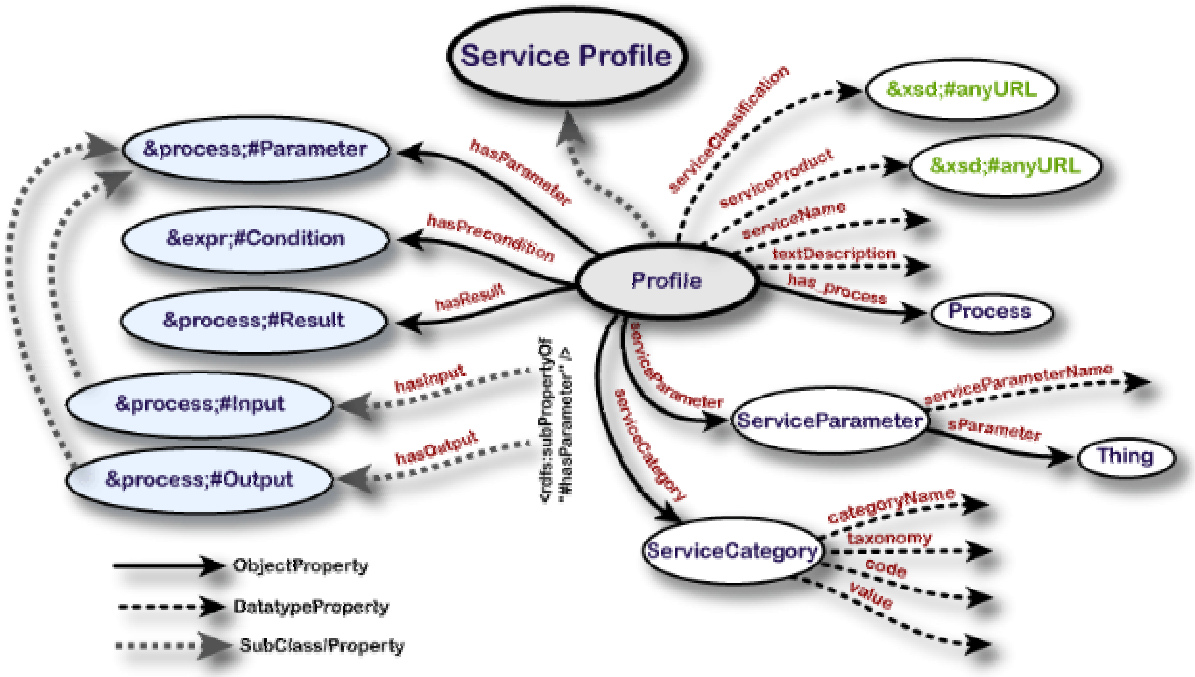
- Katera organizacija je ponudnik storitve – tukaj se lahko vpišejo kontaktne informacije ponudnika storitve ali osebe, ki bi nam lahko dala dodatne informacije o uporabi storitve. Ta podatek se pri avtomatični obdelavi storitve s strani agenta običajno izpusti, je pa lahko zelo uporaben v primeru neuspele avtomatične obdelave.

- Kakšno funkcijo storitev izvede – to je ključna informacija, ki jo ponuja *ServiceProfile*. Tu je opisana transformacija, ki jo storitev naredi ali bolj natančno kakšno obliko vhodnih podatkov zahteva in v kakšni obliki bodo izhodni podatki po obdelavi. Dodatno so lahko opisani morebitni predpogoji za uporabo in posledice, ki nastanejo ob uporabi storitve. Primer bi bila storitev, ki prodaja knjige. Kot predpogoj zahteva veljavno plačilno kartico, kot vhodna podatka pa naslov knjige in šifro kartice. Kot rezultat bi vrnila potrjeno naročilo knjige in datum dostave, posledica bi pa bila prenos denarja v višini zneska nakupa z računa kartice na račun ponudnika storitve. Ti štirje parametri se v literaturi, kadar se jih skupno obravnava, nahajajo pod kratico IOPE (5) (*input-output-precondition-effect*).
- Opis lastnosti, ki predstavljajo možnosti storitve - tu so informacije o kategoriji, kamor se storitev uvršča, kvaliteti storitve in parametrih storitve, ki se nahajajo v storitvi ali ki bi jih storitev dodatno želela specificirati kot referenčne. Tukaj bi izpostavil uporabo razredov *serviceParameter* in *ServiceCategory*.
  serviceParameter zato, ker se nanaša točno na IOPE storitve in vsebuje 2 lastnosti:

  - *serviceParameterName* opisuje en realni parameter v storitvi (ta je lahko navadna skupina znakov ali morda URI naslov, ki opisuje njegove lastnosti)
  - *sParameter* pa lahko kaže na parameter, ki je definiran v neki zunanji ontologiji. To pomeni, da če obstaja nekje na nekem določenem URI naslovu ontologija, ki ima v svojem opisu definiran koncept 'stalni naslov', lahko vsakdo, ki bo v svoji storitvi uporabljal koncept 'stalni naslov' naredi semantični sklic iz svojega parametra na ta določen URI. Podobno se pri XSD ponavadi sklicuje na URI naslov, kjer so vsi elementi XSD točno definirani (http://www.w3.org/2001/XMLSchema) in ti opisi so kot takšni dani na razpolago v uporabo ostalim, ki želijo v svojih WSDL datotekah z njimi opisati elemente.

  ServiceCategory je pa zanimiva iz stališča, ker omogoča klasifikacijo storitve. Podobno kot pri *serviceParameter* se tudi tukaj lahko naredi sklic na zunanjo ontologijo in se pove, da spletna storitev spada pod npr. kategorijo storitev gradbene informatike. Ta podatek bi še dodatno olajšal agentu analizo *ServiceProfile* storitve. Pod posameznimi *ServiceCategory* področji bodo razvijalci semantičnega spleta s časom ustvarili ontologije, ki bodo v pomoč izdelovalcem spletnih storitev pri poimenovanju parametrov

in definiranju položaja svoje storitve v ontologiji kategorije (npr. vhodni podatek storitve bosta ime in priimek zaposlenega, rezultat bo pa njegovo trenutno gradbišče). Tako bo prišlo do večjega števila semantičnih storitev, ki uporabljajo enako (mogoče s sklicevanjem celo isto!) ontologijo in taksonomijo. To ni pogoj za delovanje semantičnih spletnih storitev (rešitev različnih poimenovanj konceptov je definiranje pravil, ki lahko izenačijo konceptualno vrednost), bo pa omogočilo lažje in boljše delovanje agentov, ki bodo lahko izvajali kompozicije storitev in tako vračali odgovore na vse bolj kompleksna vprašanja uporabnikov.



Slika 5: Prikaz strukture OWL-s ServiceProfile

Vir: [http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/]

ServiceModel

ServiceModel (*describedBy*) opisuje 'Kako storitev deluje' ali pogled na interakcijo med uporabnikom in storitvijo kot na proces. Tukaj se nahaja natančnejši semantični opis zahtev, pogojev pod katerimi se bodo dosegli posamezni rezultati in kjer je potrebno tudi opis procesov

po korakih, ki bodo pripeljali do teh rezultatov. Ali drugače, opisuje kako sprožiti storitev in kaj se bo zgodilo, ko bo storitev aktivirana.
Proces ni program, ki bi ga lahko izvršili, ampak je specifikacija načinov, na katere lahko uporabnik uporablja storitev. Tukaj ločimo 2 vrsti procesov:

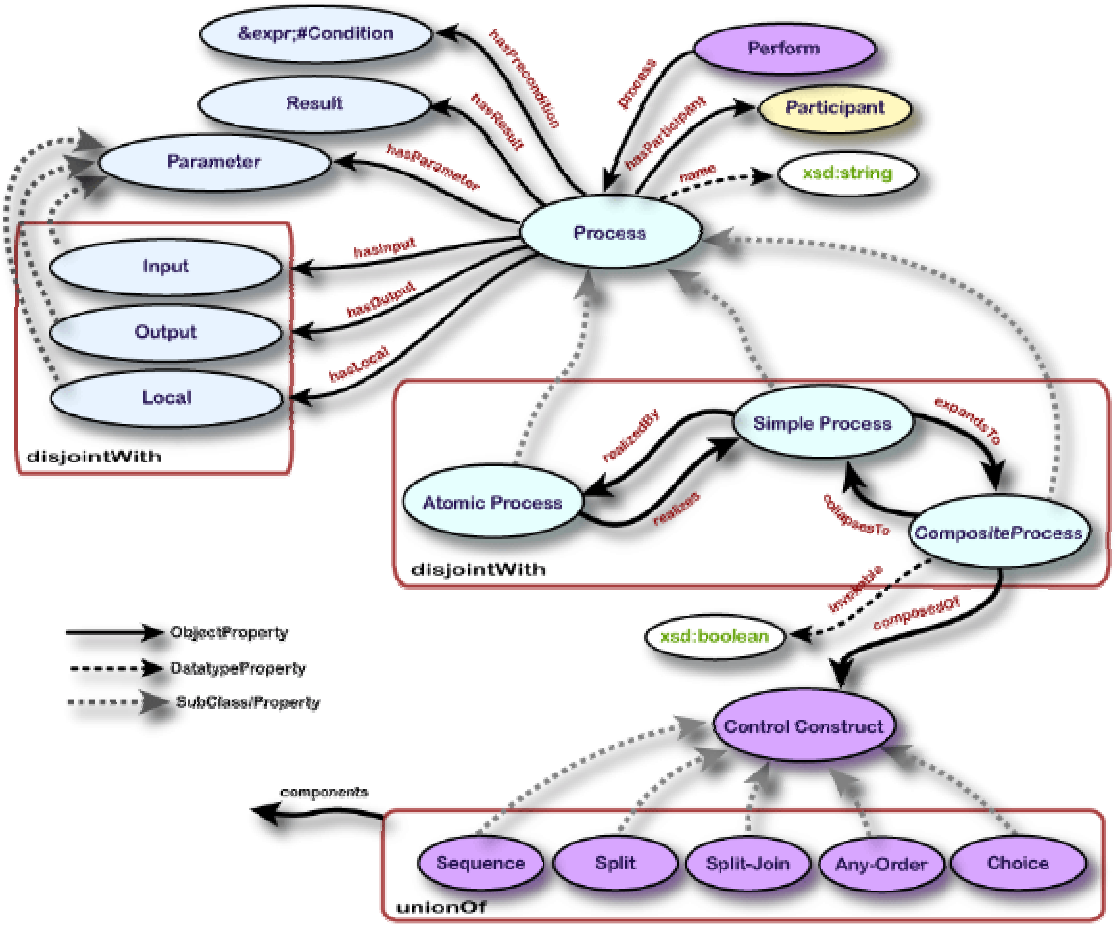
- osnovni (*atomic*) proces je opis storitve, ki pričakuje 1 sporočilo (lahko kompleksno) in ki kot rezultat vrne 1 sporočilo (lahko kompleksno)
- sestavljeni (*composite*) proces pa ohranja neko stanje med uporabnikom in storitvijo. Z vsakim naslednjim sporočilom, ki ga pošlje uporabnik ta napreduje skozi proces do končnega rezultata.

Preprosti (*simple*) proces, ki se tudi lahko pojavlja v *ServiceModel* opisu pa služi predstavitvi osnovnega procesa iz nekega drugega stališča, ali pa predstavlja sestavljeni proces v poenostavljeni obliki zaradi potreb planiranja ali predstavitev. V prvem primeru ima izvršitev preprostega procesa osnovni proces, v drugem se pa za izvršitev preprostegi proces razširi v sestavljeni proces.

Namen procesa je lahko vrnjena informacija, ki je nastala na osnovi vhodne informacije (produkcija informacije je opisana z vhodnimi in izhodnimi podatki), drugi namen pa je, da se naredi neka sprememba v stanju (ta je opisana s predpogoji in posledicami). Proces ima lahko katerokoli število vhodnih in izhodnih podatkov (tudi 0), prav tako tudi število predpogojev in posledic ni omejeno. Vsi predpogoji morajo biti izpolnjeni, da se lahko storitev izvrši do konca.

Za bolj kompleksne storitve, ki vsebujejo več korakov, lahko agenti *ServiceModel* opis uporabijo na 4 načine:

- za bolj poglobljeno analizo ustreznosti storitve
- da lahko kombinirajo več opisov storitev in tako opravijo določeno nalogo
- med potekom uporabe storitve lahko koordinirajo aktivnosti različnih udeležencev
- imajo možnost nadzorovanja izvajanja storitve



Slika 6: Prikaz strukture OWL-s ServiceModel ali Process

Vir: [http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/]

Grounding

ServiceGrounding (*supports*) opisuje 'Kako aktivirati storitev'. *Profile* in *Model* sta zamišljena kot abstraktni specifikaciji, ki ne opisujeta točno določenih oblik sporočil (*message formats*), protokolov in drugih za delovanje storitve specifičnih podatkov, ampak je temu namenjen *Grounding*. Njegova glavna funkcija je prikaz kako bodo (abstraktni) vhodni in izhodni podatki atomic procesa konkretno realizirani kot sporočila, ki nosijo te podatke v enem od specifičnih prenosnih formatov. *Grounding* tako med ostalim vsebuje konkretne detajle v zvezi s protokolom, sporočili (*message formats*), serializacijo, transportom sporočil in URI naslovi storitve.

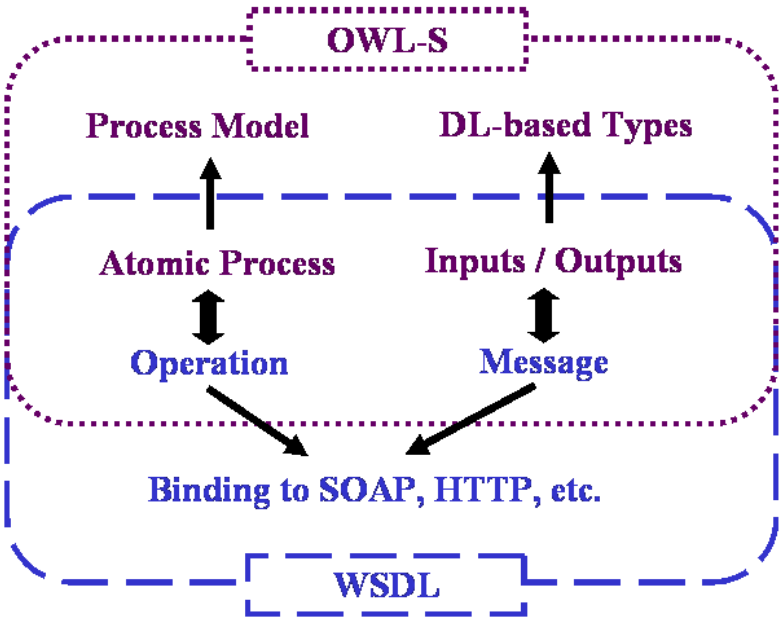
OWL-s in WSDL se prekrivata na področjih opisov abstraktnih informacij, ki obsegajo vhodne in izhodne podatke, ampak kot je že bilo omenjeno, WSDL samo z XSD opisom ne more zapisati semantičnih povezav kot jih lahko OWL z razredi ontologije [20]. Ima pa WSDL standard dobro zasnovano aktiviranje storitve in je hkrati že precej dobro uveljavljen v splošni uporabi, zato so se snovalci OWL-s jezika odločili, da bo OWL-s *Grounding* v splošnem dosleden konceptu WSDL *Binding*. Povezovanju OWL-s in WSDL jezika ustvarja komplementarno uporabo obeh jezikov in sicer kot uporabo OWL razredov kot abstraktnih tipov delov sporočil (*message part)*, ki so deklarirani v WSDL in z opiranjem na *binding* WSDL standarda. Ker ima OWL-s XML sintakso in ker se njegova deklaracija osnovnih procesov ter vhodnih in izhodnih sporočil že lepo ujema z WSDL, je razširitev WSDL *binding*-a (npr. SOAP *binding*) za uporabo z OWL-s precej enostavna. OWL-s metoda semantične anotacije spletne storitve skuša združiti najboljše iz obeh jezikov in tako zagotoviti vse potrebne podatke za agente in njihovo delovanje.

Bistvo povezovanja OWL-s/WSDL je zajeto v izdelavi enega *Grounding* primera, ki zajema vse potrebne informacije za uporabo storitve. In ker ima primer *Grounding* vse te informacije, modifikacija WSDL dokumenta zaradi združevanja z OWL-s ni potrebna, je pa koristno podati informacije o povezovanju tudi v samem WSDL opisu. Ustvarjalci WSDL so že pri izdelavi standarda predvidevali, da XSD tako v sedanjosti in še bolj izrazito v prihodnosti ne bo sposoben opisati vseh oblik podatkov v uporabi, zato so omogočili uporabo drugih opisnih jezikov preko razširitvenih možnosti, vgrajenih v WSDL datoteko. Tako se lahko povezanost WSDL z OWL-s naredi preko razširitev WSDL elementov:

- Types
- Message
- Operation
- Binding

V element *types* lahko vključimo poljubne OWL-s deklaracije, v elementu *message* pa lahko definiramo dele sporočila in njihovo povezavo z OWL razredi ontologije kot njihovimi abstraktnimi tipi. V elementu *operation* OWL-s ponuja nov atribut z namenom predstavitve povezave med dano operacijo in OWL-s osnovnim procesom. V elementu *binding* praktično ni nove razširitve, ampak določitev novega dekodiranja dokumenta (npr.

http://www.w3.org/2002/07/owl) za uporabo s SOAP *binding*-om, ki je definiran v WSDL. Z eno samo izjemo (predlagani novi atribut za operation element) celotna povezava OWL-s in WSDL poteka preko razširitvenih možnosti WSDL standarda.



Slika 7: Prikaz povezav med OWL-s in WSDL

Vir: [http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/]

Prikaz možnosti povezovanja z OWL-s koncepti preko razširitvenih možnosti WSDL 1.1 (povezave so krepko označene):

```
<wsdl:definitions name="nmtoken"? targetNamespace="uri"?
                  xmlns:owl-s-wsdl="http://www.daml.org/services/owl-s/wsdl/">
    <import namespace="uri" location="uri"/>*
    <wsdl:documentation .... /> ?

    <wsdl:types> ?
        <wsdl:documentation .... />?
        <xsd:schema .... />*
        <-- extensibility element --> *
        <rdf:RDF namespace-declarations ...> .... </rdf:RDF/>*
```

```
    </wsdl:types>

    <wsdl:message name="nmtoken"> *
        <wsdl:documentation .... />?
        <!-- This spec adds attribute owl-s-parameter -->
        <part name="nmtoken" element="qname"? type="qname"? owl-s-wsdl:owl-s-
parameter="qname"?/> *
    </wsdl:message>

    <wsdl:portType name="nmtoken">*
        <wsdl:documentation .... />?
        <!-- This spec proposes attribute owl-s-process -->
        <wsdl:operation name="nmtoken" owl-s-wsdl:owl-s-process="qname"?>*
           <wsdl:documentation .... /> ?
           <wsdl:input name="nmtoken"? message="qname">?
               <wsdl:documentation .... /> ?
           </wsdl:input>
           <wsdl:output name="nmtoken"? message="qname">?
               <wsdl:documentation .... /> ?
           </wsdl:output>
           <wsdl:fault name="nmtoken" message="qname"> *
               <wsdl:documentation .... /> ?
           </wsdl:fault>
        </wsdl:operation>
    </wsdl:portType>

    <wsdl:binding name="nmtoken" type="qname">*
        <wsdl:documentation .... />?
        <-- extensibility element --> *
        <wsdl:operation name="nmtoken">*
           <wsdl:documentation .... /> ?
           <-- extensibility element --> *
           <wsdl:input> ?
               <wsdl:documentation .... /> ?
               <-- extensibility element -->
           </wsdl:input>
           <wsdl:output> ?
               <wsdl:documentation .... /> ?
               <-- extensibility element --> *
           </wsdl:output>
           <wsdl:fault name="nmtoken"> *
               <wsdl:documentation .... /> ?
               <-- extensibility element --> *
           </wsdl:fault>
        </wsdl:operation>
```

```
    </wsdl:binding>

    <wsdl:service name="nmtoken"> *
        <wsdl:documentation .... />?
        <wsdl:port name="nmtoken" binding="qname"> *
           <wsdl:documentation .... /> ?
           <-- extensibility element -->
        </wsdl:port>
        <-- extensibility element -->
    </wsdl:service>

    <-- extensibility element --> *

</wsdl:definitions>
```

Zgoraj so opisani mehanizmi za povezavo OWL-s/WSDL preko WSDL opisa storitve. V OWL-s opisu se ta povezanost definira preko *WsdlGrounding* razreda, ki je podrazred *Grounding-a*. Vsaka instanca *WsdlGrounding* razreda vsebuje seznam *WsdlAtomicProcessGrounding* instanc, ki se navezujejo na WSDL elemente in sicer preko naslednjih lastnosti (*properties*):

- *wsdlVersion*: URI naslov ki kaže na verzijo WSDL dokumenta.
- *wsdlDocument*: URI naslov WSDL dokumenta, na katerega se navezuje ta *Grounding*.
- *wsdlOperation*: URI naslov WSDL operacije danega osnovnega procesa.
- *wsdlService*, *wsdlPort* (neobvezno): URI naslov WSDL storitve (ali vhoda storitve) ki ponuja operacijo. Vsebina *wsdlOperation* je lahko (ali pa tudi ne) unikatno določen WSDL vhod, s katerim se naredi povezava. Če je več vhodov, ki ponujajo dostop do operacije, lahko program za izdelavo OWL-s izbere katerega koli izmed tistih, ki so na voljo.
- *wsdlInputMessage*: predmet, ki vsebuje URI naslov od WSDL sporočila, ki prenaša vhodne podatke od obravnavanega osnovnega procesa..
- *wsdlInput*: predmet, ki vsebuje povezavo para in sicer OWL opisa vhodnih podatkov z deli vhodnega WSDL sporočila. Predvideno je, da obstaja en par za vsak vhodni podatek. Vsak tak par je definiran v instanci *WsdlInputMessageMap* predmeta. En element para (predstavljen v *wsdlMessagePart*) definira del sporočila z URI, drugi element para pa pokaže, kako dobiti *message part* iz enega ali več vhodnih podatkov OWL-S osnovnega procesa. V najbolj preprostem primeru se del sporočila navezuje samo na en OWL-S vhodni podatek, hkrati pa je sklic na ta podatek narejen direktno iz WSDL.
- *wsdlOutputMessage*: podobno kot *wsdlInputMessage*, ampak za izhodne podatke.

- *wsdlOutput*: podobno kot *wsdlInput*, ampak za izhodne podatke.

OWL-s ontologija storitve ima glede predlaganih 3 razredov 2 omejitvi:

- storitev je lahko opisana kvečjemu z enim *Model* razredom in
- OWL-s/WSDL kombinacija se navezuje na natanko eno samo storitev

Število razredov *ServiceProfile* in *ServiceModel* namenoma ni navzdol omejeno, čeprav storitev načeloma potrebuje vse tri opise za kompleten opis storitve. Namreč pojavijo se lahko primeri, kjer bi bil delni opis bolj ustrezna rešitev. Iz istega razloga ni zgornje omejitve števila opisov za *ServiceProfile* in Service*Grounding* (zelo uporabno bi bilo imeti na voljo več opisov in več možnosti aktiviranja storitve).
OWL-s tudi ne daje nobenih omejitev na povezanost *ServiceProfile* in *ServiceModel* razredov, čeprav opisujeta iste IOPE. Razlog za to je omogočanje primera, ko bi npr. ponudnik storitve omogočal nakup knjige. Za izvedbo storitve bi nudil iskanje knjige po njegovi bazi na osnovi nekih določenih vhodnih podatkov, izbiro knjige in nakup. Ker pa želi, da njegov iskalnik knjig uporabljajo samo potencialni kupci, se tako odloči, da bo oglaševal samo nakup knjig, ne pa tudi iskanja knjig. Razlog je lahko npr. preobremenitev strežnika z zahtevami. Problem, ki se pa lahko v primeru neusklajenosti *ServiceProfile* in *ServiceModel* razredov pojavi pa je, da *ServiceProfile* oglašuje drugačno storitev, kot jo ima v svojem opisu *ServiceModel*. In tako bi agent pričakoval čisto druge pogoje in vhodne podatke, kot pa so za pravilno uporabo storitve potrebni. Seveda proces ne bi stekel, storitev pa je za avtomatično obdelavo popolnoma neuporabna.

Zgoraj opisani pristop OWL-s s tremi razredi kot ponudniki semantičnih informacij za storitev je samo osnovni predlagan pristop k semantičnemu opisu storitve, ki je uporaben v večini primerov. V primeru specifične storitve ali specifičnih potreb je seveda model mogoče zastaviti tudi drugače.

4.2 WSDL-s

WSDL-s je predlog semantične anotacije WSDL opisov spletnih storitev, ki ga je pripravila skupina LSDIS Lab iz University of Georgia (http://lsdis.cs.uga.edu) v sodelovanju z računalniškim podjetjem IBM (http://www.ibm.com/).

4.2.1 Zasnova modela

WSDL-s je semantično anotiran WSDL 2.0 dokument spletne storitve ali način dodajanja semantičnega opisa v WSDL 2.0 dokument preko njegovih razširitvenih možnosti. Čeprav je WSDL 2.0 standard v tem trenutku še v fazi potrditve, je osnutek dokumenta, ki je bil objavljen marca 2004 s strani W3C organizacije kot predlog novega standarda osnova WSDL-s koncepta semantične anotacije spletne storitve.

Pristop temelji na opisu storitve kot prikazu operacij, ki jih storitev izvaja ter na njihovih vhodnih in izhodnih podatkih. Predlog LSDIS Lab-a za semantično anotacijo WSDL 1.1 dokumenta je zelo podoben predlogu, ki ga predlaga OWL-s koncept za boljšo predstavitev in razumevanje semantičnega opisa in povezav med OWL-s in WSDL (poglavje 4.1.2). Glede na to, da ima WSDL 1.1 standard določene razširitvene možnosti niti ni toliko presenetljiva podobnost predlogov. Vendar v OWL-s so semantične anotacije WSDL dokumenta dodatne informacije (ne potrebne), WSDL-s pa temelji ravno na razširitvenih možnostih WSDL dokumenta.

WSDL-s koncept semantične anotacije tako temelji na naslednjih pravilih [17] [16]:

- Koncept mora temeljiti na obstoječih standardih za spletne storitve in delovati kot nadgradnja obstoječega dela na tem področju, saj je v spletne storitve bilo do zdaj vloženih kar precej sredstev tako finančnih kot tudi delovnih ur ljudi. Zato je lahko nov koncept nesprejemljiv.
- Mehanizem semantične anotacije bi moral biti neodvisen od semantičnega opisnega jezika. Trenutno obstaja nekaj različic takšnih jezikov (OWL, WSMO, UML - Universal Markup Language). Vsak od teh jezikov ima svojo stopnjo možnosti semantičnega opisa in podporo razvijalcev, zato je potrebno zagotoviti možnost lastne izbire razvijalca ali organizacije, kateri jezik bodo uporabili.

- Mehanizem za semantično anotacijo mora omogočati anotacijo z več različni jeziki. To bi omogočalo ponudniku storitve da semantično anotira storitev za več različnih iskalnikov storitev.
- Podatkovni tipi v storitvi so v tem trenutku večinoma opisani z XML Schema. Takšen opis podatkov je že zelo uveljavljen, in čeprav WSDL 2.0 omogoča opis podatkovnih tipov z drugimi konstrukti npr. OWL razredi, ga je zaradi splošne uporabljenosti potrebno ohraniti kot osnovo za opis podatkovnega tipa. Tako se vrši nadgradnja opisov, ne pa uveljavlja nov način opisovanja storitev, kar je temeljna ideja WSDL-s.
- Omogočiti je potrebno povezovalni mehanizem med XSD podatkovnimi tipi in ontologijami. V primeru bi to pomenilo, da je osnovni model za storitev opisan z OWL, povezovanje WSDL XSD elementov pa bi lahko opravili z jezkiki, ki so usposobljeni za to npr. OWL, RDF/S, XSLT (31), XQuery (29)…

4.2.2 Semantična anotacija WSDL po konceptu WSDL-s

Predloga za semantično anotacijo WSDL 1.1 sta si toliko podobna, da predloga skupine LSIDS ne bom posebej predstavljal, nahaja se pa na spletni strani http://lsdis.cs.uga.edu/projects/meteor-s/ in služi samo kot prehodna stopnja do WSDL-s predloga.

WSDL-s temelji na standardu 2.0. Spet se uporabijo razširitvene možnosti standarda za navezovanje elementov na razrede v zunanji ontologiji. OWL-s je predstavil metodo semantične predstavitve storitve, ki lahko temelji samo na WSDL opisu, WSDL-s pa mora temeljiti na eni ali več ontologij.

Primer semantično anotiranih ali dodanih razširitvenih WSDL 2.0 elementov [16]:

Domena (URI naslov) na ontologijo storitve, ki vsebuje koncepte, na osnovi katerih deluje storitev

<definitions

xmlns:rosetta = http://webster.cs.uga.edu/azami/pips.owl# ……

…
>

<operation name = "checkStatus" pattern="mep:in-out" >

Action element je dopolnjen s konceptom v ontologiji, ki opisuje delovanje operacije storitve
<action **element = "rosetta:QueryOrderStatus" />**

Input in output elementa se sklicujeta na razrede ontologije
<input messageLabel = "statusQuery" **element = "rosetta:PurchaseOrderStatusQuery" />**
<output messageLabel = "status" **element = "rosetta:PurchaseOrderStatusResponse" />**

Naslednji tag 'Constraint' opisuje predpogoje in posledice operacije
<pre condition="PurchaseOrderStatusQuery.orderStatusDoc.?PurchaseOrder !=null"
/>
</operation>

WSDL-s kot svoje prednosti pred OWL-s omenja naslednje [35] :

- Razvijalci in ponudniki storitev lahko semantično anotirajo storitev skozi nadgradnjo, saj delo z WSDL že poznajo in ne s popolnoma konceptom, kot ga ponuja OWL-s
- Lahko sami izberejo jezik semantičnega opisa
- Relativno enostavna bi bila nadgradnja obstoječih programov za izdelavo opisov spletnih storitev
- Omenjajo tudi podvajanje nekaterih podatkov WSDL opisa skozi *ServiceProfile* OWL-s
- OWL-s predvideva, da za semantični opis vsi uporabljajo OWL jezik

4.3 WSMO

WSMO (*Web Service Modeling Ontology*) je tretja izmed največkrat omenjenih tehnologij semantičnih spletnih storitev. Temelji na WSMF(24) (*Web Service Management Framework*) [39], kjer WSMF predstavlja celovit pristop k spletnim storitvam, WSMO pa ga nadgrajuje kot ontologija za opis elementov, ki skupaj ustvarjajo semantični opis spletne storitve. WSMO je zapisan z WSML (25) (*Web Service Modeling Language*) jezikom, dodatno pa je razvit WSMX (27) (*Web Service Execution Environment*), ki ustvarja okolje za izvršitev ukazov in ostalo delo s semantičnimi spletnimi storitvami na WSMO.
Tehnologija je predlagana s strani Digital Enterprise Research Institute – DERI (http://www.deri.org) , ki je med vodilnimi evropskimi inštituti na področju semantičnih spletnih stroritev.

4.3.1 Zasnova modela

Koncepti in ideje, na katerih je zasnovan WSMO [37]:

- URI - je osnovni koncept interneta in je tudi tukaj osnova za unikatno identifikacijo virov na internetu. Dodatno so za WSMO na tem mestu sprejeti še nekateri drugi W3C standardi za lokacijsko pozicioniranje npr. domene (*namespace*)
- Ontologija - predstavlja uveljavljen model predstavitve znanja (*knowledge representation*) in se vse bolj uveljavlja kot eden osnovnih konceptov semantičnega spleta. Vsi opisi virov in izmenjanih podatkov tako v WSMO temeljijo na ontologijah.
- Striktno razdruževanje virov – vsi WSMO viri so definirani neodvisno in prostorsko porazdeljeno od ostalih, kar je eden izmed temeljev interneta
- Centralno posredovanje (*mediation*) – striktnemu razdruževanju virov sledi heterogenost izrazoslovja, temeljnih ontologij, protokolov in procesov, ki nastanejo v tako odprtih sistemih. To heteregonest bi reševali s pomočjo posrednikov (*mediators*), ki so eden temeljnih elementov WSMO.
- Ločevanje na osnovi vloge do spletne storitve v sistemu – navadno ima uporabnik neko željo, ki naj bi jo izpolnile storitve, na drugi strani so pa storitve, ki so na voljo. Kot primer je tukaj uporabnik, ki bi želel rezervirati počitniško sobo glede na vreme in ugodnosti za otroke, na drugi strani pa storitve navadno pokrivajo polete in zasedenost prenočišč.

- Opis storitve in njena implementacija – WSMO ločuje ta koncepta, podobno kot sta *ServiceProfile* in *ServiceGrounding* v OWL-s

Vsi elementi WSMO ontologije so definirani z meta-meta- modelirnim jezikom, ki temelji na *Meta Object Facility* (MOF). Z namenom omogočanja kompletnega opisa elementa je vsak WSMO element opisan z brezfunkcijskimi lastnostmi (non-functional properties). Te temeljijo na metapodatkih, ki jih vsebuje *Dublin Core (*DC*) Metadata Set.*

Na osnovi zgoraj opisanih konceptov je WSMO sestavljen iz 4-ih glavnih elementov, ki tvorijo opis semantičnih spletnih storitev [37] [39]:

- ontologija (*Ontology*) je ključni element WSMO, ki vsebuje lokalno terminologijo za opis ostalih elementov. Prva naloga je definiranje pomena informacij, druga pa je povezovanje človeške in računalniške terminologije. To je doseženo z opisi konceptov, relacij, funkcij, aksiomov, instanc konceptov in relacij po principih, ki so značilni za koncept ontologije kot modela zapisa vseh teh podatkov. Hkrati pa element ontologija vsebuje še nekatere dodatne informacije za opis posameznih elementov (*non-functional properties*), vnešene zunanje ontologije in uporabljene posrednike (*mediators*), ki imajo funkcijo reševanja nesoglasij v terminologijah. WSMO za večino *non-functional properties* lastnosti uporablja koncepte Dublin Core Metadata Element Set (http://dublincore.org/), ki deluje podobno kot deklarirani tipi XSD za WSDL opis.

```
Class ontology
      hasNonFunctionalProperties type nonFunctionalProperties
      importsOntology type ontology
      usesMediator type ooMediator
      hasConcept type concept
      hasRelation type relation
      hasFunction type function
      hasInstance type instance
      hasAxiom type axiom
```

- cilji (*Goals*) – predstavlja perspektivo uporabnika ob uporabljanju storitve in operacij, ki bi jih morala zagotavljati storitev, da bi zadovoljila njegove potrebe. Tako bi bil

primer storitve rezervacije hotela, kjer uporabnika zanimajo dodatne ugodnosti glede varstva otrok, storitev pa navadno deluje samo na osnovi prostih kapacitet. Cilji so nasprotje od elementa Web Service, ki predstavlja pogled s strani spletne storitve (predpogoji, možnosti…). Element *Goals* vsebuje *non-functional properties* lastnosti, dodane zunanje ontologije, uporabljene posrednike, čisto specifična pa sta *requestsCapability*, ki opisuje zahtevano funkcionalnost od storitve in *requestsInterface*, ki opisuje uporabnikove želje za način interakcije s storitvijo.

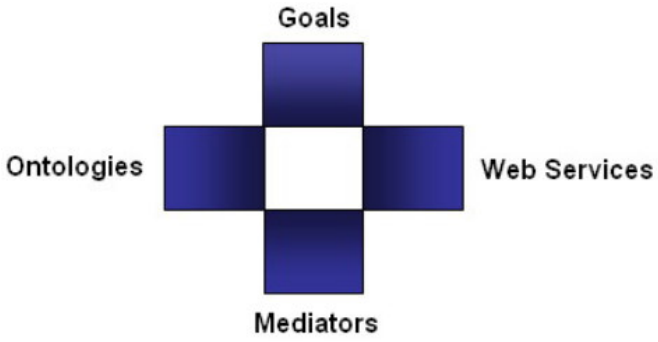
```
Class goal
      hasNonFunctionalProperties type nonFunctionalProperties
      importsOntology type ontology
      usesMediator type {ooMediator, ggMediator}
      requestsCapability type capability multiplicity = single-valued
      requestsInterface type interface
```

- spletne storitve (*Web services*) - element ponuja opis storitve preko *non-functional properties* lastnosti, dodanih zunanjih ontologij, uporabljenih posrednikov, zmožnosti storitve in vmesnika za uporabo. V zmožnostih storitve so zajeti predpogoji, posledice, predpostavke in stanje po uporabi, za vsako storitev pa obstaja natanko en takšen opis. Zmožnosti so lahko z posredniki povezane z določenimi cilji, kar je lahko uporabna informacija v fazi iskanja ustrezne storitve. Vmesnik vsebuje pomembne informacije za uporabo in sicer *'choreograpy'*, ki opisuje kako vzpostaviti povezavo s storitvijo s strani uporabnika in *'orchestration'*, ki opisuje če storitev uporablja druge storitve za svoje delovanje in če jih, na kakšen način. Ta opisa sta namenjena tudi kompozicijam procesov iz več različnih storitev.

```
Class service
      hasNonFunctionalProperties type nonFunctionalProperties
      importsOntology type ontology
      usesMediator type {ooMediator, wwMediator}
      hasCapability type capability multiplicity = single-valued
      hasInterface type interface
```

- posredniki (*Mediators*) – so temeljni element za reševanje problemov interoperabilnosti med različnimi WSMO elementi, ki so lahko heterogeni. Deluje na dveh nivojih in sicer na nivoju podatkov, kjer prihaja do različnih terminologij in nivoju procesa, kjer je potrebno posredovanje ob komunikaciji med različnimi storitvami. Uporabljajo se 4 vrste posrednikov:
  - *OOMediators* – ob vnosu ontologije ta posrednik rešuje probleme razlik v terminologijah
  - *GGMediators* – z njimi lahko povezujemo cilje med seboj
  - *WGMediators* – povezujejo element spletne storitve s cilji in tako sporočijo, da storitev delno ali v celoti izpolnjuje povezani cilj, ali pa da mora biti za uporabo vseh zmožnosti storitve razrešen omenjeni cilj.
  - *WWMediators* – povezujejo heterogene spletne storitve med seboj z namenom omogočanja interoperabilnosti

```
Class mediator
      hasNonFunctionalProperties type nonFunctionalProperties
      importsOntology type ontology
      hasSource type {ontology, goal, service, mediator}
      hasTarget type {ontology, goal, service, mediator}
      hasMediationService type {goal, service, wwMediator}
```



Slika 8: Glavni elementi WSMO ontologije

Vir: [http://www.wsmo.org]

Pomembno je tukaj omeniti da iz razlogov [41]:

- WSMO ne temelji na standardih OWL ali RDF, ki ju je W3C organizacija označila kot temeljna standarda semantičnega spleta

- WSMO 'choreography', ki opisuje način vzpostavljanja povezave s storitvijo je precej oddaljena od predloga W3C predloga WS-CDL (Web Services Choreography Description Language)
- Koncept WSMO nima povezave z nižjimi opisnimi jeziki kot je npr. WSDL

razvijalci WSMO trenutno nimajo podpore W3C organizacije za uveljavljanje WSMO kot W3C standarda.

4.4 Primerjava modelov semantičnih opisov spletnih storitev

OWL-s zagovarja idejo *'Complete do not compete'* in s tem ustvarjanje semantične plasti za opis in uporabo vseh standardov, ki so se že od samega začetka uveljavili kot dobri npr. WSDL *binding*. Za stare razvijalce spletnih storitev je koncept popolnoma nov in zahteva nekaj dodatnega izobraževanja za osvojitev metode. Hkrati je najstarejši in ima največjo podporo, tako s strani W3C kot s strani razvijalcev in se bo, če ne bo večjih sprememb na tem področju, verjetno uveljavil kot primarni standard za semantične spletne storitve. OWL-s je pravzaprav OWL zapis in je kot tak tudi komplatibilen s priporočili W3C, česar npr. WSMO ne izpolnjuje.

WSDL-s je najmlajši in zanimivo, temelji na konceptu, ki ga trenutno najbolj razvita metoda (OWL-s) vsebuje kot neobveznega. Glavno vodilo metode je, da se izkoristijo možnosti WSDL razširitev in se tako izogne prevelikemu dodatnemu izobraževanju razvijalcev, saj je trenutno podpora spletnim storitvam v precejšnjem vzponu. Preveliko spreminjanje izdelave spletnih storitev bi lahko povzročilo zaton ene bolj obetavnih spletnih tehnologij. Ideja je za osvojitev relativno preprosta, sem pa imel precej težav s programi za anotacijo, kjer ni vse delovalo, kot je bilo opisano v navodilih. Na tem mestu so se programi za OWL-s precej bolje izkazali.

WSMO je zastavil v celoti novo okolje za semantične spletne storitve, zato bi bil po mojem mnenju prehod iz osnovnih spletnih storitev na WSMO koncept precej težji kot za druga dva primera. Dejstvo, da ne temelji na OWL ali RDF, kar je prvo priporočilo s strani W3C kadar je govor o semantičnem spletu in hkrati da je WSDL zelo malo prisoten, bo verjetno odločilnega pomena, saj vse več na novo sprejetih tehnologij temelji na XML. In če se spletne

storitve, pa naj bodo običajne ali semantične, uvajajo zaradi interoperabilnosti, potem je zaradi splošne uveljavljenosti XML na tem področju njegova uporaba že skoraj da obvezna.

Na osnovi zgoraj navedenih primerjav in zaključkov sem se odločil, da praktični primer semantične spletne storitve izdelam na osnovi konceptov OWL-s in WSDL-s, ki temeljita na XML tehnologijah, imata podporo W3C organizacije ter precej dobro programsko opremo, ki je ključnega pomena za izvedbo semantičnega opisa storitve. Delno ju lahko obravnavamo tudi kot komplementarna koncepta, saj je priporočilo OWL-s za semantično anotacijo samega WSDL dokumenta skoraj identično predlogu semantične anotacije, ki ga predlagajo razvijalci WSDL-s za WSDL 1.1. Precej verjetno bo tudi predlog OWL-s za anotacijo WSDL 2.0 dokumenta zelo podoben predlogu WSDL-s.

5 PRAKTIČNI PRIMER

Zaradi kompleksnosti in potrebne stopnje interakcije je za gradbeno področje razvitih zelo malo pravih (XML podprtih) spletnih storitev. Hkrati veliko razvijalcev programske opreme za gradbeno stroko uporablja svoje koncepte in oznake in tako je prišlo do situacije, ko je velika večina obstoječih programov samostojnih, brez osnove za interakcijo z drugimi programi. Temu je verjetno v veliki meri pomagalo pomanjkanje uveljavljenih in splošno sprejetih standardov za aplikacije na področju gradbeništva, se pa trenutno na tem področju precej intenzivno dela. Spletne storitve, ki ne temeljijo na WSDL ali nimajo vhodnih in izhodnih podatkov narejenih na osnovi standardnih shem (npr. XSD), niso primerne za prikaz postopkov semantičnih anotacij. Zato sem se odločil narediti spletno storitev na osnovi Študentskega informacijskega sistema (17) (ŠIS) Fakultete za gradbeništvo in geodezijo v Ljubljani, ki ga zelo dobro poznam, saj na njegovem razvoju delamo zadnja 3 leta in ki ima določene podobnosti s sistemi, ki se v gradbeništvu uporabljajo npr. za vodenje evidenc o zaposlenih.

5.1 Spletna storitev SisFggPredmet

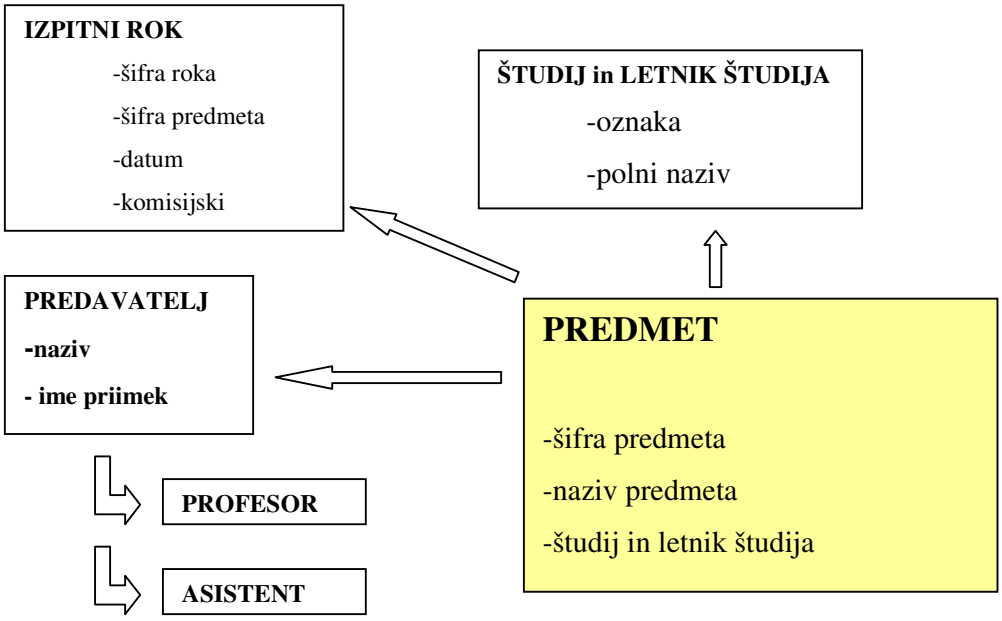
5.1.1 Opis informacijskega sistema ŠIS

ŠIS je spletni informacijski sistem za študente Fakultete za gradbeništvo in geodzijo Univerze v Ljubljani, ki stoji na Apache/PHP (12)/mySQL (7) sistemu in operacijskem sistemu Linux SuSe. Poleg objav aktualnih dogodkov vsebuje tudi celoten predmetnik študijev, ki jih fakulteta izvaja. Predmetnik je sestavljen iz 5-ih študijev, ki obsegajo 18 letnikov in imajo skupaj približno 270 predmetov. Ravno na koncept predmeta sem se osredotočil pri izdelavi storitve, bolj detajlno pa je predstavljen v naslednjem poglavju Ontologija predmeta.

5.1.2 Ontologija predmeta

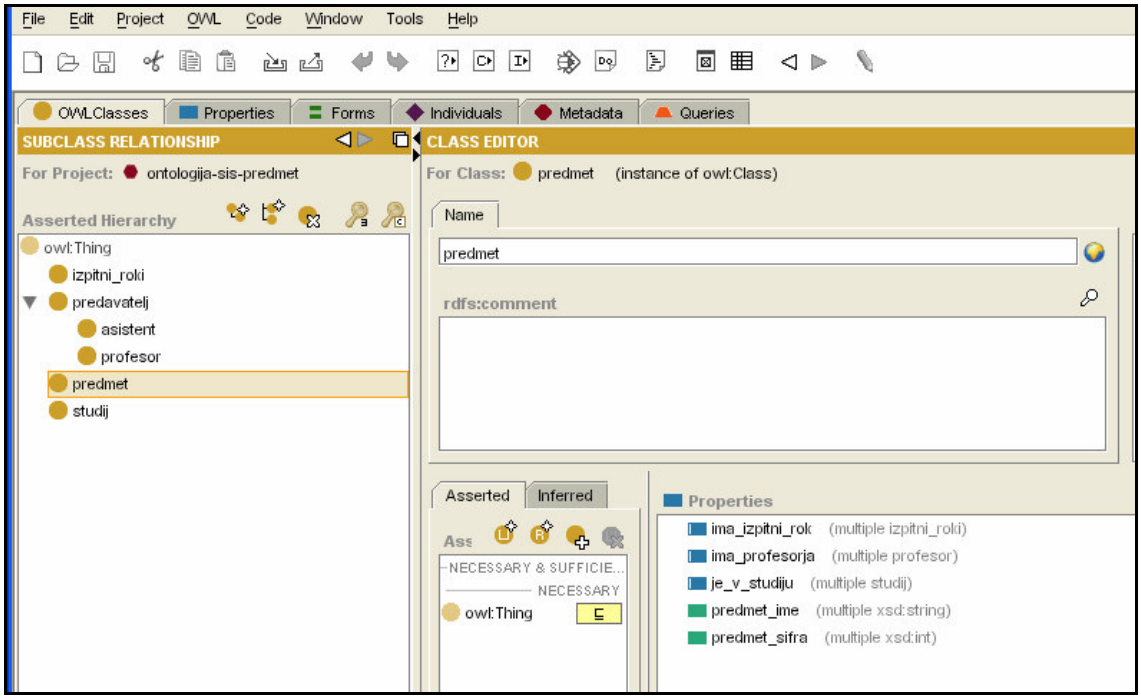
Vsak predmet v ŠIS-u ima svojo šifro, naziv predmeta, podatke o predavateljih ter študij in letnik, kamor spada. Posamezni predmeti imajo poleg podatkov v bazi še svojo mapo z datotekami, ki vsebujejo dodatne informacije o predmetu in študijsko gradivo za pomoč pri učenju. Predmet je na študij povezan z oznako, ki pove h kateremu študiju in v kateri letnik predmet spada (npr. gradbeništvo univerzitetni študij 1.letnik ima oznako gruni1). Predavatelja

sta v ŠIS-u podana kot podatka za posamezni predmet. Odločil sem se, da ju v ontologiji zaradi preglednosti postavim v poseben razred. Na predmete se dodatno navezujejo še izpitni roki, kjer ima vsak izpitni rok svojo šifro, predmet, za katerega je razpisan, datum opravljanja in podatek, če je rok komisijski ali ne. Spletno storitev, ki sem jo poimenoval SisFggPredmet, sem tako zasnoval okrog koncepta predmeta, osredotočil pa sem se na informacije, ki se za posamezne predmete nahajajo v bazi.



Slika 9: Prikaz koncepta *predmet* v študentskem informacijskem sistemu

OWL zapis ontologije sem naredil s programom Protege (http://protege.stanford.edu/) in dodatnim modulom OWLClasses [20]. Najprej sem definiral vse razrede in podrazrede, nato pa sem jih povezal z lastnostmi (properties) npr. profesor *predava* predmet. Program hkrati omogoča definiranje tudi inverznih lastnosti npr. predmet *ima profesorja* profesor.

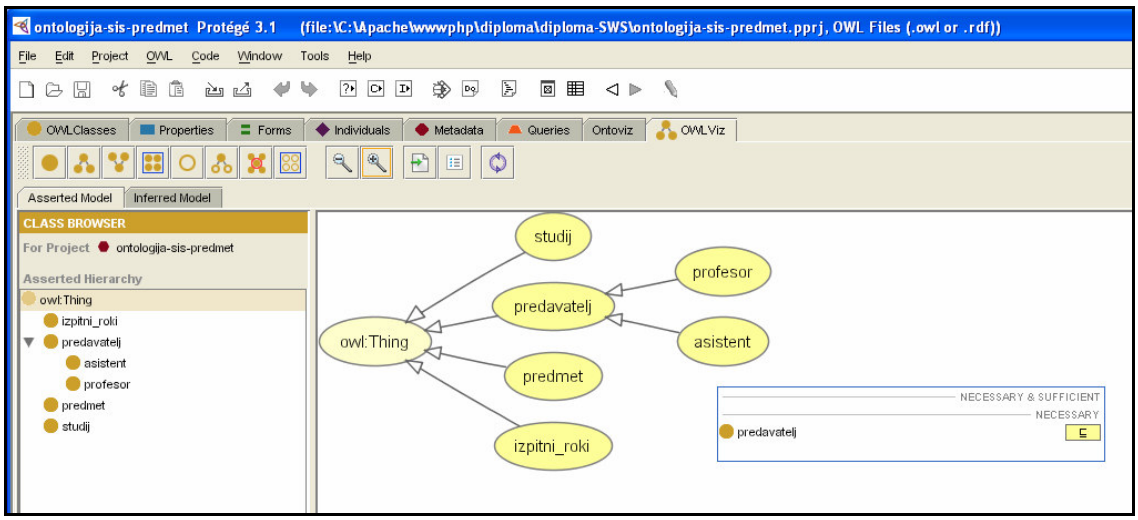


Slika 10: Prikaz izdelave konceptov ontologije in njihovih lastnosti v Protege-ju

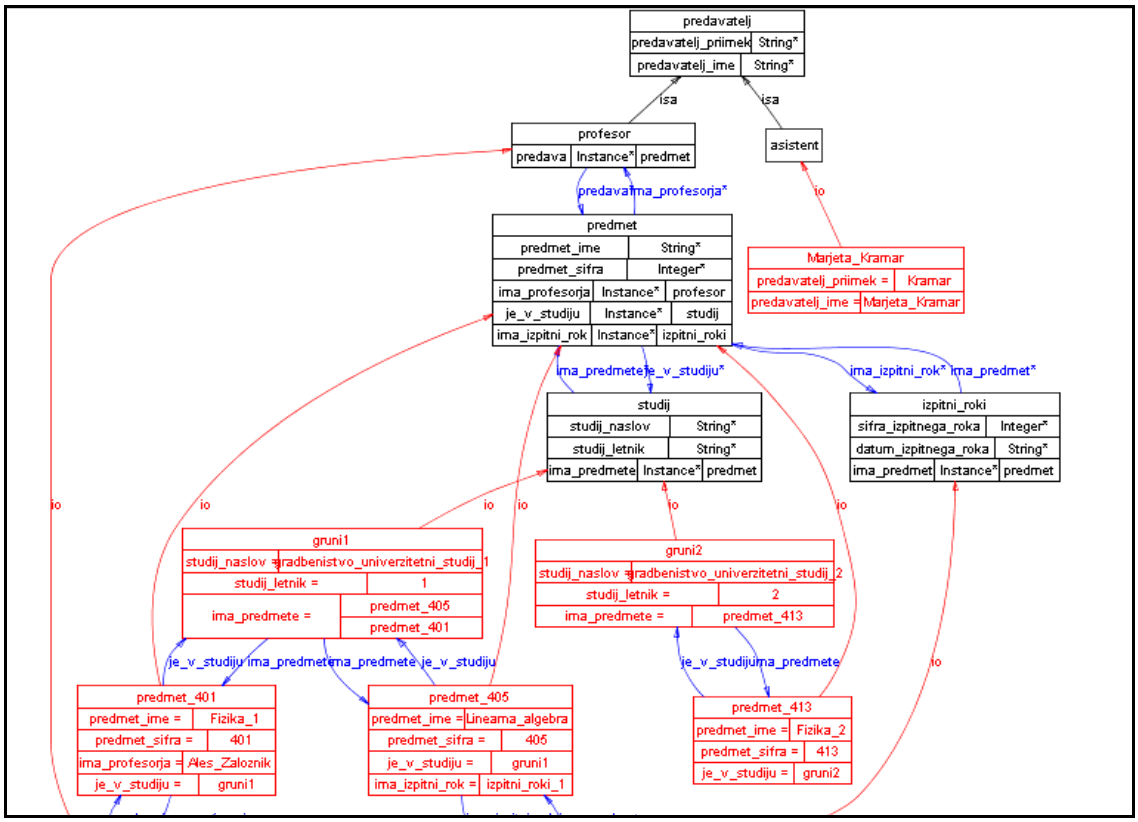
Protege omogoča tudi vnos instanc v OWL zapis. Dodal sem nekaj teh podatkov, ampak bolj za primer in za kontrolo pravilnosti zapisa ontologije s poizvedbami po OWL datoteki, ki so prav tako mogoče v tem programu. OWL zapis bi lahko vseboval poleg zapisa konceptov (*taxonomy*) in njihovih lastnosti (*inference rules*) tudi vse podatke, ki bi bili kot instance zapisani zraven. Kot takšen bi lahko datoteka, v kateri bi bili vsi ti podatki shranjeni, služila in kot ontologija sistema in kot njegova baza. Vendar se čas trajanja dela z velikostjo OWL datoteke progresivno veča. OWL datoteke so obvladljive za obdelavo do velikosti nekaj sto MByte-ov, nato so časi obdelave in iskanja predolgi za normalno delo. V prihodnosti se bo zaradi te težave verjetno uveljavil postopek povezovanja OWL zapisa konceptov in povezav na storitev, ki bodo služili za predstavitev storitev agentom, podatke bo pa storitev iskala in izpisovala iz baze, katere delovanje je glede na OWL zapis ontologije neprimerno hitrejše.
Protege s svojimi moduli ponuja tudi dobre grafične prikaze ontologij. V nadaljevanju sta dva takšna prikaza, OWL zapis ontologije je pa priložen pod prilogo B.

Na tem mestu bi pohvalil program s svojimi razširitvenimi moduli. Protege je za potrebe moje diplomske naloge tekel brezhibno, uporabniški vmesnik za izdelavo ontologij je zelo dober, rekel

bi celo odličen. Pri izdelavi in razumevanju precej kompliciranega stvari kot je ontologija v računalniških sistemih je program takšnega nivoja zelo zaželjen.



Slika 11: Prikaz strukture razredov z modulom OWL Viz



Slika 12: Prikaz strukture razredov in njihovih povezav ter vnesenih instanc z modulom Onto Viz

5.1.3 Opis storitve

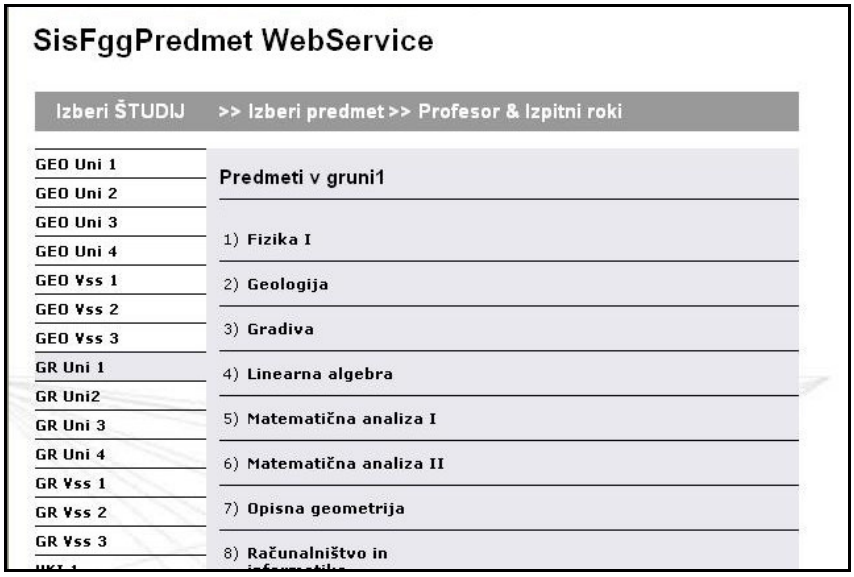
Storitev omogoča izpis predmetov po posameznih študijih in nato za izbrani predmet še izpis podatkov o profesorju in izpitnih rokih, ki so za ta predmet razpisani. Tako sem storitev razdelil na 4 operacije:

- izpisi_predmete_operacija, ki vzame za vhodni podatek oznako študija in vrne seznam predmetov, ki so v izbranem letniku tega študija
- izberi_predmet_operacija, ki ima za nalogo samo to, da na osnovi vhodne šifre predmeta naprej upošteva, da bo uporabnik nadaljnje operacije izvajal za ta predmet
- izpitni_roki_predmeta_operacija vrne razpisane izpitne roke za predmet, ki je določen na osnovi podane šifre predmeta
- profesor_predmeta_operacija na osnovi podane šifre predmeta vrne podatek o profesorju, ki ta predmet predava

Za lažji prikaz delovanja storitve sem poleg storitve izdelal še dinamično spletno stran kot vmesnik, ki sproža spletno storitev s pomočjo PHP skripte in NuSOAP PHP razreda. NuSOAP [13] z vključitvijo v PHP aplikacije omogoča enostavno uporabo WSDL dokumentov in SOAP preko HTTP protokola s PHP jezikom. Na tem razredu temelji tudi delovanje same spletne storitve na strani ponudnika. Vmesnik s hyperlink povezavami na spletni strani določa vhodne podatke in zahteve za storitev, temu pa sledi izpis vrnjenih rezultatov spet v HTML obliki. Omogoča uporabo vseh operacij in sicer po korakih od začetne izbire študija do izpisa iskanega profesorja ali izpitnih rokov posameznega predmeta.Vmesnik se nahaja na naslovu http://www.student-info.net/diploma/SWS-client.php.

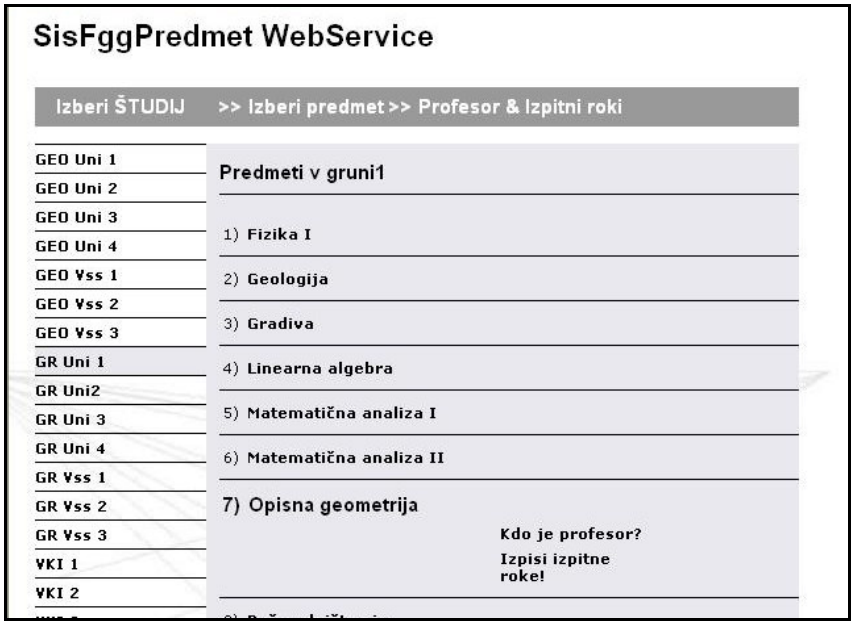
Slika 13: Prikaz začetnega stanja vmesnika, kjer je potrebno izbrati študij in letnik študija.

Operacija 'izpisi_predmete_operacija' je primarna operacija storitve. Na osnovi oznake študija in letnika (npr. gradbeništvo univerzitetni študij 1.letnik ima oznako 'gruni1') storitev preko te operacije pokliče PHP funkcijo, ki se priključi na mySQL bazo in iz baze s standardnimi SQL (16) stavki pridobi potrebne podatke o predmetih študija. Storitev te podatke vrne v obliki Array, ki ima dve polji. V prvem polju se nahaja naziv predmeta (npr. Opisna geometrija), v drugem pa unikatna šifra tega predmeta v ŠIS (v primeru Opisne geometrije bi bila ta 408). V vmesniku se rezultat te operacije vidi kot izpis predmetov, ki se nahajajo v tem letniku izbranega študija. Tukaj uporabnik izbere predmet z operacijo 'izberi_predmet_operacija', vmesnik pa bo to operacijo pozval na osnovi klika na povezavo posameznega predmeta. Obe nadaljnji operaciji iskanja profesorjev in izpitnih rokov se izvajata na osnovi šifre predmeta, zato je do teh informacij najlažje priti preko operacije izpisa predmetov, saj storitev uporabniku prikaže nazive predmetov, po katerih jih ta pozna, nadaljnje iskanje pa mu omogoči z uporabo šifre izbranega predmeta, ki je vpisana v povezavo predmeta in tako določa vhodni podatek za nadaljnje zahteve.



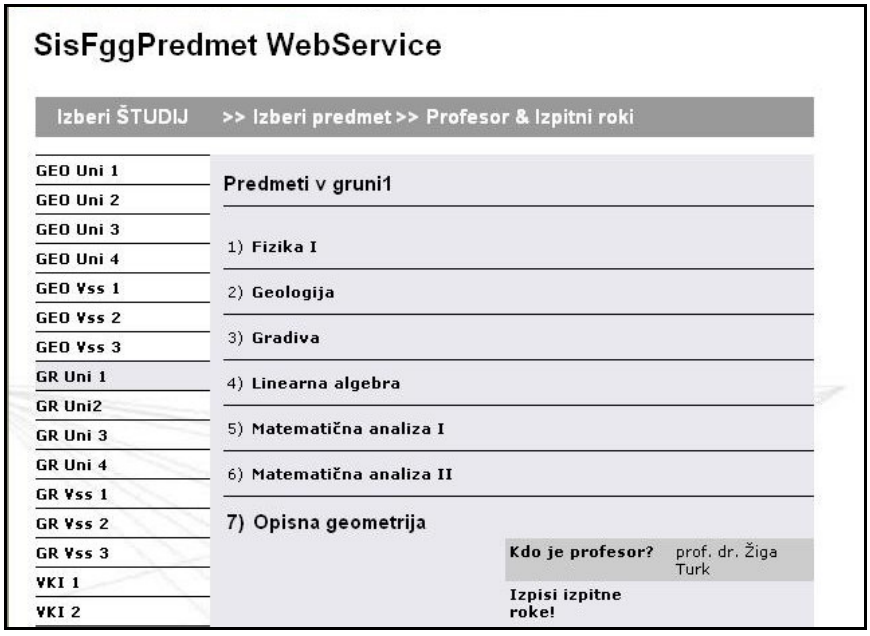
Slika 14: Izpis predmetov na vmesniku pri izbranem študiju gradbeništva za univerzitetni študij 1. letnika.

Po kliku na izbrani predmet lahko vmesnik predstavi stanje pred izbiro ene izmed obeh končnih operacij: iskanja profesorja tega predmeta (profesor_predmeta_operacija) ali izpisa izpitnih rokov, ki so bili razpisani za ta predmet (izpitni_roki_predmeta_operacija). To kaže slika, kjer se vidi možnost izbire naslednje operacije z dvema povezavama.



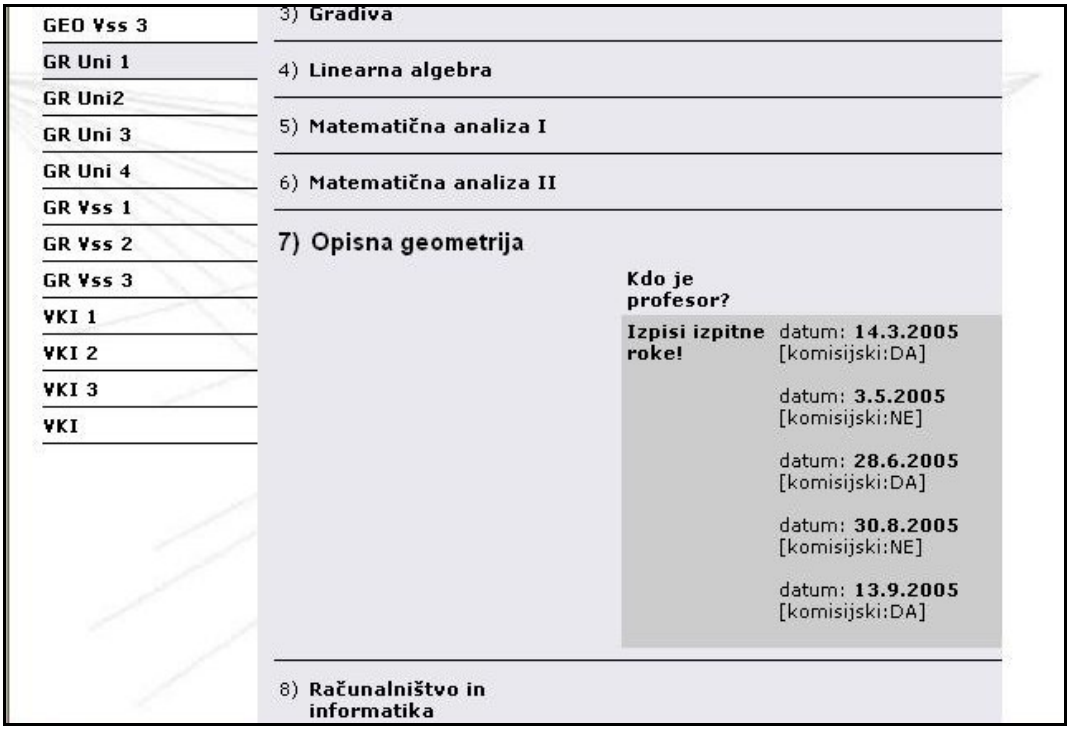
Slika 15: Vmesnik kaže izbrani predmet in nadaljne možnosti storitve

V tej točki sledi izbira zahteve po končnih informacijah, ki jih storitev ponuja, obe operaciji pa kot vhodni podatek sprejemata šifro predmeta. Preko vmesnika se lahko z izbiro povezave 'Kdo je profesor?' pokliče operacijo 'profesor_predmeta_operacija', če bi pa želeli operacijo iskanja profesorja klicati direktno iz svoje aplikacije, bi morali šifro izbranega predmeta vpisati v funkcijo, ki bi klicala storitev. Če v bazi ta podatek obstaja, je rezultat te operacije vrnjen podatek z nazivom in imenom profesorja. Na naslednji sliki je prikazan rezultat takšne zahteve.



Slika 16: Vmesnik kaže odgovor operacije izpis profesorja izbranega predmeta

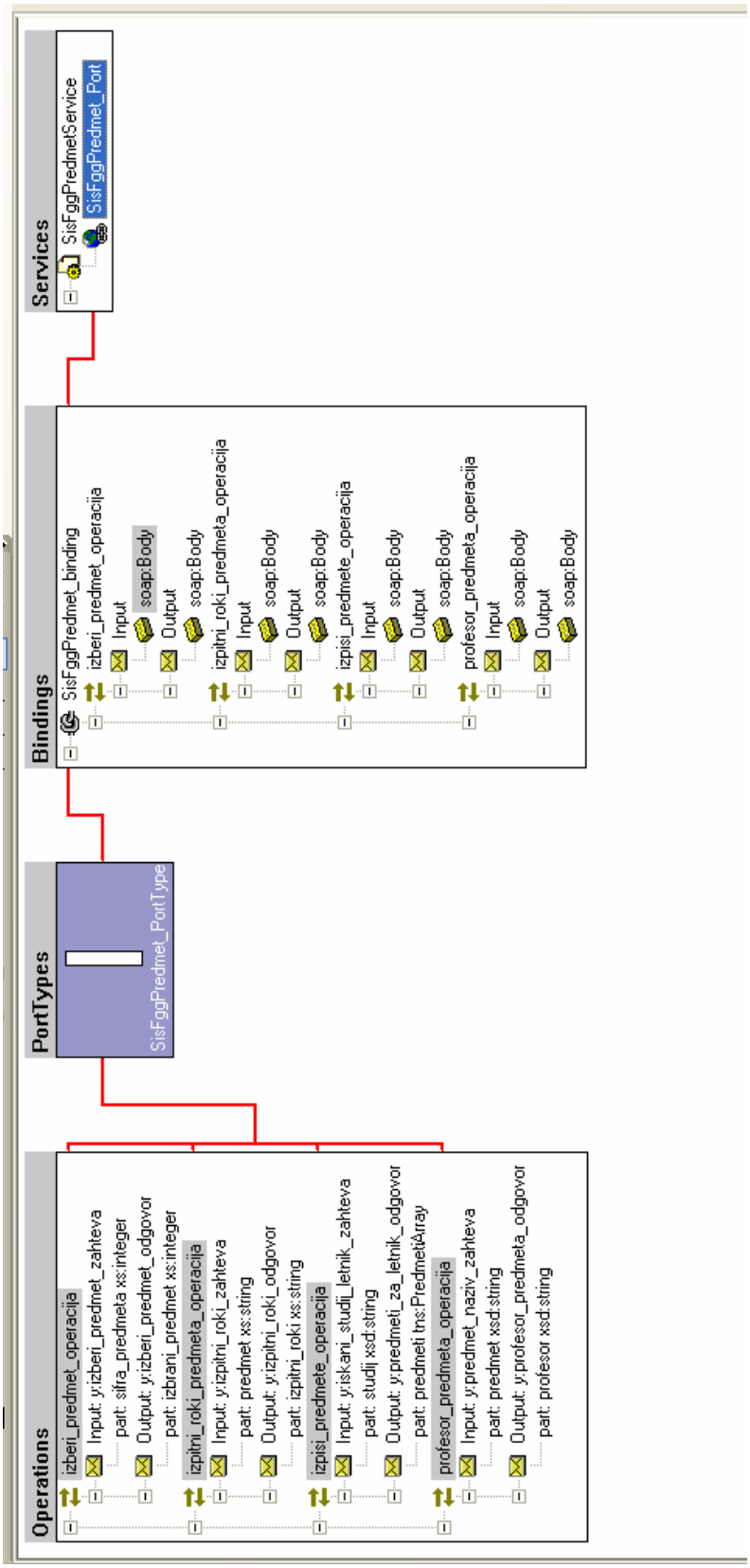
Druga možnost je izbira operacije 'izpitni_roki_predmeta_operacija', ki bo vrnila podatke o razpisanih izpitnih rokih. Operacija tako kot ostali dve pokliče funkcijo, ki po priključitvi na bazo sistema poišče podatke o izpitnih rokih izključno za ta predmet in jih pošlje skozi odgovor k uporabniku storitve. Povezava 'Izpisi izpitne roke' preko vmesnika kliče to operacijo. Rezultat je podatkovnega tipa string, prikazan pa je na naslednji sliki.



Slika 17: Izpis izpitnih rokov za izbrani predmet

Opis WSDL dokumenta

Vse 4 operacije tečejo preko istega vhoda storitve (SisFggPredmet_PortType), *binding* storitve pa poteka preko SOAP HTTP protokola (SisFggPredmet_binding). Razen enega so vsi vhodni podatki podatkovnega tipa XSD string ali integer. Operacija izberi_predmet_operacija ima tako vhodni kot izhodni podatek xsd:integer, vse ostale operacije pa z eno uporabljajo podatkovni tip xsd:string. Izjema je izhodni podatek iz operacije 'izpisi_predmete_operacija' in sicer je tipa Array (PredmetiArray). S tem podatkovnim tipom sem lahko poslal informacije o predmetih študija v strukturirani obliki, iz katere se lahko na strani uporabnika dobijo posebej podatki o nazivih predmetov in posebej šifre predmetov. Sam WSDL dokument kot opis spletne storitve sem izdelal s programom Altova XMLSpy (http://www.altova.com), ki je prikazan na sliki spodaj. Pravilnost zapisa sem kontroliral z WSDL Validatorjem [27] in WSDL Anylizer [40], WSDL dokument pa je priložen v prilogi A.



Slika 18: Prikaz strukture spletne storitve SisFggPredmet kot jo prikaže program XMLSpy

5.2 Izdelava OWL-s

Teoretični koncept OWL-s ontologije je bil predstavljen v poglavju 4.1. V nadaljevanju bom na primeru spletne storitve SisFggPredmet prikazal postopek za izdelavo te ontologije. Predpogoja za izdelavo semantičnega opisa sta WSDL datoteka spletne storitve in OWL zapis ontologije, na kateri bo temeljil OWL-s opis in ki opisuje koncepte, ki se pojavljajo v storitvi. OWL-s ontologijo sem izdelal na 2 načina:

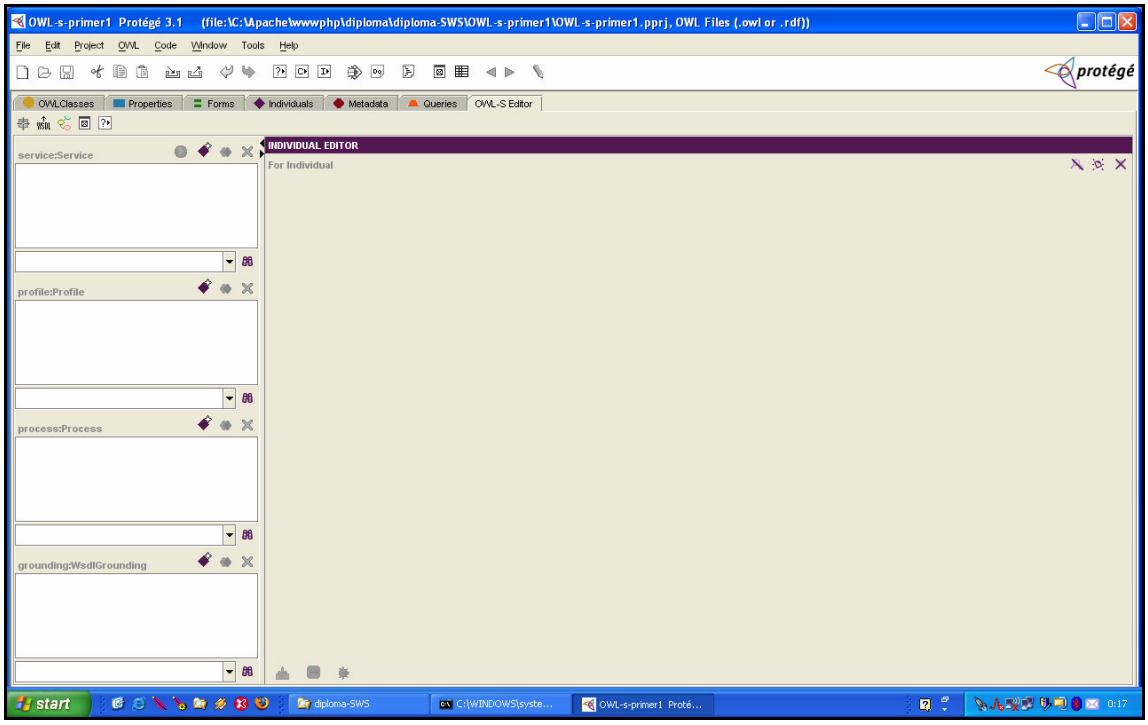
- Natančni način, kjer je vsakega od glavnih razredov potrebno narediti po korakih
- Avtomatizirani način, kjer se OWL-s naredi samo na osnovi WSDL opisa storitva brez povezovanja s kakšno temeljno ontologijo. Za ta primer sem v SisFggPredmet storitvi ohranil samo eno operacijo (profesor_predmeta_operacija), ker je to dovolj za demonstracijo izdelave semantičnega opisa po tej metodi.

5.2.1 Uporabljena programska oprema

Za izdelavo OWL-s ontologije sem uporabil modul OWLs [9] [19] za Protege, ki deluje v kombinaciji z modulom OWL Class. Dodatno sem za ustvarjanje Composite procesov s pomočjo grafičnega vmesnika potreboval program Graphviz. [30]

5.2.2 Izdelava OWL-s na podlagi ontologije predmeta storitve in WSDL

Za semantično anotacijo storitve sem aktiviral OWL-s tab, ki je modul za Protégé. Na začetku sem vnesel protege projekt ontologije predmeta in vklopil OWL-s zavihek (*tab*) . Modul je vnesel v projekt veliko zunanjih domen, ki so potrebne za izdelavo OWL-s opisa. Če niso potrebne spremembe osnovne ontologije (v tem primeru ontologija-sis-predmet.owl), se lahko celotni proces izdelave semantičnega opisa naredi v OWL-s zavihku. Ta na začetku kaže osnovno razdelitev strukture na *service*, *profile*, *process* in *grounding*.



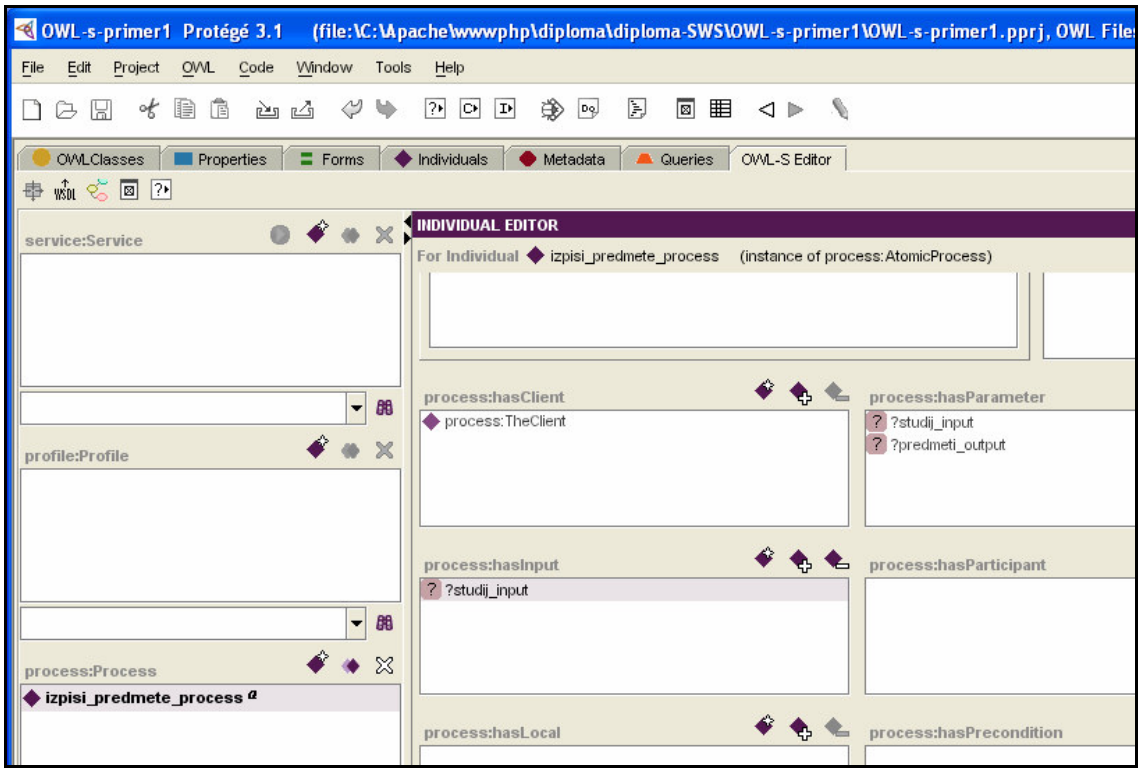
Slika 19: Začetni obrazec v OWL-s modulu Protege-ja

Izdelal bom primer, ki temelji na štirih operacijah storitve: izpis predmetov študija, izbira predmeta, izpis podatka o profesorju in izpis izpitnih rokov za izbrani predmet. Navodila za izdelavo so v priročniku 'An OWL-s Editor Tutorial' [9].

ServiceModel ali Process

Najprej bom definiral instance za razred *Process* in sicer je potrebno opisati 3 operacije, ki jih storitev izvaja. Vse 3 bom opisal kot osnovne procese, ki se kasneje lahko povežejo tudi v sestavljen proces.
Na začetku tega dela se definira vhodne in izhodne podatke posameznih procesov in sicer ime in podatkovni tip. Ti podatki storitve SisFggPredmet so opisani že v poglavju 5.1.3 Opis storitve. Tukaj se lahko podatkovni tip definira na dva načina: prvi je ta, da se mu določi kot podatkovni tip URI naslov nekega koncepta iz ontologije (ta je lahko tudi zunanja). Drugi je pa ta, da se mu določi en standardni podatkovni tip npr. *xsd:integer*.
Tako lahko zdaj naredim novo instanco za Process in izberem *'Atomic Process'*. V poljih za opis procesa sem tako definiral:

- ime procesa
- vhodne in izhodne podatke, kjer sem poleg imen določil tudi s katerimi koncepti so ti podatki skladni (npr. vhodnemu podatku studij_input sem določil koncept studij iz osnovne ontologije)

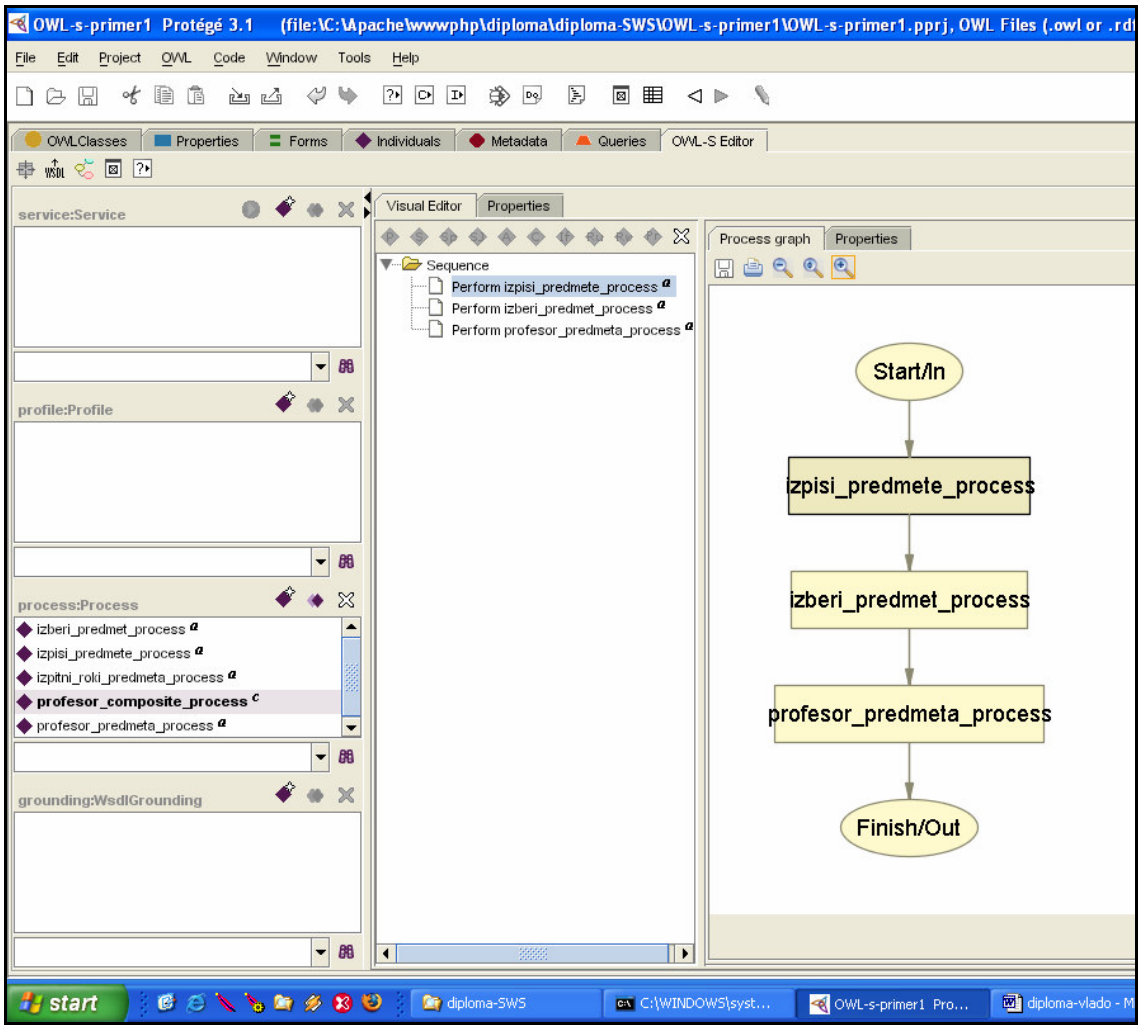


Slika 20: Prikaz situacije po definiranju vhodnih in izhodnih podatkov za osnovni proces, ki bo prikazoval operacijo storitve izpisi_predmete_operacija

Vse procese je potrebno definirati na takšen način, tako sem nato definiral še procese izberi_predmet_process, izpitni_roki_predmeta_process in profesor_predmeta_process.

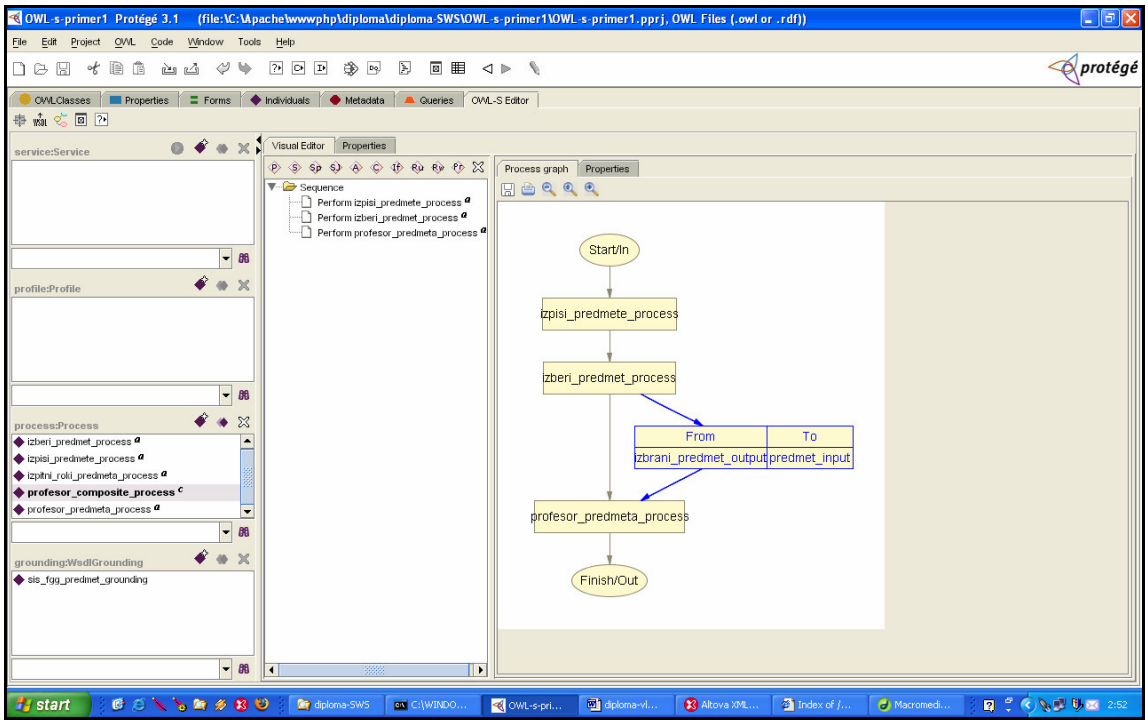
Ko so deklarirani vsi osnovni procesi je mogoče narediti še sestavljen proces iz kombinacije osnovnih procesov. Jaz bom naredil sestavljen proces iz procesov izpisi_predmete_process, izberi_predmet_process in profesor_predmeta_process. Pri tem procesu sem predpostavil, da je zgrajen iz zaporedja: vnos študija (izpisi_predmete_process) –> izbira predmeta (izberi_predmet_process) –> pokaži profesorja predmeta (profesor_predmeta_process). To

zaporedje sem zgradil z gradniki, ki so v OWL-s urejevalniku za sestavljene procese, uporabil pa sem operaciji zaporedje (*sequence*) in izvedi (*perform*). Urejevalnik omogoča še nekatere druge kot so razdeli, združi, izberi, 'ponavljaj dokler'…



Slika 21: Ustvarjanje sestavljenega procesa z OWL-s modulom

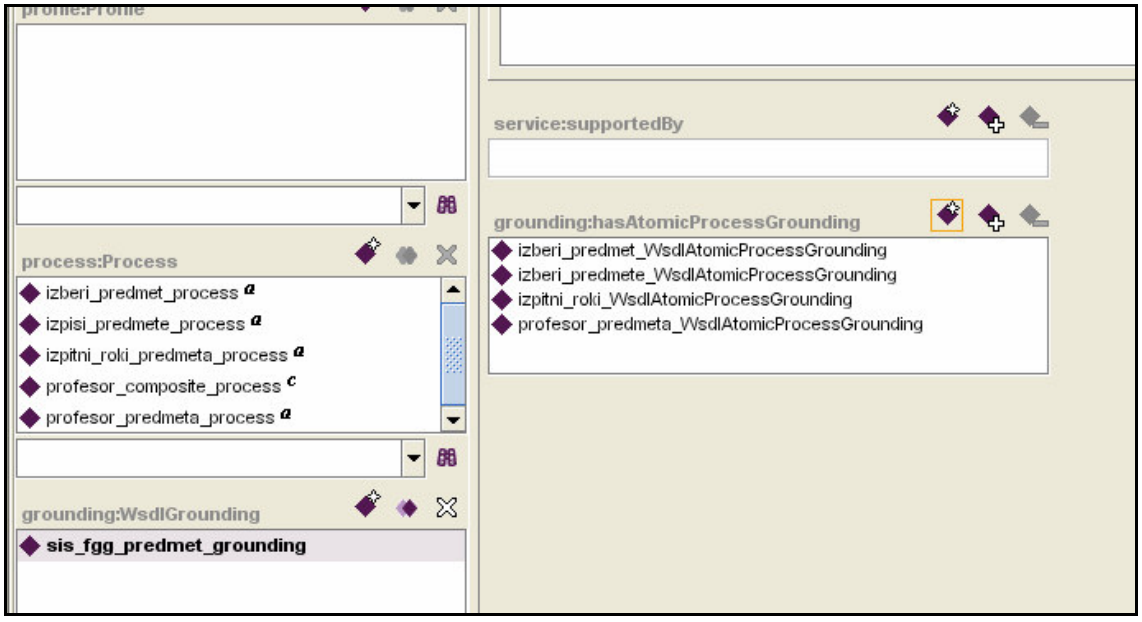
Ko sem imel zaporedje procesov, sem definiral še potek izmenjave podatkov med njimi (*dataflow*). To sem tudi naredil po navodilih, kjer je jasno opisan potek korakov za definiranje izmenjave podatkov. Urejevalnik omogoča tudi zelo pregleden grafični prikaz celotnega procesa in izmenjave podatkov med njimi.



Slika 22: Grafični prikaz celotnega sestavljenega procesa in izmenjave podatkov med posameznimi procesi.

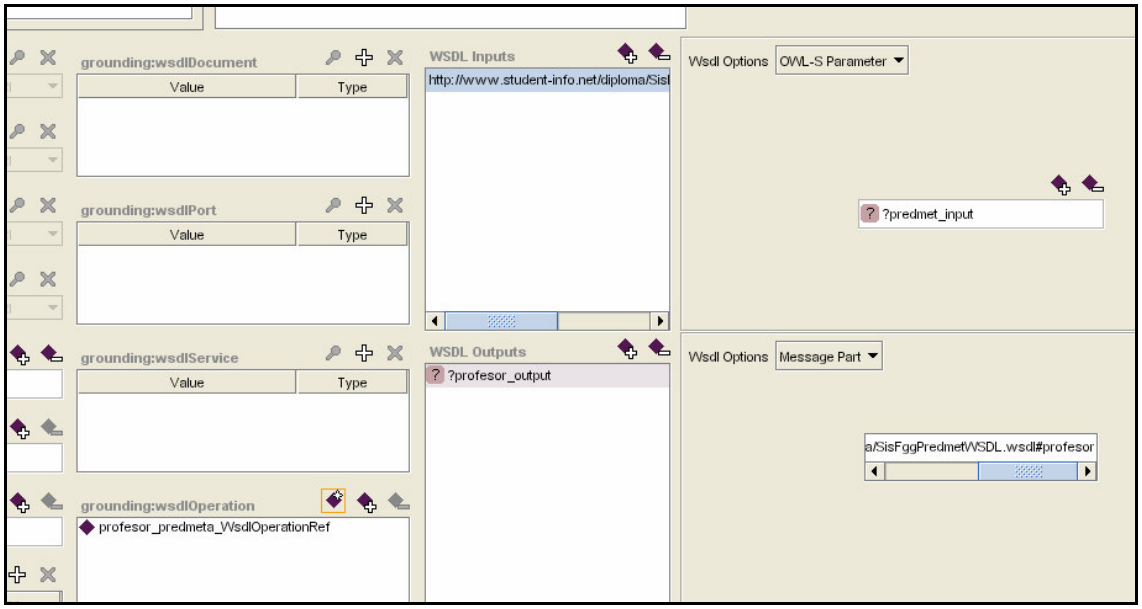
ServiceGrounding

Sledi definiranje *ServiceGrounding*-a, kjer je potrebno tako kot pri *Processu* najprej ustvariti eno instanco *Grounding-a*. Jaz sem svojo poimenoval sis_fgg_predmet_grounding. Za vsak osnovni proces je potrebno definirati instanco *WsdlAtomicProcessGrounding*. Tu sem za vsakega določil *owlsProcess*, kjer se izbira med deklariranimi osnovnimi procesi. Za nadaljnje delo je potrebno poznati URI lokacijo WSDL opisa storitve. V mojem primeru se ta nahaja na http naslovu http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl.



Slika 23: prikaz definiranih instanc Groundinga za vsak osnovni proces

Zdaj je na vrsti povezovanje med vhodnimi in izhodnimi podatki *Grounding-a* in deli sporočil (*message part*) WSDL dokumenta storitve. Tukaj je mogoče izbirati med možnostjo navezovanja OWL-s parametra na del sporočila, ali obratno del sporočila na OWL-s parameter. Jaz sem izbral prvo možnost. Tako sem npr. OWL-s parameter predmet_input, ki je definiran na osnovi razreda predmet v osnovni ontologiji, povezal z delom sporočila predmet, ki je vhodni podatek za operacijo profesor_predmeta_operacija. Tukaj je za posamezne *grounding-e* potrebno še definirati *grounding:WsdlOperation* in sicer z vpisom URI naslova operacije, ki ponazarja ta osnovni proces (npr. http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl#profesor_predmeta_operacija) in vhodom storitve, preko katerega poteka operacija storitve (http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl#SisFggPredmet_PortType).

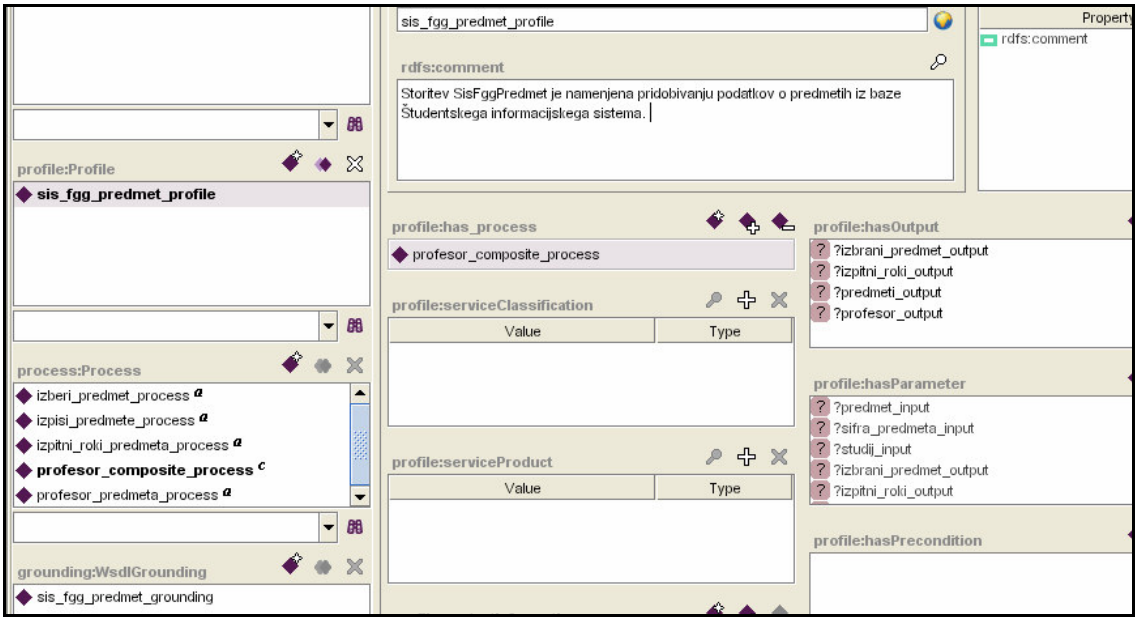


Slika 24: Prikaz definiranja povezav med OWL-s razredi in deli sporočila v WSDL dokumentu

ServiceProfile

Spet je potrebno najprej narediti novo instanco, ki sem jo jaz poimenoval sis_fgg_predmet_profile. Naslednji korak je opis storitve povezati s procesom. V mojem primeru sem storitev povezal s sestavljenim procesom profesor_composite_process. Mogoče je seveda storitev urediti tudi okrog osnovnega procesa.

Podatki o vhodnih in izhodnih podatkih storitve se morajo do neke mere ujemati s podatki *Process-a*, seveda je pa jih je mogoče, kot je bilo že omenjeno pri teoretičnem opisu OWL-s, tudi drugače prikazati. V večini primerov se bodo podatki v *Profile* in *Process* ujemali in tako sem tudi jaz izbral vse, že prej deklarirane vhodne in izhodne podatke, ki se pojavljajo v *Process* opisu in bodo kot takšni prikazani tudi v opisu storitve *Profile*.



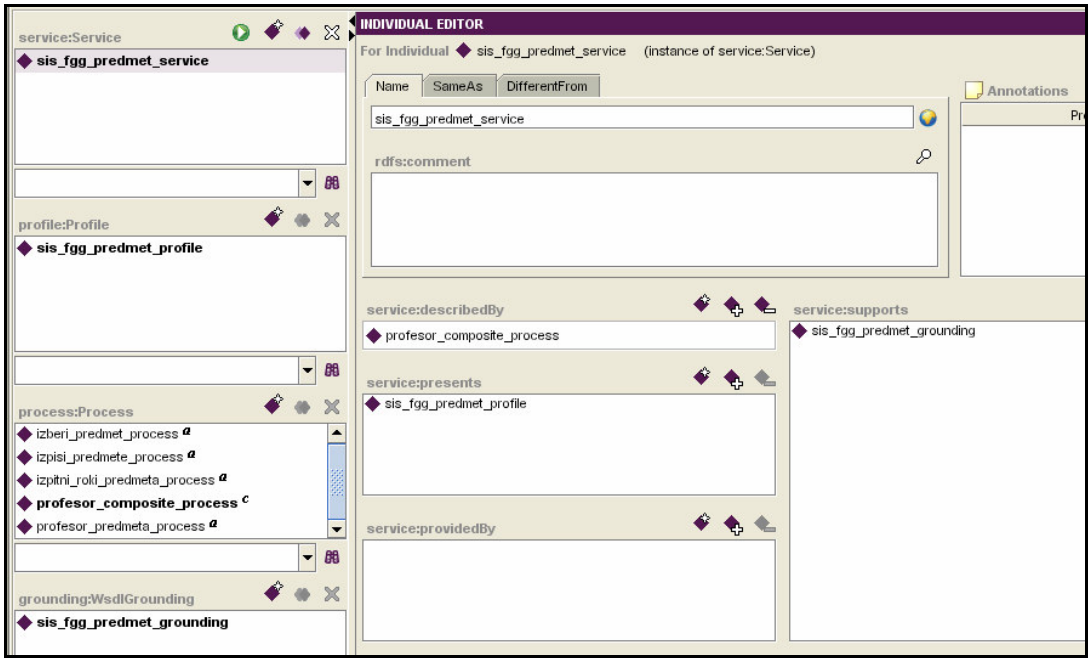
Slika 25: Prikaz podatkov v OWL-s ontologiji Profile za SisFggPredmet storitev

Service

Nazadnje sledi še deklaracija *Service*, ki povezuje ostale 3 razrede ontologije v celoto. Potrebno je definirati vse 3 njene lastnosti in tako sem za moj primer izbral prej izdelane razrede:

- *Process* (*desribedBy*): profesor_composite_process
- *Profile* (*presents*): sis_fgg_predmet_profile
- *Grounding* (*supports*): sis_fgg_predmet_grounding

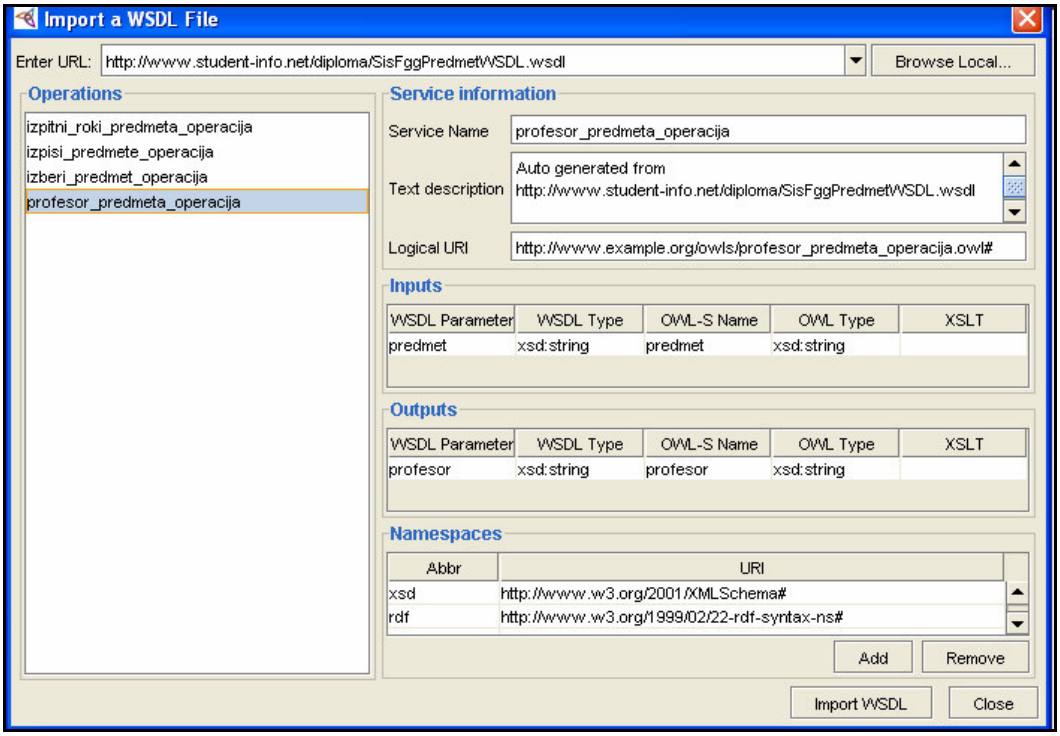
OWL zapis celotne ontologije je priložen v prilogo C.



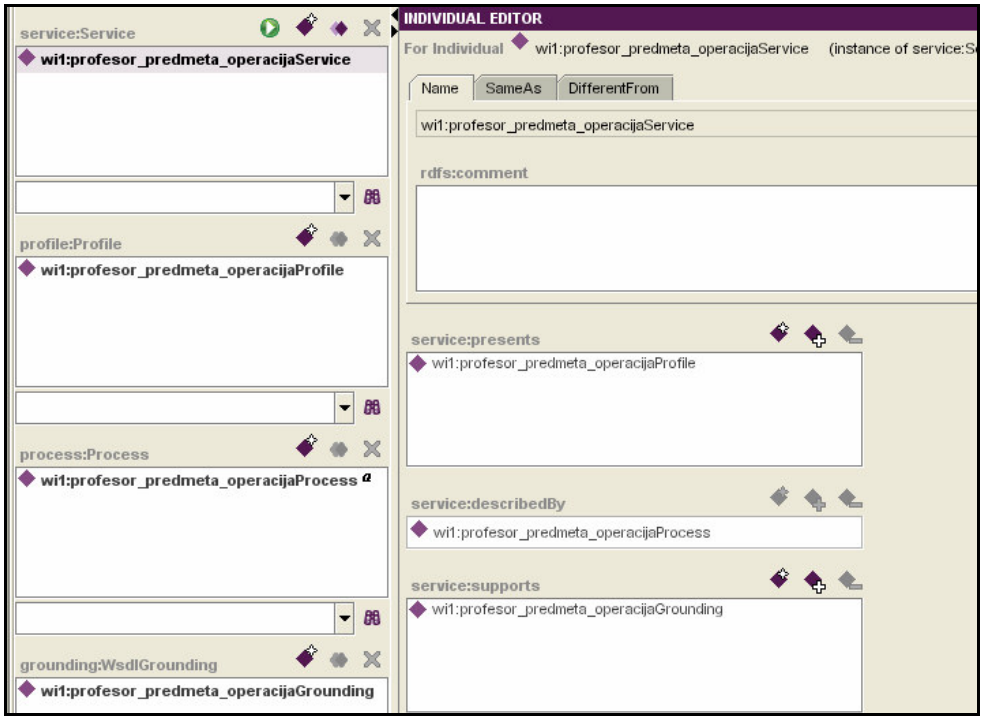
Slika 26: Prikaz Service opisa, ki povezuje ostale 3 OWL-s ontologije

5.2.3 Avtomatizirana izdelava OWL-s samo na podlagi WSDL

OWL-s modul ima še eno možnost izdelave OWL-s ontologije, ki je precej hitrejša kot ta, ki sem jo opisal v prejšnjem poglavju in temelji na WSDL2OWLS aplikaciji [28], ki je vgrajena v OWL-s modul. Namenjena je za osnovne procese (posamezne operacije) in temelji samo na WSDL datoteki. Nikakršna osnovna ontologija konceptov ni potrebna. S to metodo naj bi se obdelale že obstoječe preproste storitve. Celoten postopek se sestoji iz vnosa WSDL datoteke (lahko je tudi URI WSDL dokumenta). V obrazcu program WSDL2OWLS prikaže vse podatke, ki jih je lahko povzel iz WSDL opisa za izbrano operacijo. Mogoče je še spremeniti opis storitve in dodati ali odstraniti še kakšno domeno, ostalo pa je prepuščenu avtomatični obdelavi. Celoten postopek traja približno toliko, kolikor vzame računalniku časa da analizira WSDL dokument in naredi OWL-s ontologijo te operacije. OWL zapis te ontologije za operacijo profesor_predmeta_operacija je v prilogi D.



Slika 27: Prikaz vnosa WSDL dokumenta za avtomatizirano metodo izdelave OWL-s ontologije



Slika 28: Rezultat avtomatizirane izdelave ontologije OWL-s

V prilogo E sem dodal OWL-s *Grounding* za osnovni proces operacije iskanja profesorja predmeta, ki je bila narejena z avtomatično metodo izdelave (import WSDL). Povezave med WSDL in OWL-s Groundingom so označene s krepkim textom. Na enak način se da narediti tudi povezave med WSDL in OWL-s, če je bila ontologija storitve narejena z detajlnim načinom. WSDL dokumentu ni potrebno posebej dodajati semantične anotacije preko razširitvenih elementov pri nobenem od teh postopkov, ker se agenti pri tej metodi semantičnega opisa orientirajo samo na OWL-s zapis. Sem pa zaradi boljše preglednosti povezav med elementi storitve in njenim semantičnim opisom za moj primer dodal še prikaz teh povezav preko dodatnih atributov in razširitvenih elementov v WSDL opisu SisFggPredmet storitve – priloga F.

Primerjava obeh postopkov

Dolgi in detajlni postopek je potrebno uporabiti v naslednjih primerih:

- Če želimo ustvariti OWL-s zapis, ki ga lahko naknadno popravljamo
- Če je cilj semantično predstaviti sestavljeni proces in ne samo osnovni
- Če je potrebno povezovanje WSDL *message part-ov* s koncepti ontologije, na kateri storitev temelji ali morebiti celo s kakšno zunanjo ontologijo
- Če obstaja potreba po bolj natančnih opisih za oglaševanje storitve

Izdelava OWL-s opisa storitve na daljši način ni preveč zahtevna, zahteva pa razumevanje načina delovanja OWL-s. V primeru:

- da tega znanja nimamo ali če se ne želimo toliko poglobiti v OWL-s semantične opise spletnih storitev
- če je storitev sestavljena iz ene same operacije kot osnovnega procesa
- če so imena delov sporočil takšna, da bi jih agenti lahko vsaj delno povezali kot možno ustrezno storitev in jo zato ponudili v izbiri z ostalimi najdenimi storitvami
- da ni prevelike potrebe po bolj kompleksnem oglaševanju storitve

pa bi bil hitri način bolj primerna izbira izdelave semantične spletne storitve.

Hitri način izdelave bi verjetno bil zelo uporaben v primeru, ko bi obstajala neka referenčna, splošno sprejeta osnovna ontologija področja, na katerem dela razvijalec storitev in bi upošteval 3 pravila:

- v svojih OWL-s ontologijah nima sklicevanja na razrede te ontologije, ampak vse svoje koncepte zastavlja tako, kot so zastavljeni osnovni referenčni koncepti
- da bi uporabljal v svojih storitvah enaka imena, kot so v osnovni ontologiji tudi za svoje koncepte
- določil bi klasifikacijo storitve. Ta bi odpravila potencialne težave, ki bi nastale, če bi ontologije različnih področij imele zaradi specifičnih potreb malo spremenjene iste koncepte

Tako bi lahko agent iz klasifikacije videl, če storitev ustreza področju iskanja, iz imena dela sporočila bi videl, za kateri koncept gre, in če je koncept v ozadju delovanja operacije enak konceptu ontologije, takrat storitev uporabniku ali agentu vrne rezultat, katerega položaj je v ontologiji enolično določljiv, pričakovan in agentu razumljiv.

5.3 Izdelava WSDL-s

Za primer izdelave WSDL-s sem spet uporabil WSDL opis storitve SisFggPredmet in ontologijo predmeta. Namen je pokazati, kako se povezuje WSDL elemente in razrede ontologije in sicer v primeru WSDL 1.1 in WSDL 2.0, ki je tudi WSDL-s opis storitve.

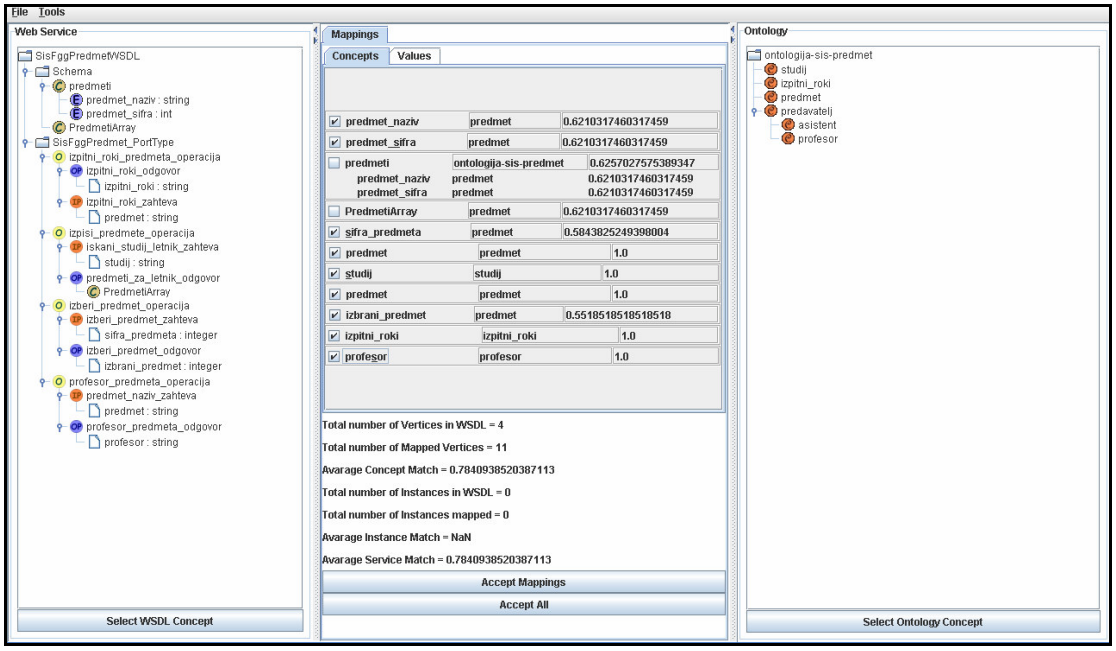
5.3.1 Uporabljena programska oprema

Za primer izdelave semantične anotacije sem uporabil program METEOR-s [17] in njegov modul MWSAF [18]. METEOR-s je razvit z namenom omogočanja avtomatičnega in polavtomatičnega anotiranja WSDL z razredi ontologije, poleg tega pa ima v svojem priročniku omenjeno podporo tudi za iskanje semantičnih WSDL-s opisov storitev po registru, v svojem razvoju bo pa omogočil tudi polavtomatično kompozicijo procesov različnih storitev. Za svoje delovanje zahteva Eclipse 3.0 (http://www.eclipse.org), Java 5 (http://java.sun.com), Ant (http://ant.apache.org), Tomcat (http://jakarta.apache.org/tomcat), Axis (http://ws.apache.org/axis) in WordNet 1.7+ (http://www.cogsci.princeton.edu/~wn/wn2.0.shtml) za delovanje modula MWSAF. Tukaj

moram povedati, da mi po nastavitvi vseh teh programov niso delale vse funkcije METEOR-s (npr. iskanje po registrih opisov storitev, kompozicija procesov iz različnih storitev…).

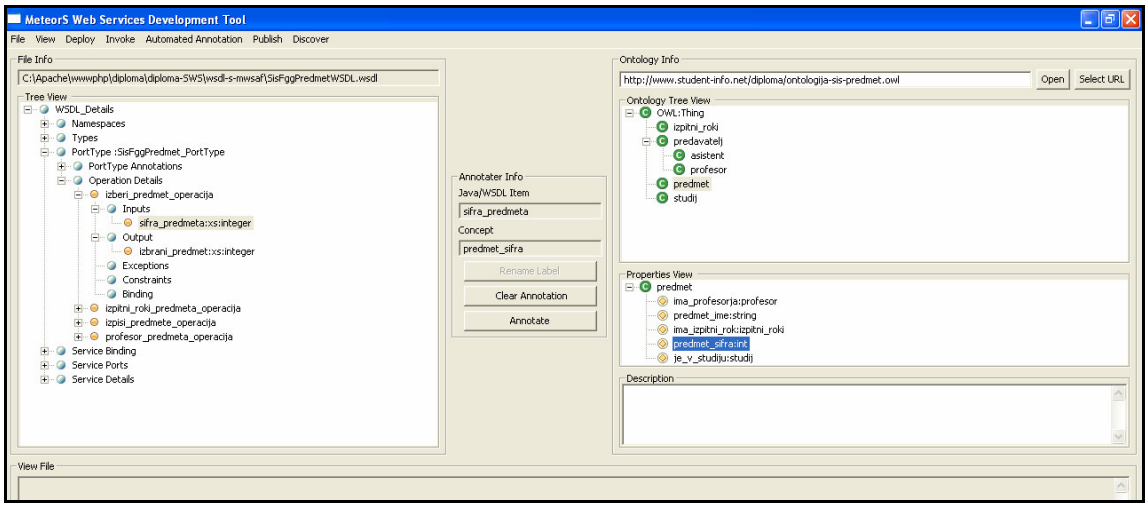
5.3.2 Možnosti semantične anotacije na podlagi standarda WSDL opisa storitve

Program temelji na preprostem principu: v enem oknu je narejen prikaz WSDL elementov, ki jih lahko semantično anotiramo, v drugem pa so prikazani razredi ontologije, ki je lahko vnešena tudi iz URL naslova.
Polavtomatični način izgleda tako, da modul MWSAF na osnovi podobnosti imen kot nabora znakov poda možnosti ustvarjanja semantičnih povezav. Če bi vedeli, da se imena elementov in razredov dobro ujemajo, bi lahko uporabili funkcijo avtomatične anotacije programa. Zaradi tega je pomembno uporabljati imena, ki so sprejeta v večjem krogu organizacij in razvijalcev. Če se imena vsaj delno ujemajo, je precej manj dela pri semantični obdelavi WSDL datoteke. V srednjem oknu so poleg podobnih imen napisani tudi deleži ujemanja imen. Pri avtomatičnem načinu se tako pred anotacijo določi, do katere stopnje se morata imeni koncepta ontologije in elementa WSDL ujemati, da se naredi semantična anotacija elementa.



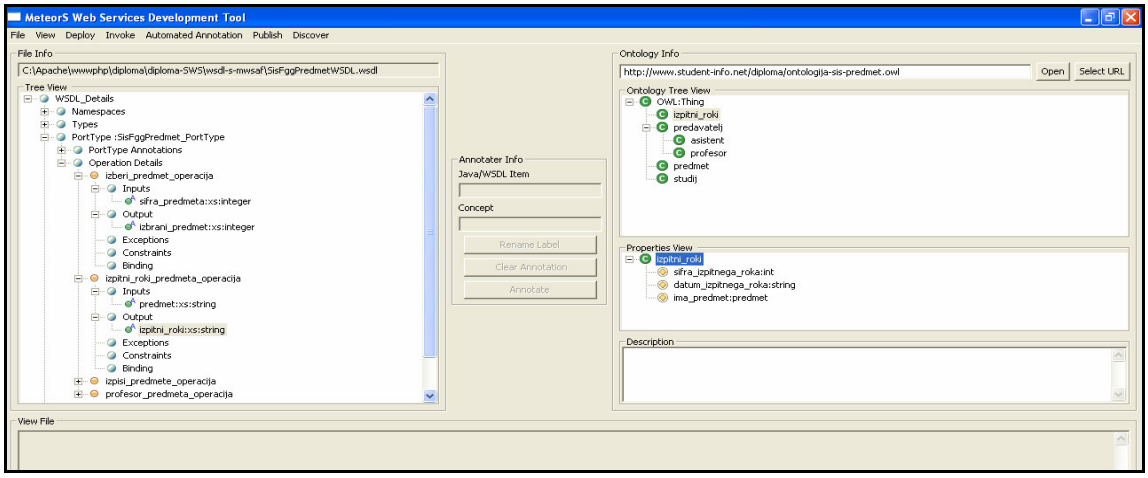
Slika 29: Prikaz ponujenih povezav in izbira s strani uporabnika programa MWSAF

Semantično anotacijo sem izdelal s programom METEOR-s, kjer anotacija temelji na MWSAF modulu, ampak za razliko od dela samo z modulom MWSAF mi je tukaj uspelo vzpostaviti funkcije za izpis anotirane WSDL datoteke. Prav tako sem pričel z izbiro WSDL datoteke storitve in ontologije, vendar za razliko od MWSAF ponuja METEOR-s pogled tudi na lastnosti razredov ontologije, kar je pri tako detajlnih zadevah kot je predmet, ki ima šifro in svoj naziv zelo pomembno pri točnosti povezav.



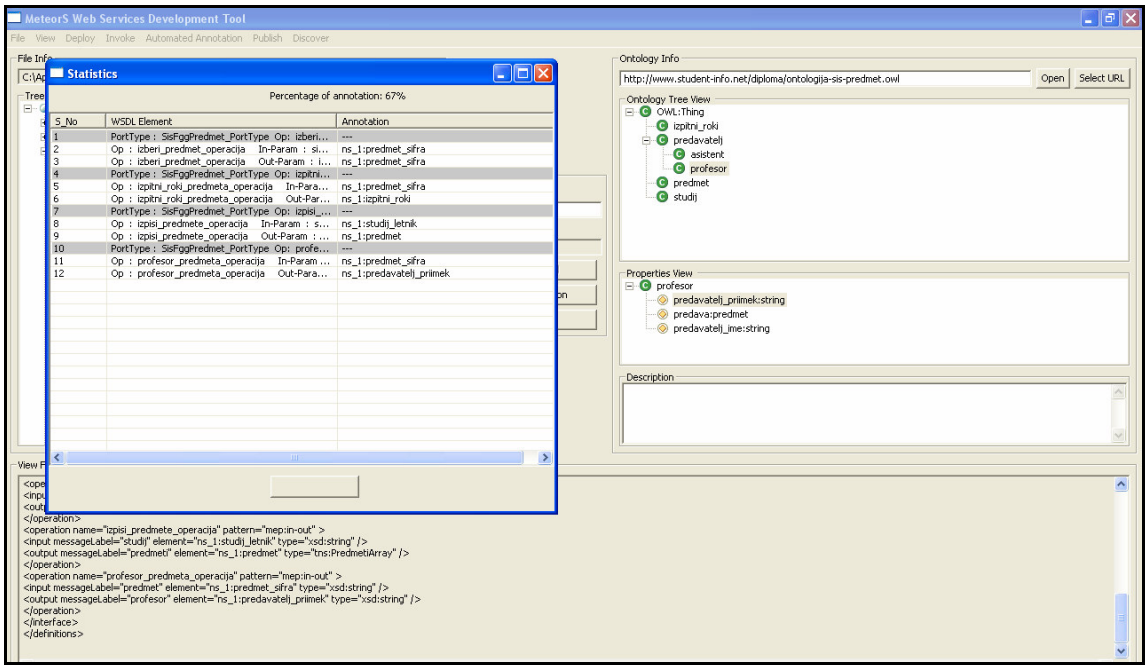
Slika 30: Prikaz elementov WSDL in razredov ontologije v METEOR-s

Tukaj sem semantično anotacijo izvedel 'ročno' in sicer kot kaže prejšnja slika sem izbral en element WSDL, ki ga je mogoče anotirati in nato še ustrezen koncept na strani ontologije. Primerjavo sem sprožil z gumbom *'Annotate'*. METEOR-s anotiran element posebej označi z drugačno ikono (zelen krog s črko A pred imenom elementa). Tako sem naredil povezave na vseh elementih, ki jih lahko semantično anotiram na osnovi te ontologije.



Slika 31: Prikaz anotiranih elementov v METEOR-s

Po končani anotaciji lahko program prikaže povzetek vseh anotacij.



Slika 32: Povzetek semantičnega anotiranja z METEOR-s

Med priloge G in H sta priložena:

- semantična anotacija WSDL 1.1 opisa storitve s programom METEOR-s (priloga G)

- WSDL-s opis storitve SisFggPredmet. WSDL 2.0 opis storitve je program METEOR-s sestavil na osnovi WSDL 1.1 datoteke (priloga H)

5.4 Simulacija odkrivanja spletnih storitev na osnovi dodanih semantičnih opisov

V tem poglavju sem želel izvesti lokalno simulacijo iskanja agenta (*matchmaking*), vendar mi tega s programi, ki sem jih našel na spletu in ki so bili opisani kot delne implementacije agenta, tega ni uspelo. Simulacijo sem želel izvesti z uporabo semantične opise storitve iz praktičnega primera, saj bi tako lahko spreminjal nekatere parametre kot so npr. ime vhodnih podatkov in opazoval, kako te spremembe vplivajo na simuliranega agenta in njegove rezultate iskanja. Dokaz za uporabnost semantičnih storitev bi izvedel tako, da bi prvotnemu semantičnemu opisu storitve zamenjal imena vhodnih in izhodnih parametrov ter ga shranil kot semantični opis druge storitve. Ker se ti parametri sklicujejo na isto ontologijo in tudi na iste koncepte, potem bi v rezultatu iskanja po registru (mapi semantičnimi opisi) program moral vrniti obe storitvi. Za rezultate iskanja ne sme biti ključno ime parametrov, ampak mora agent pogledati, katere koncepte storitev uporablja.

Poskusil sem s programi:

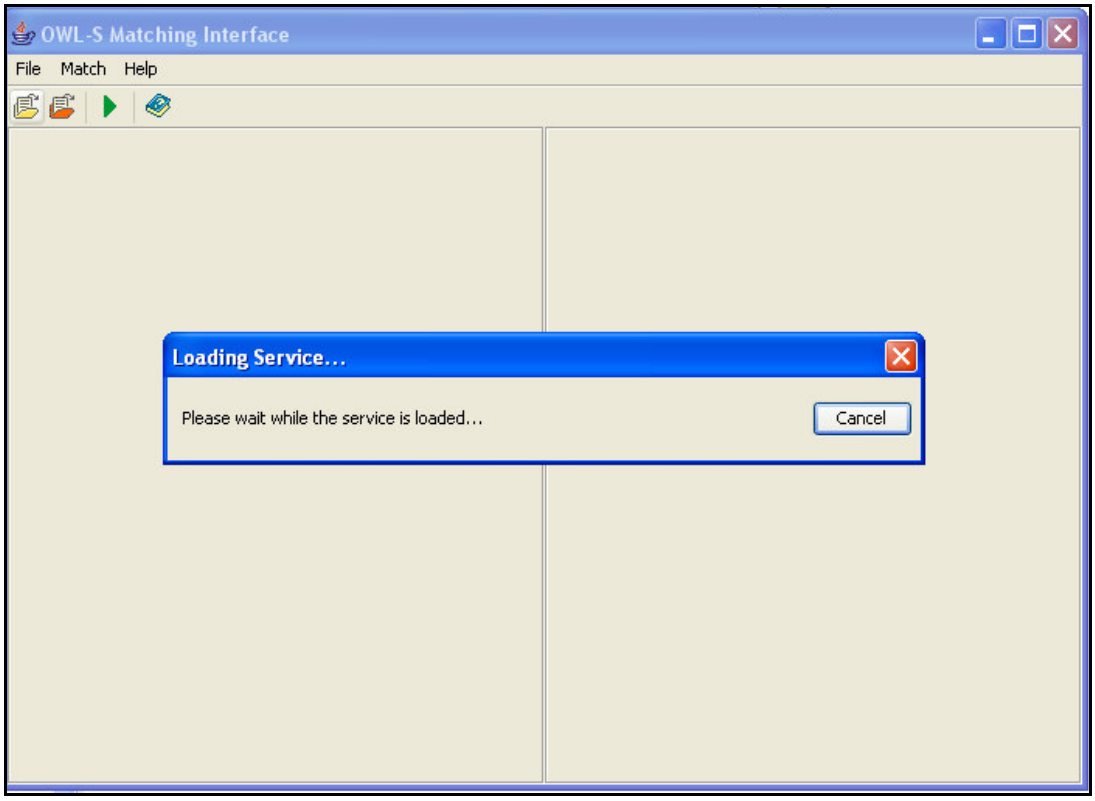
- OWL-s MX (http://www.dfki.de/~klusch/owls-mx/), ki deluje preko ukazne vrstice na operacijskem sistemu Windows.

- OWL-s Matcher (http://ivs.te-brlin.de) mi je po težavni namestitvi uspelo zagnati, vendar program ni uspel izpolni niti prvega vnosa OWL-s ontologije.

- METEOR-s (ki je že bil opisan v poglavju izdelave WSDL-s) je zelo dodelan program, veliko funkcij je delalo normalno, ampak tudi tukaj ni uspela simulacija iskanja ustrezne storitve.

Večina programov, ki je narejenih za semantične spletne storitve uporablja za dodatno podporo odprtokodne programe kot so Axis, Maven, Tomcat in zunanje Java module.

Predvidevam, da je bila simulacija neuspešna zaradi slabe namestitvene kombinacije vseh potrebnih programov, operacijskega sistema in uporabe različnih standardov. Tukaj mislim na razliko med standardi, za katere je bil program testiran in standardi uporabljenih datotek, ki sem jih uvažal kot produkte ostalih programov npr. Protege-ja. Za tekočo izvedbo takšne simulacije bo potrebno počakati na celovito narejen program, dovolj dobro testiran za različne situacije in ki bo imel vgrajene funkcije, ki opozarjajo na napake. Te sem pri teh testiranih programih najbolj pogrešal, saj nisem mogel odpravljati napak, ampak sem jih poskušal poiskati na osnovi nedelovanja posameznih funkcij.

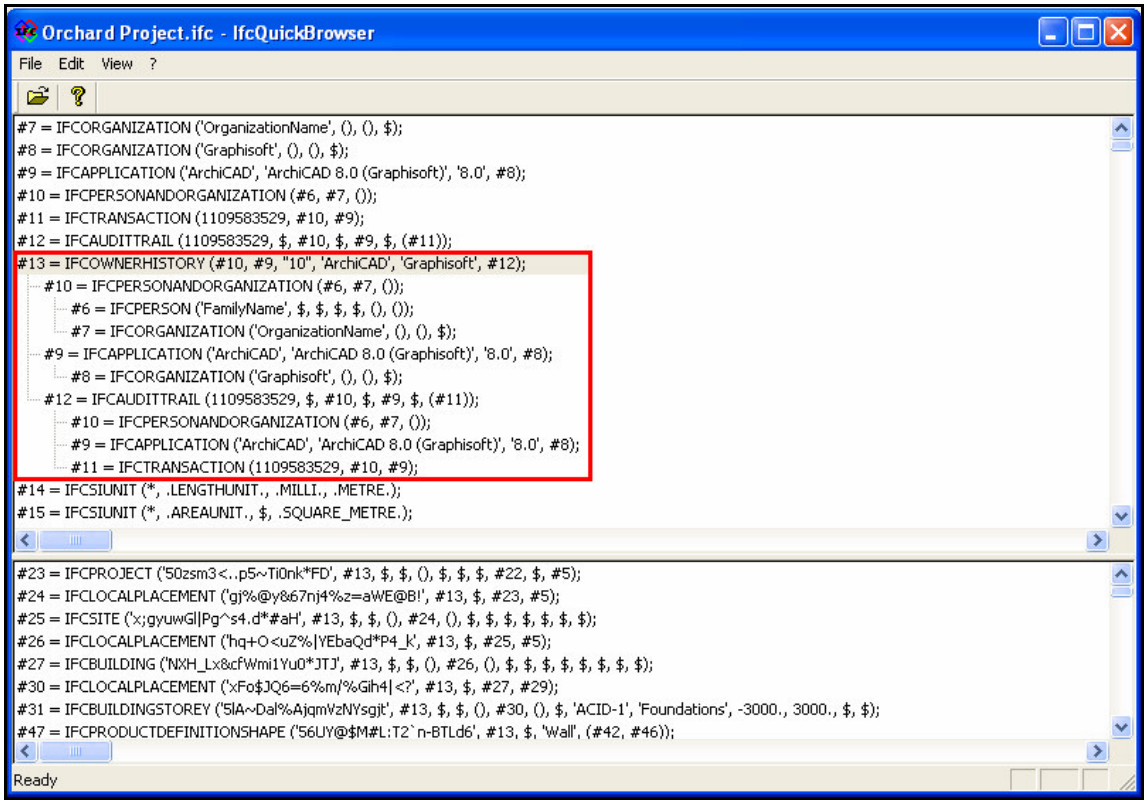


Slika 33: Prikaz grafičnega vmesnika OWL-s Matcher programa

6 MOŽNOSTI UPORABE SEMANTIČNIH SPLETNIH STORITEV V GRADBENIŠTVU

Semantične spletne storitve verjetno ne bodo imele neposrednega vpliva na izvajanje del na gradbišču, lahko pa naredijo velik napredek v fazi načrtovanja in projektiranja gradbenega objekta. V prihodnosti bo na področju IT v gradbeni industriji verjetno največ pozornosti potrebno nameniti kompatibilnosti različnih programov v uporabi. Ti so trenutno večinoma nekompatibilni, kar je eden izmed razlogov za relativno malo število programov v množični uporabi (npr. AutoCAD, SAP 2000…), saj mora program zadostiti večini potreb uporabnika, da ga ta vzame kot osnovni pripomoček za svoje delo. Zato tukaj iz te množične uporabe izpade velika večina manjših, bolj specializiranih aplikacij. Obsojene so na zelo ozek krog uporabnikov, veliko ostalih, ki bi jih za svoje delo mestoma lahko uporabili, pa jih sploh ne pozna. Kljub temu, da verjetno nekatere aplikacije dajejo zelo dobre rezultate.

Zaradi potreb gradbene industrije po kompatibilnosti je bil razvit produktni model IFC (4) (*Industry Foundation Classes*), ki je s standardom sprejeta oblika zapisa podatkov celotnega objekta in ga uporablja že kar nekaj programov za računalniško projektiranje in modeliranje konstrukcij (npr. ArchiCAD). IFC tako daje osnovo za prestrukturiranje vseh obstoječih aplikacijah, če jih seveda ne bi bilo lažje jih na novo zasnovati in sprogramirati. Hkrati je vodilo za vse novonastajajoče, če želijo biti kompatibilne z uveljavljenimi programi, ki ta koncept že uporabljajo in z vsemi novimi, ki bodo na njegovi osnovi narejeni. Kompatibilnost v tem primeru pomeni, da lahko z enim programom naredimo en del projekta, po končanem delu pa iste datoteke projekta odpremo v drugem programu, ki je lahko zasnovan kot nadgradnja prejšnjega ali pa omogoča izdelavo ostalih delov projekta. V obeh primerih se podatki prenesejo naprej in so na voljo za nadaljno uporabo ali obdelavo.
Podobnen model so pred nekaj leti naredili tudi v energetskem združenju ZDA, saj so videli, da se stroški prilagajanja izmenjanih podatkov za nadaljno uporabo in analizo s količino teh podatkov strmo večajo.



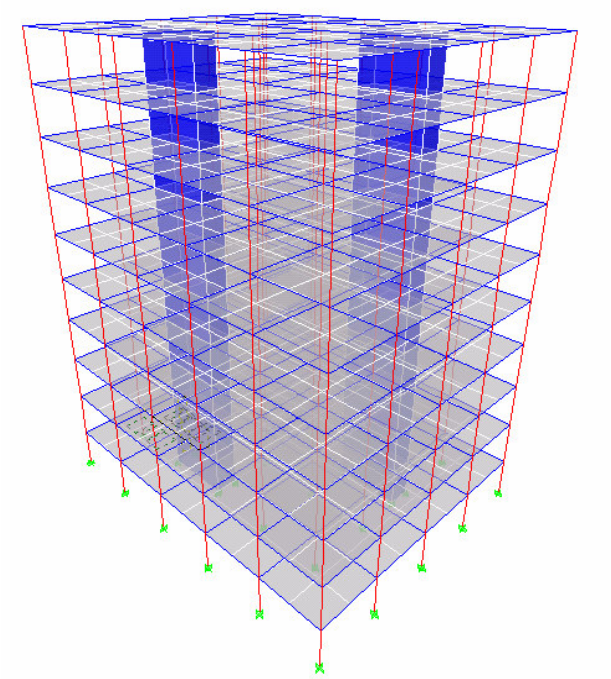
Slika 34: Primer zapisa IFC

Vir: [http://www.inteligrid.com]

Model IFC tako veliko obljublja na področju gradbene informatike, vendar trendi v IT dodatno narekujejo prehod na uporabo programov kot storitev preko interneta. Razlogi za takšno usmerjenost so navedeni v uvodu te diplomske naloge, kjer so naštete pozitivne lastnosti spletnih storitev.

Tukaj kot naslednja razvojna stopnja nastopijo semantične spletne storitve, ki poleg tega, da so to v osnovi spletne storitve, lahko v veliki meri izkoriščajo tudi dobro zasnovo IFC modela. Le-ta je zelo primeren za zapis ontologije vseh konceptov, ki se pojavljajo v načrtu gradbenega objekta, saj je tudi IFC sam nekakšna ontologija. Če bi na tem mestu pomislili, zakaj bi pa potem sploh delali temeljno ontologijo, je odgovor precej preprost - semantične spletne storitve bodo prinesle velike spremembe na veliko področjih, ne samo gradbeništva. Zato jih je potrebno vpeljati kot splošne, ker s tem pa ostaja možnost izkoričanja tudi vseh nadaljnih izboljšav, ki bodo razvite na tem področju.

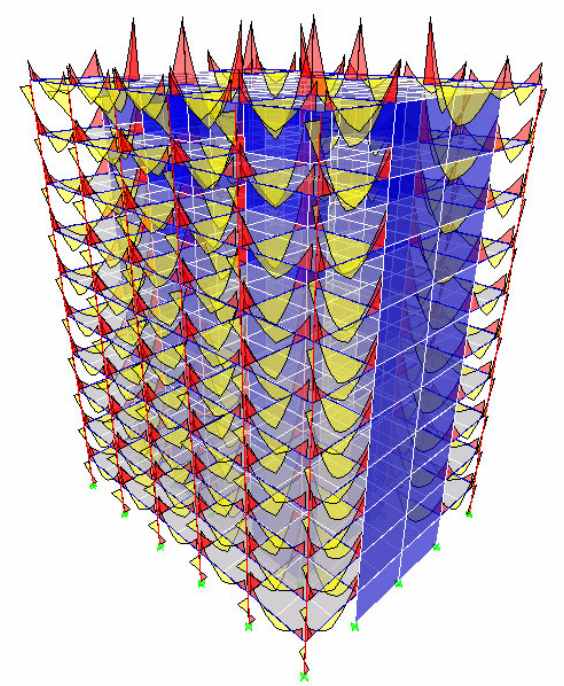
Semantične storitve bi seveda zagotovile tudi bolj merodajne rezultate iskanja ustreznih storitev po registru. S časom bo na voljo vse več registriranih storitev, med delno ustreznimi (izpolnjujejo samo del zahtev uporabnika) pa bi se imele možnost pojaviti tudi tiste manjše. Z avtomatizirano kompozicijo manjših storitev, ki bi jo delali agenti, bi lahko v nekaterih primerih prišli do enakih ali zelo podobnih rezultatov, kot bi jih dobili ob uporabi ene večje, bolj kompleksne storitve. Zakaj bi pa delali z več storitvami, če lahko vse opravimo z eno? Hkrati pa je še nekaj uvodnega dodatnega dela s spoznavanjem vsake storitve in njenega delovanja. Odgovor je spet precej preprost – cena. Kompleksnejše storitve so navadno licenčno plačljive, medtem ko so manjše ponavadi brezplačne, saj se razvijalci zavedajo, da so rešili le del problema ali pokrili le del celotnega procesa izdelave projekta. Semantične spletne storitve z agenti tukaj naredijo korak še en naprej, glede na trenutni potek dela. Kombinacije kompozicij manjših storitev, ki bi jih ustvarili agenti v svojem iskanju in ponudili uporabniku poleg večjih storitev (ki bi lahko samostojno zadostile vsem zahtevam uporabnika) zelo zmanjšajo začetni časovni in delovni vložek analize novih storitev. Zelo dober prikaz ustreznosti ponudi že agent in s tem privarčuje ogromno časa, ki bi ga za samostojno iskanje porabil iskalec ustrezne storitve. Hkrati pa je računalniška analiza po avtomatiziranih postopkih navadno precej bolj učinkovita, kot če bi jo izvajal človek. Za primer bi lahko vzeli izdelavo statičnega izračuna na osnovi podane konstrukcije in obtežbe. Eno storitev bi npr. lahko uporabili za izdelavo načrta konstrukcije (slika spodaj) in njen prikaz.



Slika 35: Storitev, ki ponuja prikaz konstrukcije objekta

Vir: [http://www.inteligrid.com]

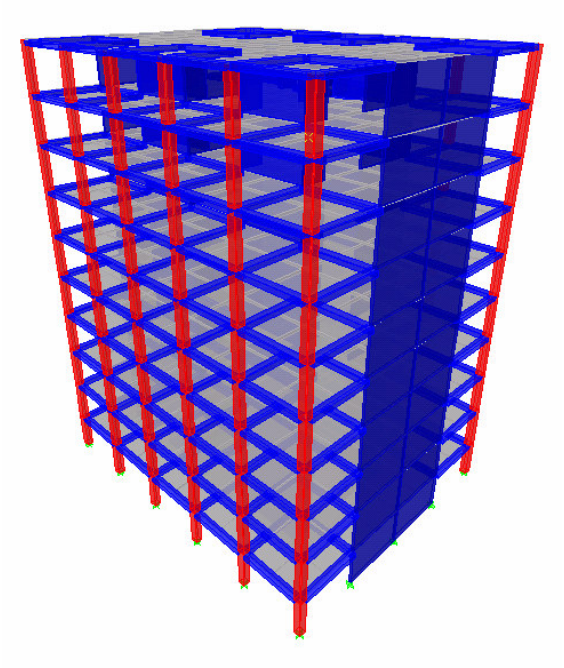
Naslednja storitev mora biti sposobna narediti statično analizo konstrukcije, dimenzioniranje in prikazat rezultatov (kot ta na naslednji sliki). V tem primeru bi bilo na voljo verjetno največ možnosti, saj obstajajo programi, ki naredijo analizo, ne omogočajo pa grafičnega prikaza, le-ta je omejen samo na dve dimenziji, analize uporabljajo različne metode za izračune… Vse skupaj je seveda odvisno od zahtev, potreb, pa tudi zmožnosti, saj nekatere storitve zahtevajo zelo veliko računalniških virov in časa.



Slika 36: Grafični prikaz rezultatov statične analize konstrukcije

Vir: [http://www.inteligrid.com]

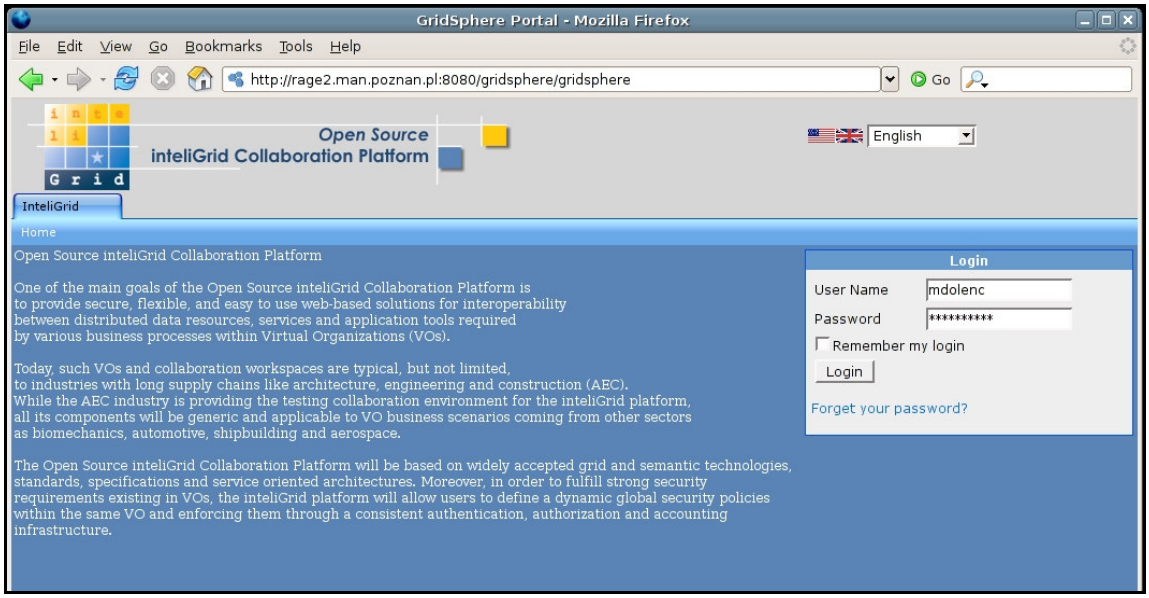
Po končani statični analizi spet vzamemo storitev, ki omogoča prikaz načrta konstrukcije in ki bo upoštevala rezultate, pridobljene v prejšnjem koraku. Seveda ni potrebno, da je to storitev, ki je bila že v začetku uporabljena, je pa njena uporaba že preverjena in kot takšna omogoča tekoče delo naprej.



Slika 37: Storitev omogoča prikaz dimenzij posameznih elementov konstrukcije
Vir: [http://www.inteligrid.com]

Zgoraj prikazani primer seveda ne pomeni, da bodo manjše storitve kot semantično opisane in z možnostmi kompozicij prevladale nad večjimi in bolj kompleksnimi. Te bodo verjetno še vedno obdržale večji delež izvedenih operacij, saj imajo velik krog stalnih uporabnikov, omogočajo tudi nekatere specifične funkcije, vendar so v tem primeru tudi druge možnosti na voljo. Tukaj je pa izbira v rokah iskalca storitve, ki se bo odločal glede na svoje potrebe in zmožnosti.

Organizacijsko okolje za izvajanje takšne uporabe semantičnih spletnih storitev je, kot že v uvodu omenjeno, Virtualna organizacija. Eden projektov, katerega cilj je vzpostaviti ogrodje za vzpostavitev gradbene VO in ki bo podpiralo in izkoriščalo prednosti semantičnih spletnih storitev je Inteligrid (http://www.inteligrid.com). V enem izmed svojih scenarijev predvideva, kako bo potekala izdelava načrta večnadstropne stavbe s strani podjetja, ki naj bi ta projekt naredilo.



Slika 38: Prijava v virtualno organizacijo InteliGrid

Vir: [http://www.inteligrid.com]

Podjetje bo za izvedbo projekta najprej ustanovilo virtualno podjetje na Inteligrid ogrodju, preko katerega bo izpeljalo izdelavo celotnega načrta. Ogrodje je namenjeno gradbeni VO in lahko vsebuje vse standardizirane koncepte, zapisane v ontologiji gradbene stroke. Seveda lahko zaradi prekompleksnosti temelji na posameznih manjših, bolj specializiranih ontologijah, ki opisujejo koncepte posameznih področij (npr. jeklene konstrukcije. Podjetje po prijavi in definiranju vseh zahtev za izdelavo načrta sproži iskanje agenta po imeniku storitev, ki so registrirane in delujejo v okviru ogrodja. Agent poišče storitve, ki ustrezajo podanim zahtevam, prav tako pa poizkuša ustvariti kombinacije več manjših storitev, katerih kompozicija v delovni proces bi zadostila potrebam uporabnika. Poleg vseh možnosti so kot rezultat iskanja prikazane tudi informacije glede cene uporabe, količin potrebnih računalniških virov, časa izvajanja... Te informacije morajo obsegati vse, kar lahko vpliva na odločitev podjetja, katero storitev ali kombinacijo storitev bo uporabilo. Tukaj se prikažejo velike prednosti VO in ogrodja, ko lahko podjetje za svojo analizo (seveda glede na potrebe) izbira tudi med zelo dragimi programskimi opremami, katerih nakupa si samo ne more privoščiti. Na drugi strani si ponudnik te opreme s takšnim načinom izposoje lahko povrne del investicije. Hkrati se podjetje lahko odloči tudi glede natančnosti analize, saj se potrebe po računalniškem procesiranju z višanjem stopnje natančnosti navadno eksponentno povečujejo.

Preko ogrodja je mogoče dobiti v uporabo velike količine računalniških virov, saj ima ogrodje s svojim načinom delovanja zelo velike zmogljivosti. Če je priključenih veliko število računalnikov, so lahko te zmogljivosti večje tudi od superračunalnikov. Bolj kompleksne analize lahko z zmogljivostmi navadnega računalnika trajajo več dni, kar je v današnjem času običajno nesprejemljivo. Analogno s primerom drage programske opreme so superračunalniki navadno tudi vključeni v takšne sisteme in so dani v uporabo, spet z namenom delnega povračila zelo velike investicije. S takšnim deljenjem virov bi se lahko bistveno povečala kakovost izdelave analiz, ki danes trpi večinoma zaradi pomanjkanja denarja za vrhunsko programsko opremo in časa, ki bi bil potreben za bolj natančne analize.
Po izbiri ustrezne storitve podjetje kot uporabnik po navodilih poda storitvi vhodne podatke, ki ob predpisani interakciji vrne rezultate analize. Celoten proces izbire in uporabe vključno z vsemi izmenjanimi podatki je zavarovan pred morebitnimi zlorabami informacij, kar je tudi ena izmed primarnih zahtev uporabnikov do ogrodja. Obveznosti obeh stran so za uporabo storitve prepisane v pogodbi SLA (*Service Level Agreement*). Ko so naloge in obveznosti vseh vpletenih subjektov izpolnjene, se lahko podjetje s končano analizo odjavi iz sistema. V primeru, da je bila naloga podjetja samo izdelava načrta bodočega objekta, lahko podjetje zapre virtualno podjetje. Vse skupaj poteka po navodilih, ki jih določi administrator Inteligrid ogrodja in ki beleži ter hrani podatke o vseh aktivnostih, ki so bile izvedene. V primeru, da bi bili podatki načrta še kdaj potrebni (npr. študija načrta za potrebe projektiranja podobnega objekta, revizija načrtovanja v primeru porušitve…), so ti podatki v skladu s pravicami dostopni v arhivu ogrodja.

Kot ogrodje bo Inteligrid gostil ogrodne storitve (*grid services*), kar pomeni, da bi bilo potrebno semantične spletne storitve še dodatno nekoliko modificirati. Ogrodne storitve imajo še nakaj zahtev za izvajanje, omogočajo pa še nekatere dodatne možnosti. So pa semantične storitve še vedno osnova celotnega koncepta, tako tudi ogrodnih storitev.

7 ZAKLJUČEK

Semantične spletne storitve so tehnologija prihodnosti, saj izkoriščajo koncepte, ki so se izkazali za uspešne in se vse bolj uveljavljajo v splošni uporabi (standardne spletne storitve), hkrati pa jih nadgrajujejo s koncepti boljše računalniške operabilnosti, ki bodo prisotne v naslednji razvojni stopnji interneta – semantičnemu spletu. V diplomski nalogi so opisane tri trenutno najbolj najbolj razvite metode semantičnega opisa in sicer OWL-s, WSDL-s in WSMO. Prva dva temeljita na nadgradnji obstoječih konceptov z upoštevanjem predlogov W3C organizacije tudi glede razvoja semantičnega spleta, WSMO pa je zasnovam na čisto novi osnovi, celo brez temeljenja na XML standardu. Ker je XML osnova za vzpostavljanje interoperabilnosti menim, da WSMO zaradi takšnega pristopa ne bo imel večinske podpore v skupnosti semantičnega spleta. V primerjavi med OWL-s in WSDL-s pa glede na stopnjo razvoja, programsko opremo, ki je na voljo za posamezno metodo in podporo pri W3C ter razvijalci semantičnih spletnih storitev pa predvidevam, da je OWL-s korak pred WSDL-s in da bo v prihodnosti najbolj razširjen standard za semantični opis.

Da so pa semantične spletne storitve s svojo obetajočo idejo še vedno v razvoju, se je pokazalo pri iskanju programov, ki bi lokalno simulirali delovanje agentov. Na konkretne rezultate in splošno uporabo bo potrebno počakati še nekaj časa, zato menim, da bo v prihodnosti največ dela na zasnovi in testiranju različnih algoritmov za agente. Šele z njimi bodo semantične spletne storitve lahko prišle izven kroga razvijalcev. Čas bo pokazal, ali je to res ena temeljnih tehnologij informacijske tehnologije prihodnosti, ki bo poleg ostalih področij vplivala tudi na področje gradbeništva.

8 PRILOGE

PRILOGA A: WSDL opis spletne storitve SisFggPredmet

Datoteka SisFggPredmetWSDL.wsdl

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2005 rel. 3 U (http://www.altova.com) by vlado (estudent)
-->
<wsdl:definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:y="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl" xmlns:ns="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:si="http://soapinterop.org/xsd" xmlns:tns="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl" targetNamespace="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl">
      <wsdl:types>
            <xsd:schema targetNamespace="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl">
                  <xsd:import
namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
                  <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/"/>
                  <xsd:complexType name="predmeti">
                        <xsd:all>
                              <xsd:element name="predmet_naziv"
type="xsd:string"/>
                              <xsd:element name="predmet_sifra" type="xsd:int"/>
                        </xsd:all>
                  </xsd:complexType>
                  <xsd:complexType name="PredmetiArray">
                        <xsd:complexContent>
                              <xsd:restriction base="SOAP-ENC:Array">
```

```
                                        <xsd:attribute ref="SOAP-ENC:arrayType"
wsdl:arrayType="tns:predmeti[]"/>
                                    </xsd:restriction>
                                </xsd:complexContent>
                        </xsd:complexType>
                </xsd:schema>
        </wsdl:types>
        <message name="iskani_studij_letnik_zahteva">
                <part name="studij" type="xsd:string"/>
        </message>
        <message name="predmeti_za_letnik_odgovor">
                <part name="predmeti" type="tns:PredmetiArray"/>
        </message>
        <message name="predmet_naziv_zahteva">
                <part name="predmet" type="xsd:string"/>
        </message>
        <message name="profesor_predmeta_odgovor">
                <part name="profesor" type="xsd:string"/>
        </message>
        <wsdl:message name="izpitni_roki_zahteva">
                <wsdl:part name="predmet" type="xs:string"/>
        </wsdl:message>
        <wsdl:message name="izpitni_roki_odgovor">
                <wsdl:part name="izpitni_roki" type="xs:string"/>
        </wsdl:message>
        <wsdl:message name="izberi_predmet_zahteva">
                <wsdl:part name="sifra_predmeta" element="" type="xs:integer"/>
        </wsdl:message>
        <wsdl:message name="izberi_predmet_odgovor">
                <wsdl:part name="izbrani_predmet" element="" type="xs:integer"/>
        </wsdl:message>
        <portType name="SisFggPredmet_PortType">
                <wsdl:operation name="izberi_predmet_operacija">
                        <wsdl:input message="y:izberi_predmet_zahteva"/>
                        <wsdl:output message="y:izberi_predmet_odgovor"/>
                </wsdl:operation>
                <wsdl:operation name="izpitni_roki_predmeta_operacija">
                        <wsdl:input message="y:izpitni_roki_zahteva"/>
                        <wsdl:output message="y:izpitni_roki_odgovor"/>
                </wsdl:operation>
                <operation name="izpisi_predmete_operacija">
                        <input message="tns:iskani_studij_letnik_zahteva"/>
                        <output message="tns:predmeti_za_letnik_odgovor"/>
                </operation>
                <operation name="profesor_predmeta_operacija">
```

```
                    <input message="tns:predmet_naziv_zahteva"/>
                    <output message="tns:profesor_predmeta_odgovor"/>
              </operation>
       </portType>
       <binding name="SisFggPredmet_binding" type="tns:SisFggPredmet_PortType">
              <soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>
              <wsdl:operation name="izberi_predmet_operacija">
                    <soap:operation soapAction="urn:#izberi_predmet_operacija"
style="rpc"/>
                    <input>
                          <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="tns:SisFggPredmetService"/>
                    </input>
                    <output>
                          <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="tns:SisFggPredmetService"/>
                    </output>
              </wsdl:operation>
              <wsdl:operation name="izpitni_roki_predmeta_operacija">
                    <soap:operation soapAction="urn:#izpitni_roki_predmeta"/>
                    <input>
                          <soap:body use="literal"/>
                    </input>
                    <output>
                          <soap:body use="literal"/>
                    </output>
              </wsdl:operation>
              <operation name="izpisi_predmete_operacija">
                    <soap:operation
soapAction="tns:SisFggPredmetService#izpisi_predmete_operacija" style="rpc"/>
                    <input>
                          <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="tns:SisFggPredmetService"/>
                    </input>
                    <output>
                          <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="tns:SisFggPredmetService"/>
                    </output>
              </operation>
              <operation name="profesor_predmeta_operacija">
```

```
                    <soap:operation soapAction="http://www.student-
info.net/diploma/SWS-server.php" style="rpc"/>
                    <input>
                          <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="tns:SisFggPredmetService"/>
                    </input>
                    <output>
                          <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="tns:SisFggPredmetService"/>
                    </output>
             </operation>
      </binding>
      <service name="SisFggPredmetService">
             <port name="SisFggPredmet_Port" binding="tns:SisFggPredmet_binding">
                    <soap:address location="http://www.student-info.net/diploma/SWS-
server.php"/>
             </port>
      </service>
</wsdl:definitions>
```

PRILOGA B: OWL zapis ontologije ŠIS predmeta

Datoteka sis-predmet-owl.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns="http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
  xml:base="http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="asistent">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="predavatelj"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="profesor">
    <rdfs:subClassOf rdf:resource="#predavatelj"/>
  </owl:Class>
  <owl:Class rdf:ID="predmet"/>
  <owl:Class rdf:ID="studij"/>
  <owl:Class rdf:ID="izpitni_roki"/>
  <owl:ObjectProperty rdf:ID="ima_profesorja">
    <owl:inverseOf>
      <owl:ObjectProperty rdf:ID="predava"/>
    </owl:inverseOf>
    <rdfs:range rdf:resource="#profesor"/>
    <rdfs:domain rdf:resource="#predmet"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="ima_predmete">
    <owl:inverseOf>
      <owl:ObjectProperty rdf:ID="je_v_studiju"/>
    </owl:inverseOf>
    <rdfs:domain rdf:resource="#studij"/>
    <rdfs:range rdf:resource="#predmet"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#predava">
    <rdfs:range rdf:resource="#predmet"/>
    <rdfs:domain rdf:resource="#profesor"/>
```

```
    <owl:inverseOf rdf:resource="#ima_profesorja"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="ima_predmet">
    <rdfs:domain rdf:resource="#izpitni_roki"/>
    <owl:inverseOf>
      <owl:ObjectProperty rdf:ID="ima_izpitni_rok"/>
    </owl:inverseOf>
    <rdfs:range rdf:resource="#predmet"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#ima_izpitni_rok">
    <owl:inverseOf rdf:resource="#ima_predmet"/>
    <rdfs:range rdf:resource="#izpitni_roki"/>
    <rdfs:domain rdf:resource="#predmet"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#je_v_studiju">
    <rdfs:range rdf:resource="#studij"/>
    <rdfs:domain rdf:resource="#predmet"/>
    <owl:inverseOf rdf:resource="#ima_predmete"/>
  </owl:ObjectProperty>
  <owl:DatatypeProperty rdf:ID="studij_naslov">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#studij"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="sifra_izpitnega_roka">
    <rdfs:domain rdf:resource="#izpitni_roki"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="predavatelj_priimek">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#predavatelj"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="predmet_ime">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#predmet"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="predavatelj_ime">
    <rdfs:domain rdf:resource="#predavatelj"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="studij_letnik">
    <rdfs:domain rdf:resource="#studij"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="datum_izpitnega_roka">
    <rdfs:domain rdf:resource="#izpitni_roki"/>
```

```
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="predmet_sifra">
    <rdfs:domain rdf:resource="#predmet"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
  </owl:DatatypeProperty>
  <studij rdf:ID="gruni2">
    <studij_naslov rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >gradbenistvo_univerzitetni_studij_2</studij_naslov>
    <studij_letnik rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >2</studij_letnik>
    <ima_predmete>
      <predmet rdf:ID="predmet_413">
        <predmet_sifra rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >413</predmet_sifra>
        <je_v_studiju rdf:resource="#gruni2"/>
        <predmet_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Fizika_2</predmet_ime>
      </predmet>
    </ima_predmete>
  </studij>
  <izpitni_roki rdf:ID="izpitni_roki_1">
    <sifra_izpitnega_roka rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >1</sifra_izpitnega_roka>
    <datum_izpitnega_roka rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >15.9.2005</datum_izpitnega_roka>
    <ima_predmet>
      <predmet rdf:ID="predmet_405">
        <predmet_sifra rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >405</predmet_sifra>
        <predmet_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Linearna_algebra</predmet_ime>
        <je_v_studiju>
          <studij rdf:ID="gruni1">
            <ima_predmete rdf:resource="#predmet_405"/>
            <ima_predmete>
              <predmet rdf:ID="predmet_401">
                <predmet_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >Fizika_1</predmet_ime>
                <ima_profesorja>
                  <profesor rdf:ID="Ales_Zaloznik">
                    <predava rdf:resource="#predmet_401"/>
                    <predavatelj_priimek
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                    >Zaloznik</predavatelj_priimek>
```

```
                    <predavatelj_ime
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                    >Ales</predavatelj_ime>
                  </profesor>
                </ima_profesorja>
                <predmet_sifra rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                >401</predmet_sifra>
                <je_v_studiju rdf:resource="#gruni1"/>
              </predmet>
            </ima_predmete>
            <studij_letnik rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >1</studij_letnik>
            <studij_naslov rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >gradbenistvo_univerzitetni_studij_1</studij_naslov>
          </studij>
        </je_v_studiju>
        <ima_izpitni_rok rdf:resource="#izpitni_roki_1"/>
      </predmet>
    </ima_predmet>
  </izpitni_roki>
  <asistent rdf:ID="Marjeta_Kramar">
    <predavatelj_priimek rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Kramar</predavatelj_priimek>
    <predavatelj_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Marjeta_Kramar</predavatelj_ime>
  </asistent>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 2.1, Build 284)
http://protege.stanford.edu -->
```

PRILOGA C: OWL-s ontologija spletne storitve SisFggPredmet

Celotna OWL-s ontologija spletne storitve s štirimi osnovnimi in enim sestavljenim procesom, izdelana po natačni metodi

datoteka: OWL-s-primer1.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
    xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:time="http://www.isi.edu/~pan/damltime/time-entry.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:expr="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
    xmlns="http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
  xml:base="http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://www.daml.org/services/owl-s/1.1/Grounding.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/rules/proposal/swrlb.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/rules/proposal/swrl.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/services/owl-s/1.1/Profile.owl"/>
  </owl:Ontology>
  <owl:Class rdf:ID="asistent">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="predavatelj"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="profesor">
    <rdfs:subClassOf rdf:resource="#predavatelj"/>
  </owl:Class>
  <owl:Class rdf:ID="predmet"/>
```

```
<owl:Class rdf:ID="studij"/>
<owl:Class rdf:ID="izpitni_roki"/>
<owl:ObjectProperty rdf:ID="ima_profesorja">
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="predava"/>
  </owl:inverseOf>
  <rdfs:range rdf:resource="#profesor"/>
  <rdfs:domain rdf:resource="#predmet"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="ima_predmete">
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="je_v_studiju"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#studij"/>
  <rdfs:range rdf:resource="#predmet"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#predava">
  <rdfs:range rdf:resource="#predmet"/>
  <rdfs:domain rdf:resource="#profesor"/>
  <owl:inverseOf rdf:resource="#ima_profesorja"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="ima_predmet">
  <rdfs:domain rdf:resource="#izpitni_roki"/>
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="ima_izpitni_rok"/>
  </owl:inverseOf>
  <rdfs:range rdf:resource="#predmet"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#ima_izpitni_rok">
  <owl:inverseOf rdf:resource="#ima_predmet"/>
  <rdfs:range rdf:resource="#izpitni_roki"/>
  <rdfs:domain rdf:resource="#predmet"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#je_v_studiju">
  <rdfs:domain rdf:resource="#predmet"/>
  <rdfs:range rdf:resource="#studij"/>
  <owl:inverseOf rdf:resource="#ima_predmete"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID="predavatelj_priimek">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#predavatelj"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="sifra_izpitnega_roka">
  <rdfs:domain rdf:resource="#izpitni_roki"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
```

```
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="studij_naslov">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#studij"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="predmet_ime">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#predmet"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="predavatelj_ime">
  <rdfs:domain rdf:resource="#predavatelj"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="studij_letnik">
  <rdfs:domain rdf:resource="#studij"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="datum_izpitnega_roka">
  <rdfs:domain rdf:resource="#izpitni_roki"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="predmet_sifra">
  <rdfs:domain rdf:resource="#predmet"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
</owl:DatatypeProperty>
<studij rdf:ID="gruni2">
  <studij_naslov rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >gradbenistvo_univerzitetni_studij_2</studij_naslov>
  <studij_letnik rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >2</studij_letnik>
  <ima_predmete>
    <predmet rdf:ID="predmet_413">
      <predmet_sifra rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >413</predmet_sifra>
      <je_v_studiju rdf:resource="#gruni2"/>
      <predmet_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Fizika_2</predmet_ime>
    </predmet>
  </ima_predmete>
</studij>
<process:ValueOf rdf:ID="ValueOf_29">
  <process:fromProcess>
    <process:Perform rdf:ID="izberi_predmet_perform">
      <process:process>
        <process:AtomicProcess rdf:ID="izberi_predmet_process">
```

```
            <process:hasInput>
              <process:Input rdf:ID="sifra_predmeta_input">
                <process:parameterType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                >http://www.w3.org/2001/XMLSchema#integer</process:parameterType>
              </process:Input>
            </process:hasInput>
            <process:hasOutput>
              <process:Output rdf:ID="izbrani_predmet_output">
                <process:parameterType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                >http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl#predmet</process:parameterType>
              </process:Output>
            </process:hasOutput>
          </process:AtomicProcess>
        </process:process>
      </process:Perform>
    </process:fromProcess>
    <process:theVar rdf:resource="#izbrani_predmet_output"/>
  </process:ValueOf>
  <grounding:WsdlOutputMessageMap rdf:ID="WsdlOutputMessageMap_48">
    <grounding:owlsParameter>
      <process:Output rdf:ID="izpitni_roki_output">
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.w3.org/2001/XMLSchema#string</process:parameterType>
      </process:Output>
    </grounding:owlsParameter>
    <grounding:wsdlMessagePart rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#izpitni_roki</grounding:wsdlMessagePart>
  </grounding:WsdlOutputMessageMap>
  <process:Input rdf:ID="Input_15"/>
  <grounding:WsdlInputMessageMap rdf:ID="WsdlInputMessageMap_44">
    <grounding:owlsParameter>
      <process:Input rdf:ID="studij_input">
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl#studij</process:parameterType>
      </process:Input>
    </grounding:owlsParameter>
    <grounding:wsdlMessagePart rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#studij</grounding:wsdlMessagePart>
  </grounding:WsdlInputMessageMap>
```

```
  <process:AtomicProcess rdf:ID="izpisi_predmete_process">
    <process:hasOutput>
      <process:Output rdf:ID="predmeti_output">
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.w3.org/2001/XMLSchema#anyURI</process:parameterType>
      </process:Output>
    </process:hasOutput>
    <process:hasInput rdf:resource="#studij_input"/>
  </process:AtomicProcess>
  <process:Input rdf:ID="Input_3"/>
  <grounding:WsdlAtomicProcessGrounding
rdf:ID="izberi_predmet_WsdlAtomicProcessGrounding">
    <grounding:wsdlOperation>
      <grounding:WsdlOperationRef rdf:ID="izberi_predmet_WsdlOperationRef">
        <grounding:portType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#SisFggPredmet_PortType</grounding:portType>
        <grounding:operation rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#izberi_predmet_operacija</grounding:operation>
      </grounding:WsdlOperationRef>
    </grounding:wsdlOperation>
    <grounding:wsdlOutput>
      <grounding:WsdlOutputMessageMap rdf:ID="WsdlOutputMessageMap_43">
        <grounding:owlsParameter rdf:resource="#izbrani_predmet_output"/>
        <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#izbrani_predmet</grounding:wsdlMessagePart>
      </grounding:WsdlOutputMessageMap>
    </grounding:wsdlOutput>
    <grounding:wsdlInput>
      <grounding:WsdlInputMessageMap rdf:ID="WsdlInputMessageMap_42">
        <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#sifra_predmeta</grounding:wsdlMessagePart>
        <grounding:owlsParameter>
          <process:Input rdf:ID="predmet_input">
            <process:parameterType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
            >http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl#predmet</process:parameterType>
          </process:Input>
        </grounding:owlsParameter>
```

```
      </grounding:WsdlInputMessageMap>
    </grounding:wsdlInput>
    <grounding:owlsProcess rdf:resource="#izberi_predmet_process"/>
  </grounding:WsdlAtomicProcessGrounding>
  <process:Perform rdf:ID="izpisi_predmete_perform">
    <process:process rdf:resource="#izpisi_predmete_process"/>
  </process:Perform>
  <predmet rdf:ID="predmet_401">
    <predmet_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Fizika_1</predmet_ime>
    <ima_profesorja>
      <profesor rdf:ID="Ales_Zaloznik">
        <predava rdf:resource="#predmet_401"/>
        <predavatelj_priimek rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Zaloznik</predavatelj_priimek>
        <predavatelj_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Ales</predavatelj_ime>
      </profesor>
    </ima_profesorja>
    <predmet_sifra rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >401</predmet_sifra>
    <je_v_studiju>
      <studij rdf:ID="gruni1">
        <ima_predmete>
          <predmet rdf:ID="predmet_405">
            <predmet_sifra rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >405</predmet_sifra>
            <predmet_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >Linearna_algebra</predmet_ime>
            <je_v_studiju rdf:resource="#gruni1"/>
            <ima_izpitni_rok>
              <izpitni_roki rdf:ID="izpitni_roki_1">
                <sifra_izpitnega_roka
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                >1</sifra_izpitnega_roka>
                <datum_izpitnega_roka
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >15.9.2005</datum_izpitnega_roka>
                <ima_predmet rdf:resource="#predmet_405"/>
              </izpitni_roki>
            </ima_izpitni_rok>
          </predmet>
        </ima_predmete>
        <studij_letnik rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >1</studij_letnik>
```

```
        <ima_predmete rdf:resource="#predmet_401"/>
        <studij_naslov rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >gradbenistvo_univerzitetni_studij_1</studij_naslov>
      </studij>
    </je_v_studiju>
  </predmet>
  <process:Perform rdf:ID="profesor_predmeta_perform">
    <process:hasDataFrom>
      <process:InputBinding rdf:ID="InputBinding_28">
        <process:valueSource rdf:resource="#ValueOf_29"/>
        <process:toParam rdf:resource="#predmet_input"/>
      </process:InputBinding>
    </process:hasDataFrom>
    <process:process>
      <process:AtomicProcess rdf:ID="profesor_predmeta_process">
        <process:hasOutput>
          <process:Output rdf:ID="profesor_output">
            <process:parameterType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
            >http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl#profesor</process:parameterType>
          </process:Output>
        </process:hasOutput>
        <process:hasInput rdf:resource="#predmet_input"/>
      </process:AtomicProcess>
    </process:process>
  </process:Perform>
  <profile:Profile rdf:ID="sis_fgg_predmet_profile">
    <service:presentedBy>
      <service:Service rdf:ID="sis_fgg_predmet_service">
        <service:supports>
          <grounding:WsdlGrounding rdf:ID="sis_fgg_predmet_grounding">
            <grounding:hasAtomicProcessGrounding>
              <grounding:WsdlAtomicProcessGrounding
rdf:ID="izpitni_roki_WsdlAtomicProcessGrounding">
                <grounding:wsdlOperation>
                  <grounding:WsdlOperationRef rdf:ID="izpitni_roki_WsdlOperationRef">
                    <grounding:portType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#SisFggPredmet_PortType</grounding:portType>
                    <grounding:operation
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#izpitni_roki_operacija</grounding:operation>
```

```
                </grounding:WsdlOperationRef>
              </grounding:wsdlOperation>
              <grounding:owlsProcess>
                <process:AtomicProcess rdf:ID="izpitni_roki_predmeta_process">
                  <process:hasInput rdf:resource="#predmet_input"/>
                  <process:hasOutput rdf:resource="#izpitni_roki_output"/>
                </process:AtomicProcess>
              </grounding:owlsProcess>
              <grounding:wsdlInput>
                <grounding:WsdlInputMessageMap rdf:ID="WsdlInputMessageMap_47">
                  <grounding:owlsParameter rdf:resource="#predmet_input"/>
                  <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                  >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#predmet</grounding:wsdlMessagePart>
                </grounding:WsdlInputMessageMap>
              </grounding:wsdlInput>
              <grounding:wsdlOutput rdf:resource="#WsdlOutputMessageMap_48"/>
            </grounding:WsdlAtomicProcessGrounding>
          </grounding:hasAtomicProcessGrounding>
          <grounding:hasAtomicProcessGrounding>
            <grounding:WsdlAtomicProcessGrounding
rdf:ID="izpisi_predmete_WsdlAtomicProcessGrounding">
              <grounding:wsdlInput rdf:resource="#WsdlInputMessageMap_44"/>
              <grounding:wsdlOutput>
                <grounding:WsdlOutputMessageMap rdf:ID="WsdlOutputMessageMap_45">
                  <grounding:owlsParameter rdf:resource="#predmeti_output"/>
                  <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                  >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#predmeti</grounding:wsdlMessagePart>
                </grounding:WsdlOutputMessageMap>
              </grounding:wsdlOutput>
              <grounding:wsdlOperation>
                <grounding:WsdlOperationRef
rdf:ID="izpisi_predmete_WsdlOperationRef">
                  <grounding:portType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                  >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#SisFggPredmet_PortType</grounding:portType>
                  <grounding:operation
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                  >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#izpisi_predmete_operacija</grounding:operation
>
```

```
                </grounding:WsdlOperationRef>
              </grounding:wsdlOperation>
              <grounding:owlsProcess rdf:resource="#izpisi_predmete_process"/>
            </grounding:WsdlAtomicProcessGrounding>
          </grounding:hasAtomicProcessGrounding>
          <service:supportedBy rdf:resource="#sis_fgg_predmet_service"/>
          <grounding:hasAtomicProcessGrounding>
            <grounding:WsdlAtomicProcessGrounding
rdf:ID="profesor_predmeta_WsdlAtomicProcessGrounding">
              <grounding:wsdlOutput>
                <grounding:WsdlOutputMessageMap rdf:ID="WsdlOutputMessageMap_51">
                  <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                  >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#profesor</grounding:wsdlMessagePart>
                  <grounding:owlsParameter rdf:resource="#profesor_output"/>
                </grounding:WsdlOutputMessageMap>
              </grounding:wsdlOutput>
              <grounding:owlsProcess rdf:resource="#profesor_predmeta_process"/>
              <grounding:wsdlOperation>
                <grounding:WsdlOperationRef
rdf:ID="profesor_predmeta_WsdlOperationRef">
                  <grounding:portType
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                  >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#SisFggPredmet_PortType</grounding:portType>
                  <grounding:operation
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                  >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#profesor_predmeta_operacija</grounding:operati
on>
                </grounding:WsdlOperationRef>
              </grounding:wsdlOperation>
              <grounding:wsdlInput>
                <grounding:WsdlInputMessageMap rdf:ID="WsdlInputMessageMap_50">
                  <grounding:owlsParameter rdf:resource="#predmet_input"/>
                  <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
                  >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#predmet</grounding:wsdlMessagePart>
                </grounding:WsdlInputMessageMap>
              </grounding:wsdlInput>
            </grounding:WsdlAtomicProcessGrounding>
          </grounding:hasAtomicProcessGrounding>
```

```
            <grounding:hasAtomicProcessGrounding
rdf:resource="#izberi_predmet_WsdlAtomicProcessGrounding"/>
          </grounding:WsdlGrounding>
        </service:supports>
        <service:describedBy>
          <process:CompositeProcess rdf:ID="profesor_composite_process">
            <service:describes rdf:resource="#sis_fgg_predmet_service"/>
            <process:composedOf>
              <process:Sequence rdf:ID="Sequence_19">
                <process:components>
                  <process:ControlConstructList rdf:ID="ControlConstructList_21">
                    <list:first rdf:resource="#izpisi_predmete_perform"/>
                    <list:rest>
                      <process:ControlConstructList rdf:ID="ControlConstructList_23">
                        <list:rest>
                          <process:ControlConstructList
rdf:ID="ControlConstructList_25">
                            <list:first rdf:resource="#profesor_predmeta_perform"/>
                            <list:rest rdf:resource="http://www.daml.org/services/owl-
s/1.1/generic/ObjectList.owl#nil"/>
                          </process:ControlConstructList>
                        </list:rest>
                        <list:first rdf:resource="#izberi_predmet_perform"/>
                      </process:ControlConstructList>
                    </list:rest>
                  </process:ControlConstructList>
                </process:components>
              </process:Sequence>
            </process:composedOf>
          </process:CompositeProcess>
        </service:describedBy>
        <service:presents rdf:resource="#sis_fgg_predmet_profile"/>
      </service:Service>
    </service:presentedBy>
    <profile:hasInput rdf:resource="#predmet_input"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Storitev SisFggPredmet je namenjena pridobivanju podatkov o predmetih iz baze
Študentskega informacijskega sistema. </rdfs:comment>
    <profile:hasOutput rdf:resource="#izpitni_roki_output"/>
    <profile:hasOutput rdf:resource="#profesor_output"/>
    <profile:has_process rdf:resource="#profesor_composite_process"/>
    <profile:hasOutput rdf:resource="#predmeti_output"/>
    <profile:hasInput rdf:resource="#sifra_predmeta_input"/>
    <profile:hasOutput rdf:resource="#izbrani_predmet_output"/>
    <profile:hasInput rdf:resource="#studij_input"/>
```

```
  </profile:Profile>
  <grounding:WsdlAtomicProcessGrounding rdf:ID="WsdlAtomicProcessGrounding_31"/>
  <asistent rdf:ID="Marjeta_Kramar">
    <predavatelj_priimek rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Kramar</predavatelj_priimek>
    <predavatelj_ime rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Marjeta_Kramar</predavatelj_ime>
  </asistent>
  <process:Input rdf:ID="predmeti_ccdw"/>
  <rdf:Description rdf:about="http://www.daml.org/services/owl-
s/1.1/generic/Expression.owl#AlwaysTrue">
    <expr:expressionLanguage rdf:resource="http://www.daml.org/services/owl-
s/1.1/generic/Expression.owl#SWRL"/>
    <expr:expressionBody rdf:parseType="Literal"><swrl:AtomList
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:swrl="http://www.w3.org/2003/11/swrl#" rdf:about="http://www.w3.org/1999/02/22-
rdf-syntax-ns#nil"></swrl:AtomList>
  </expr:expressionBody>
  </rdf:Description>
  <process:Input rdf:ID="Input_11"/>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 2.1, Build 284)
http://protege.stanford.edu -->
```

PRILOGA D: OWL-s ontologija avtomatizirane OWL-s metode

Celotna OWL-s ontologija storitve s samo eno operacijo – profesor_predmeta_operacija datoteka OWL-s-primer2.owl

---

```
<?xml version="1.0" encoding="WINDOWS-1250"?>
<rdf:RDF
    xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
    xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
    xmlns="http://www.example.org/owls/profesor_predmeta_operacija.owl#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:expression="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
    xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
    xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
  xml:base="http://www.example.org/owls/profesor_predmeta_operacija.owl#">
  <service:Service rdf:ID="profesor_predmeta_operacijaService">
    <service:presents>
      <profile:Profile rdf:ID="profesor_predmeta_operacijaProfile"/>
    </service:presents>
    <service:describedBy>
      <process:AtomicProcess rdf:ID="profesor_predmeta_operacijaProcess"/>
    </service:describedBy>
    <service:supports>
      <grounding:WsdlGrounding rdf:ID="profesor_predmeta_operacijaGrounding"/>
    </service:supports>
  </service:Service>
  <profile:Profile rdf:about="#profesor_predmeta_operacijaProfile">
    <profile:hasOutput>
      <process:Output rdf:ID="profesor">
        <rdfs:label>profesor</rdfs:label>
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.w3.org/2001/XMLSchema#string</process:parameterType>
      </process:Output>
    </profile:hasOutput>
    <profile:textDescription>Auto generated from http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl</profile:textDescription>
```

```
    <service:presentedBy rdf:resource="#profesor_predmeta_operacijaService"/>
    <profile:hasInput>
      <process:Input rdf:ID="predmet">
        <rdfs:label>predmet</rdfs:label>
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.w3.org/2001/XMLSchema#string</process:parameterType>
      </process:Input>
    </profile:hasInput>
    <profile:serviceName>profesor_predmeta_operacija</profile:serviceName>
  </profile:Profile>
  <process:AtomicProcess rdf:about="#profesor_predmeta_operacijaProcess">
    <process:hasInput rdf:resource="#predmet"/>
    <service:describes rdf:resource="#profesor_predmeta_operacijaService"/>
    <process:hasOutput rdf:resource="#profesor"/>
    <rdfs:label>profesor_predmeta_operacijaProcess</rdfs:label>
  </process:AtomicProcess>
  <grounding:WsdlGrounding rdf:about="#profesor_predmeta_operacijaGrounding">
    <grounding:hasAtomicProcessGrounding>
      <grounding:WsdlAtomicProcessGrounding
rdf:ID="profesor_predmeta_operacijaAtomicProcessGrounding"/>
    </grounding:hasAtomicProcessGrounding>
    <service:supportedBy rdf:resource="#profesor_predmeta_operacijaService"/>
  </grounding:WsdlGrounding>
  <grounding:WsdlAtomicProcessGrounding
rdf:about="#profesor_predmeta_operacijaAtomicProcessGrounding">
    <grounding:wsdlInput>
      <grounding:WsdlInputMessageMap>
        <grounding:owlsParameter rdf:resource="#predmet"/>
        <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#predmet</grounding:wsdlMessagePart>
      </grounding:WsdlInputMessageMap>
    </grounding:wsdlInput>
    <grounding:owlsProcess rdf:resource="#profesor_predmeta_operacijaProcess"/>
    <grounding:wsdlInputMessage rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#predmet_naziv_zahteva</grounding:wsdlInputMess
age>
    <grounding:wsdlDocument rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl</grounding:wsdlDocument>
    <grounding:wsdlOutputMessage
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
```

```
    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#profesor_predmeta_odgovor</grounding:wsdlOutpu
tMessage>
    <grounding:wsdlOperation>
      <grounding:WsdlOperationRef>
        <grounding:portType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#SisFggPredmet_Port</grounding:portType>
        <grounding:operation rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#profesor_predmeta_operacija</grounding:operati
on>
      </grounding:WsdlOperationRef>
    </grounding:wsdlOperation>
    <grounding:wsdlOutput>
      <grounding:WsdlOutputMessageMap>
        <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#profesor</grounding:wsdlMessagePart>
        <grounding:owlsParameter rdf:resource="#profesor"/>
      </grounding:WsdlOutputMessageMap>
    </grounding:wsdlOutput>
  </grounding:WsdlAtomicProcessGrounding>
</rdf:RDF>
```

PRILOGA E: Grounding avtomatizirane metode izdelave OWL-s z označenimi povezavami na WSDL dokument

Grounding ontologija za prikaz povezav OWL-s ontologije na WSDL opis storitve datoteka: profesor-predmeta-operacija.owl

---

```
<?xml version="1.0" encoding="WINDOWS-1250"?>
<rdf:RDF
    xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
    xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
    xmlns="http://www.example.org/owls/profesor_predmeta_operacija.owl#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:expression="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
    xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
    xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
  xml:base="http://www.example.org/owls/profesor_predmeta_operacija.owl#">
  <service:Service rdf:ID="profesor_predmeta_operacijaService">
    <service:presents>
      <profile:Profile rdf:ID="profesor_predmeta_operacijaProfile"/>
    </service:presents>
    <service:describedBy>
      <process:AtomicProcess rdf:ID="profesor_predmeta_operacijaProcess"/>
    </service:describedBy>
    <service:supports>
      <grounding:WsdlGrounding rdf:ID="profesor_predmeta_operacijaGrounding"/>
    </service:supports>
  </service:Service>
  <profile:Profile rdf:about="#profesor_predmeta_operacijaProfile">
    <profile:hasOutput>
      <process:Output rdf:ID="profesor">
        <rdfs:label>profesor</rdfs:label>
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.w3.org/2001/XMLSchema#string</process:parameterType>
      </process:Output>
    </profile:hasOutput>
```

```
    <profile:textDescription>Auto generated from http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl</profile:textDescription>
    <service:presentedBy rdf:resource="#profesor_predmeta_operacijaService"/>
    <profile:hasInput>
      <process:Input rdf:ID="predmet">
        <rdfs:label>predmet</rdfs:label>
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.w3.org/2001/XMLSchema#string</process:parameterType>
      </process:Input>
    </profile:hasInput>
    <profile:serviceName>profesor_predmeta_operacija</profile:serviceName>
  </profile:Profile>
  <process:AtomicProcess rdf:about="#profesor_predmeta_operacijaProcess">
    <process:hasInput rdf:resource="#predmet"/>
    <service:describes rdf:resource="#profesor_predmeta_operacijaService"/>
    <process:hasOutput rdf:resource="#profesor"/>
    <rdfs:label>profesor_predmeta_operacijaProcess</rdfs:label>
  </process:AtomicProcess>
  <grounding:WsdlGrounding rdf:about="#profesor_predmeta_operacijaGrounding">
    <grounding:hasAtomicProcessGrounding>
      <grounding:WsdlAtomicProcessGrounding
rdf:ID="profesor_predmeta_operacijaAtomicProcessGrounding"/>
    </grounding:hasAtomicProcessGrounding>
    <service:supportedBy rdf:resource="#profesor_predmeta_operacijaService"/>
  </grounding:WsdlGrounding>
  <grounding:WsdlAtomicProcessGrounding
rdf:about="#profesor_predmeta_operacijaAtomicProcessGrounding">
    <grounding:wsdlInput>
      <grounding:WsdlInputMessageMap>
        <grounding:owlsParameter rdf:resource="#predmet"/>
        <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#predmet</grounding:wsdlMessagePart>
      </grounding:WsdlInputMessageMap>
    </grounding:wsdlInput>
    <grounding:owlsProcess rdf:resource="#profesor_predmeta_operacijaProcess"/>
    <grounding:wsdlInputMessage rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#predmet_naziv_zahteva</grounding:wsdlInputMess
age>
    <grounding:wsdlDocument rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl</grounding:wsdlDocument>
```

```
    <grounding:wsdlOutputMessage
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#profesor_predmeta_odgovor</grounding:wsdlOutpu
tMessage>
    <grounding:wsdlOperation>
      <grounding:WsdlOperationRef>
        <grounding:portType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#SisFggPredmet_Port</grounding:portType>
        <grounding:operation rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#profesor_predmeta_operacija</grounding:operati
on>
      </grounding:WsdlOperationRef>
    </grounding:wsdlOperation>
    <grounding:wsdlOutput>
      <grounding:WsdlOutputMessageMap>
        <grounding:wsdlMessagePart
rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl#profesor</grounding:wsdlMessagePart>
        <grounding:owlsParameter rdf:resource="#profesor"/>
      </grounding:WsdlOutputMessageMap>
    </grounding:wsdlOutput>
  </grounding:WsdlAtomicProcessGrounding>
```

PRILOGA F: Semantično anotiran WSDL (1.1) opis storitve po predlogu OWL-s

Semantično anotiran WSDL opis storitve, ki vsebuje samo eno operacijo (profesor_predmeta_operacija) po predlogu OWL-s

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2005 rel. 3 U (http://www.altova.com) by vlado (estudent) -->
<wsdl:definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:y="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl" xmlns:ns="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:si="http://soapinterop.org/xsd" xmlns:tns="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl" targetNamespace="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl" xmlns:profesor-owl="http://www.student-
info.net/diploma/profesor-predmeta-operacija.owl#" xmlns:owl-s-
wsdl="http://www.daml.org/services/owl-s/wsdl/">
      <message name="predmet_naziv_zahteva">
            <part name="predmet" type="xsd:string" owl-s-wsdl:owl-s-
parameter="profesor-owl:predmet"/>
      </message>
      <message name="profesor_predmeta_odgovor">
            <part name="profesor" type="xsd:string" owl-s-wsdl:owl-s-
parameter="profesor-owl:profesor"/>
      </message>
      <portType name="SisFggPredmet_PortType">
            <operation name="profesor_predmeta_operacija" owl-s-wsdl:owl-s-
process="profesor-owl:profesor_predmeta_operacijaProcess">
                  <input message="tns:predmet_naziv_zahteva"/>
                  <output message="tns:profesor_predmeta_odgovor"/>
            </operation>
      </portType>
      <binding name="SisFggPredmet_binding" type="tns:SisFggPredmet_PortType">
```

```
            <soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>
            <operation name="profesor_predmeta_operacija">
                  <soap:operation soapAction="http://www.student-
info.net/diploma/SWS-server.php" style="rpc"/>
                  <input>
                        <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                        namespace="tns:SisFggPredmetService"
                        encodingStyle="http://www.daml.org/2001/03/"/>
                  </input>
                  <output>
                        <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                        namespace="tns:SisFggPredmetService"
                        encodingStyle="http://www.daml.org/2001/03/"/>
                  </output>
            </operation>
      </binding>
      <service name="SisFggPredmetService">
            <port name="SisFggPredmet_Port" binding="tns:SisFggPredmet_binding">
                  <soap:address location="http://www.student-info.net/diploma/SWS-
server.php"/>
            </port>
      </service>
</wsdl:definitions>
```

PRILOGA G: Semantična anotacija WSDL 1.1 datoteke SisFggPredmet storitve z METEOR-s

---

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:SERVICE_EXTENSION="LSDISExt"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:tns="http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:ns_1="http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:y="http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:si="http://soapinterop.org/xsd"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
targetNamespace="http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl"
xmlns:ns="http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl">
<wsdl:types>

<xsd:schema targetNamespace="http://www.student-
info.net/diploma/SisFggPredmetWSDL.wsdl">
<xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
<xsd:import namespace="http://schemas.xmlsoap.org/wsdl/"/>
<xsd:complexType name="predmeti">
<xsd:all>
<xsd:element name="predmet_naziv" type="xsd:string"/>
<xsd:element name="predmet_sifra" type="xsd:int"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="PredmetiArray">
<xsd:complexContent>
<xsd:restriction base="SOAP-ENC:Array">
<xsd:attribute ref="SOAP-ENC:arrayType" wsdl:arrayType="tns:predmeti[]"/>
```

```
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:schema>
</wsdl:types>

<wsdl:message name="izberi_predmet_zahteva" >
<wsdl:part name="sifra_predmeta"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" LSDISExt:onto-
concept="ns_1:predmet_sifra" type="xs:integer" />
</wsdl:message>
<wsdl:message name="izberi_predmet_odgovor" >
<wsdl:part name="izbrani_predmet"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" LSDISExt:onto-
concept="ns_1:predmet_sifra" type="xs:integer" />
</wsdl:message>
<wsdl:message name="izpitni_roki_zahteva" >
<wsdl:part name="predmet"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" LSDISExt:onto-
concept="ns_1:predmet_sifra" type="xs:string" />
</wsdl:message>
<wsdl:message name="izpitni_roki_odgovor" >
<wsdl:part name="izpitni_roki"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" LSDISExt:onto-
concept="ns_1:izpitni_roki" type="xs:string" />
</wsdl:message>
<wsdl:message name="iskani_studij_letnik_zahteva" >
<wsdl:part name="studij"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" LSDISExt:onto-
concept="ns_1:studij_letnik" type="xsd:string" />
</wsdl:message>
<wsdl:message name="predmeti_za_letnik_odgovor" >
<wsdl:part name="predmeti"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" LSDISExt:onto-
concept="ns_1:predmet" type="tns:PredmetiArray" />
</wsdl:message>
<wsdl:message name="predmet_naziv_zahteva" >
<wsdl:part name="predmet"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" LSDISExt:onto-
concept="ns_1:predmet_sifra" type="xsd:string" />
</wsdl:message>
<wsdl:message name="profesor_predmeta_odgovor" >
<wsdl:part name="profesor"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" LSDISExt:onto-
concept="ns_1:predavatelj_priimek" type="xsd:string" />
```

```
</wsdl:message>

<portType name="SisFggPredmet_PortType"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" >
<wsdl:operation name="izberi_predmet_operacija"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" >
<wsdl:input message="y:izberi_predmet_zahteva"/>
<wsdl:output message="y:izberi_predmet_odgovor"/>
</wsdl:operation>
<wsdl:operation name="izpitni_roki_predmeta_operacija"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" >
<wsdl:input message="y:izpitni_roki_zahteva"/>
<wsdl:output message="y:izpitni_roki_odgovor"/>
</wsdl:operation>
<wsdl:operation name="izpisi_predmete_operacija"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" >
<wsdl:input message="y:iskani_studij_letnik_zahteva"/>
<wsdl:output message="y:predmeti_za_letnik_odgovor"/>
</wsdl:operation>
<wsdl:operation name="profesor_predmeta_operacija"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions" >
<wsdl:input message="y:predmet_naziv_zahteva"/>
<wsdl:output message="y:profesor_predmeta_odgovor"/>
</wsdl:operation>
</portType>
```

PRILOGA H: WSDL-s opis storitve SisFggPredmet

Semantično anotiran WSDL 2.0 dokument (WSDL-s) spletne storitve

datoteka: meteor-s-SisFggPremet-wsdl2-anotiran.wsdl20

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:tns="http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:LSDISExt="http://lsdis.cs.edu/METEORS/WSDLExtensions"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:ns_1="http://www.student-info.net/diploma/diploma-SWS/ontologija-sis-
predmet.owl"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:y="http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:si="http://soapinterop.org/xsd"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
targetNamespace="http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl"
xmlns:ns="http://www.student-info.net/diploma/SisFggPredmetWSDL.wsdl"
>

<interface name="SisFggPredmet_PortType" businessService ="null" domain
="naics:null" location ="iso:null" description ="null" >
<operation name="izberi_predmet_operacija" pattern="mep:in-out" >
<input messageLabel="sifra_predmeta" element="ns_1:predmet_sifra" type="xs:integer"
/>
<output messageLabel="izbrani_predmet" element="ns_1:predmet_sifra"
type="xs:integer" />
</operation>
<operation name="izpitni_roki_predmeta_operacija" pattern="mep:in-out" >
<input messageLabel="predmet" element="ns_1:predmet_sifra" type="xs:string" />
<output messageLabel="izpitni_roki" element="ns_1:izpitni_roki" type="xs:string" />
</operation>
<operation name="izpisi_predmete_operacija" pattern="mep:in-out" >
<input messageLabel="studij" element="ns_1:studij_letnik" type="xsd:string" />
```

```
<output messageLabel="predmeti" element="ns_1:predmet" type="tns:PredmetiArray" />
</operation>
<operation name="profesor_predmeta_operacija" pattern="mep:in-out" >
<input messageLabel="predmet" element="ns_1:predmet_sifra" type="xsd:string" />
<output messageLabel="profesor" element="ns_1:predavatelj_priimek"
type="xsd:string" />
</operation>
</interface>
</definitions>
```

9 VIRI

1) **An OWL-s editor tutorial**

2) Barners-Lee T., **Business Plan for the Semantic Web**, 2001

3) Barners-Lee T., **Weaving the web**,1999

4) Bavec, **Na poti k teoriji virtualnih organizacij**, 2002

5) **Developers guide for METEOR-S Eclipse plugin**

6) Foster, **The anatomy of the grid**, 2001

7) Murn, **Adaptacija klasičnih inženirskih programov v spletne servise**, 2003. Dipl. naloga. Ljubljana. Univerza v Ljubljani, Fakulteta za gradbeništvo in geodezijo

8) Petrinja, Turk, Dolenc, **Uporaba ogrodij za vzpostavljanje virtualnih organizacij**, 2005

9) Saadati S., Denker G., **An OWL-s Editor Tutorial**, Version 1.1

10) **User's guide for MWSAF**

11) Wilson, Harvey, Vankeisbelck, Samad, **Enabling the constructional virtual enterprise: The OSMOS aproach**, 2001

SPLETNE STRANI

12) http://ant.apache.org

13) http://dietrich.ganx4.com/nusoap

14) http://jakarta.apache.org/tomcat

15) http://logicerror.com/semanticWeb-webdev

16) http://lsdis.cs.uga.edu/library/download/WSDL-S-V1.html

17) http://lsdis.cs.uga.edu/projects/meteor-s/

18) http://lsdis.cs.uga.edu/projects/meteor-s/mwsaf/

19) http://projects.semwebcentral.org/projects/owlseditor/

20) http://protege.stanford.edu/plugins/owl

21) http://ws.apache.org/axis

22) http://www.alphaworks.ibm.com/tech/wssem/33/
http://www.cogsci.princeton.edu/~wn/wn2.0.shtml

23) http://www.daml.org/services/owl-s/1.1/owl-s-wsdl.html

24) http://www.developer.com/services/article.php/1602051
25) http://www.ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-noy-mcguinness.html
26) http://www-ksl.stanford.edu/kst/what-is-an-ontology.html
27) http://www.mgateway.com/scripts/mgwms32.dll?MGWLPN=EXTC&wlapp=wsdlValidator&eXtcCalledFrom=MGateway
28) www.mindswap.org/
29) http://www.monitor.si/clanki.php?id=11
30) http://www.research.att.com/sw/tools/graphviz/download.html
31) http://www.scientificamerican.com/print_version.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21
32) http://www.sitepoint.com/article/semantic-web-services
33) http://www.w3.org/2001/sw/
34) http://www.w3.org/2002/ws/Activity
35) http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/
36) http://www.w3.org/2005/04/FSWS/Submissions/17/WSDL-S.htm
37) http://www.w3.org/2005/04/FSWS/Submissions/1/wsmo_position_paper.html
38) http://www.w3.org/TR/owl-features/
39) http://www.wsmo.org/TR/d3/d3.1/v0.1/20050401/
40) http://www.xmethods.com/ve2/Tools.po
41) http://xml.coverpages.org/ni2005-06-16-a.html