KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

Vaidas Nacius

INTERNETINIO SERVISO AUTOMATINIO TESTAVIMO ĮRANKIO KŪRIMAS IR TYRIMAS

Magistro baigiamasis darbas

Vadovas
Prof. dr. Eduardas Bareiša

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

INTERNETINIO SERVISO AUTOMATINIO TESTAVIMO ĮRANKIO KŪRIMAS IR TYRIMAS

Magistro baigiamasis darbas
Programų sistemų inžinerija (kodas 621E16001)

Vadovas
Prof. dr. Eduardas Bareiša

Recenzentas
Lenkt. dr. Dominykas Barisas

Projektą atliko
IFM-3/2 gr. studentas Vaidas Nacius

KAUNAS, 2015

AUTENTIŠKUMO PATVIRTINIMAS

SANTRAUKA

Automatinis web servisų testavimas susideda iš WSDL failų apdorojimo, testų generavimo ir užklausų vykdymo bei rezultatų išvedimo. Prisijungiant prie web serviso atliekama WSDL failo analizė, išanalizavus kūriami testavimo atvejai. Su tais pačiais testiniais duomenimis testuojami visi pasirinkti web servisai ir pagal gautus rezultatus įvertinamas atnaujintasis web servisas.

SUMMARY

Automated web service testing consists of a WSDL file processing, test generation, execution of queries and the derivation of results. Connecting to the web service WSDL file analysis is performed and test cases are created after the analysis. All selected web services are tested with the same test data and according to the results obtained, a new updated web service is assessed.

TURINYS

PAVEIKSLŲ SĄRAŠAS

1. ĮVADAS

Web servisai yra interneto taikomųjų programų rūšis, kurie gali vykdyti tiek paprastas užklausas, tiek sudėtingus procesus. Šiuo metu internetas yra visuotinai paplitęs, o web servisai veikia interneto naršyklėse, todėl galima sakyti, kad web servisais galima naudotis visur, kur tik yra interneto prieiga. Web servisams sparčiai plintant ir juos panaudojant tiek paprastuose, tiek sudėtinguose verslo procesuose atsiranda patikimų web servisų poreikis. Yra įvairių web servisų rūšių ir jie kūriami ant įvairiausių platformų: Net frameworks, Apache Axis java2wsdl, PHP, Oracle ir t.t. [1].

Atnaujinant, perprojektuojant web servisus atsiranda poreikis lyginti seną versiją su nauja, norint užtikrinti naujojo web proceso efektyvumą ir patikimumą [2]. Vienas iš galimų testavimo būdų yra regresinis testavimas, kuris yra skirtas būtent atnaujinamų bei perprojektuojamų sistemų testavimui.

1.1. Problematika ir aktualumas

Praktikoje dėl įvairių priežasčių dažnai tenka perprojektuoti lygtines sistemas. Padėtį apsunkina tai, kad senojoje sistemoje naudotos technologijos gali būti nebepalaikomos, o naujoje sistemos versijoje gali būti naudojamos kitos technologijos, gali pasikeisti duomenų formatai ar atsirasti papildomi funkcionalumai. Tačiau, naujosios sistemos teikiamo funkcionalumo dalis, kuri atitinka seniau veikusios sistemos funkcionalumą turi veikti taip pat.

Perprojektuojant lygtines sistemas gali būti pakeistas vidinių metodų veikimas, perkurta sistemos architektūra, ar net panaudotos visiškai naujos technologijos. Dėl tokių pakeitimų dažnai pasitaiko naujų klaidų, ko pasekoje atsiranda poreikis abiejų sistemų teikiamo funkcionalumo palyginimui, kurį galima atlikti naudojant regresinį testavimą.

Nagrinėjant ir gilinantis į kitų tyrėjų atliktus darbus apie web servisų testavimą galima sakyti, kad dauguma iš jų susikoncentruoja į web serviso kaip vieneto (vienos sistemos) testavimą ir testų automatizavimą [3] [4] [5] [17]. Tačiau yra svarbu pažvelgti į web servisų testavimą ir platesniu kampu, neapsibrėžiant vieno web serviso testavimu, sukurti metodiką, kuri leistų testuoti vienu metu kelis web servisus, dažniausiai du: seną web serviso versiją ir atnaujintą.

1.2. Darbo tikslas

Šio darbo tikslas yra sukurti ir įvertinti metodą, kuris leistų testuoti vienu metu du web servisus: seną web serviso versiją ir atnaujintą.

2. EGZISTUOJANTYS SPRENDIMAI

Web servisai panaudojami komunikacijai tarp organizacijų, organizacijos viduje, tretiesiems asmenims [14]. Rinkoje yra testavimo įrankių skirtų testuoti web servisų funkcionalumus ar jie atlieka numatytas funkcijas, tokių kaip „.NET WebService Studio“, “.NET SOAP Client”, “Soap UI”. Tačiau nepavyko rasti įrankio, kuris sugebėtų sulyginti dviejų web servisų funkcionalumus, bei lygiagrečiai juos testuoti vienu metu (Lentelė 1).

Lentelė 1. Įrankių palyginimas

	.NET WebService Studio	.NET SOAP Client	Soap UI
Funkcijų atvaizdavimas	+	-	+
Užklausų formavimas	+	+	+
XML atvaizdavimas	-	+	+
Detali užklausų analizė	-	-	+
Galimybė testuoti du web servisus vienu metu	-	-	-

3. LITERATŪROS ANALIZĖ

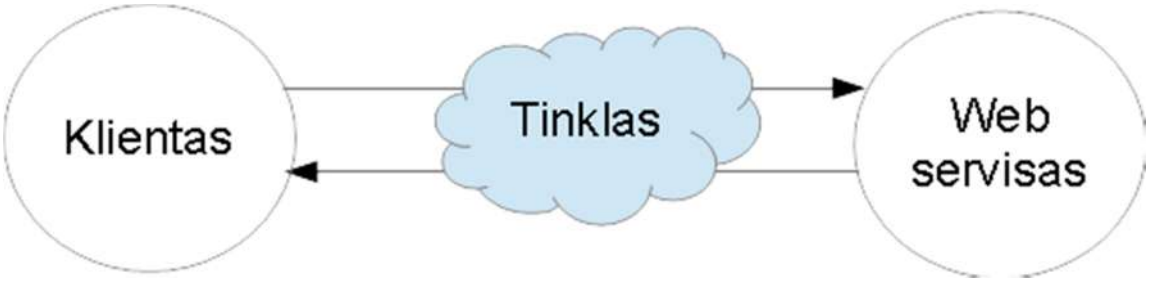
Šiame skyriuje bus apžvelgiami web servisų standartai, programinės įrangos testavimo metodai.

3.1. Web servisai

Pagrindinis reikalavimas web servisų tiekėjui ir serviso užsakovui yra gebėjimas kurti ir sintaksiškai analizuoti SOAP [10] pranešimus bei gebėjimas komunikuoti tinklo pagalba, kitaip sakant apsikeisti XML [11] pranešimais [6]. Paprastai SOAP serveris, veikiantis web aplikacijų serveryje, atlieka minėtas funkcijas. Gali būti naudojama tam tikros programinės kalbos biblioteka, užtikrinanti minėtas funkcijas.

3.1.1. Web servisų standartai

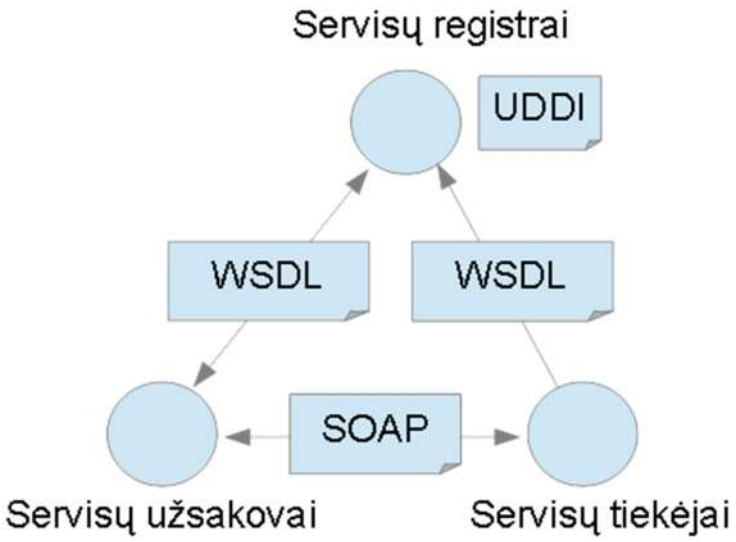
Yra sukurta daug įvairių servisų standartų, bet labiausiai iš jų paplitę yra web servisai, dėl atvirojo kodo ir gausios dokumentacijos. Web serviso sąveika su naudotoju pavaizduota (1 pav.).



1 pav. Web serviso sąveika su naudotoju

Pagrindinės web servisų mainų rolės (2 pav.) [15] [16] yra šios:

- **Servisų tiekėjai (Service Provider)** – publikuojantys tam tikrus servisus bei užtikrinantys jų funkcionalumą.
- **Servisų registrai (Service Broker)** – jie saugo informaciją apie web servisus bei atlieka tarpininkavimą tarp web servisų tiekėju ir web servisų užsakovų. Šios funkcijos gali būti atliekamos tiek vienos tiek ir kelių organizacijų.
- **Servisų užsakovai (Service Requester)** – bendradarbiauja su servisu registrais, ieškodami tinkamų servisų, bei integruodami juos su savo turimais.



2 pav. Web serviso architektūra

3.1.2. SOAP

SOAP pranešimą sudaro trys dalys: vokas, antraštė, vidinė informacija (3 pav.).

3 pav. SOAP pranešimas

SOAP vokas yra pagrindinis ir būtinas kiekvieno SOAP pranešimo elementas. Jis apibrėžia likusioje pranešimo dalyje naudojamų vardų aibę.

SOAP antraštė yra neprivalomas elementas, kuris reikalingas komunikuojančių aplikacijų specifinės pagalbinės informacijos apsikeitimui. Taigi galimi antraštės elementai neapibrėžiami SOAP specifikacijoje. Jeigu SOAP pranešime yra antraštė, tai ji seka iš karto po voko.

SOAP informacinė dalis XML formate atvaizduoja informaciją, kuria keičiamasi. Informacinė dalis yra privaloma SOAP pranešime. Jei nėra antraštės, ši dalis seka iš karto po SOAP voko.

SOAP pranešimo standartinis šablonas XML formatu (4 pav.).

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Header>
...
</soap:Header>

<soap:Body>
...
  <soap:Fault>
  ...
  </soap:Fault>
</soap:Body>

</soap:Envelope>
```

4 pav. SOAP pranešimo šablonas XML formatu

SOAP užklausos (1 pvz.) ir atsakymo (2 pvz.) pavyzdžiai, dviejų sveikųjų skaičių sudėčiai.

1 pavyzdys:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Add xmlns="http://tempuri.org/">
      <num1>int</num1>
      <num2>int</num2>
    </Add>
  </soap:Body>
</soap:Envelope>
```

2 pavyzdys:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AddResponse xmlns="http://tempuri.org/">
      <AddResult>int</AddResult>
    </AddResponse>
  </soap:Body>
</soap:Envelope>
```

3.1.3. WSDL

WSDL yra pagrįsta XML technologija ir leidžia katalogizuoti bei aprašyti web servisus [7]. WSDL yra naudojama komunikavimo proceso tarp web servisų automatizavimui. WSDL aprašo web

servisų interfeisą, prisijungimo prie web servisų protokolą (pvz. SOAP) ir galutinę web servisų būvimo vietą (t.y. nuorodą į web servisą).

WSDL dokumentai web servisų aprašymui naudoja šiuos elementus [8]:

types – konteineriai, skirti duomenų tipų aprašymams, naudojant tam tikrą sistemą (pavyzdžiui XSD).

message – abstraktus komunikuojančių duomenų aprašymas.

operation – abstraktus web serviso atliekamų veiksmų aprašymas (įėjimo išėjimo parametrai).

portType – operacijų, kurias sugeba atlikti vienas ar kitas baigtinis serviso komponentas, abstrakčių aprašymų aibė.

binding – konkreti protokolo ar duomenų formato specifikacija, tam tikram porto tipui (portType).

port – atskiras baigtinis serviso komponentas, apibūdinamas kaip sąryšio ir tinklo adreso kombinacija.

service – aprašo web servisą kaip port ar endpoint elementų kolekciją.

WSDL failo pavyzdys:

```
<?xml version="1.0"?>
<definitions name="StockQuote"

targetNamespace="http://example.com/stockquote.wsdl"
          xmlns:tns="http://example.com/stockquote.wsdl"
          xmlns:xsd1="http://example.com/stockquote.xsd"
          xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
          xmlns="http://schemas.xmlsoap.org/wsdl/">

    <types>
       <schema targetNamespace="http://example.com/stockquote.xsd"
              xmlns="http://www.w3.org/2000/10/XMLSchema">
           <element name="TradePriceRequest">
              <complexType>
                  <all>
                      <element name="tickerSymbol" type="string"/>
                  </all>
              </complexType>
           </element>
           <element name="TradePrice">
              <complexType>
                  <all>
                      <element name="price" type="float"/>
                  </all>
              </complexType>
           </element>
       </schema>
    </types>

    <message name="GetLastTradePriceInput">
        <part name="body" element="xsd1:TradePriceRequest"/>
    </message>

    <message name="GetLastTradePriceOutput">
        <part name="body" element="xsd1:TradePrice"/>
    </message>
```

```
    <portType name="StockQuotePortType">
        <operation name="GetLastTradePrice">
           <input message="tns:GetLastTradePriceInput"/>
           <output message="tns:GetLastTradePriceOutput"/>
        </operation>
    </portType>

    <binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
        <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation name="GetLastTradePrice">
           <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
           <input>
               <soap:body use="literal"/>
           </input>
           <output>
               <soap:body use="literal"/>
           </output>
        </operation>
    </binding>

    <service name="StockQuoteService">
        <documentation>My first service</documentation>
        <port name="StockQuotePort" binding="tns:StockQuoteBinding">
           <soap:address location="http://example.com/stockquote"/>
        </port>
    </service>

</definitions>
```

3.1.4. UDDI

UDDI technologijos pagalba galima surinkti ir saugoti informaciją apie web servisus [9]. Iš esmės UDDI registras, tai tarytum toks pats web servisas, teikiantis struktūrinę informaciją apie kitus web servisus. UDDI registro realizacija turėtu leisti prisijungti prie jo dviem skirtingais būdais:

Prisijungimas naudojant naršyklę per web interfeisą. Toks prisijungimo metodas yra gan paprastas būdas serviso tiekėjui publikuoti informaciją apie jo verslą ir teikiama servisą, o serviso vartotojui leidžia lengvai surasti publikuojamą informaciją.

Programinis prisijungimas naudojant UDDI API specifikaciją. Šis metodas leidžia prisijungti prie UDDI registro kaip prie web serviso ir tuomet informacijos publikavimas arba paieška vyksta dinamiškai, realiu laiku. Programinio prisijungimo metu tam, kad susisiekti su UDDI registru ir atlikti publikavimo ar paieškos operacijas, naudojami atitinkami UDDI API komandiniai pranešimai.

Ši technologija turi tris svarbias savybes, kurių buvo pasigesta iki tol vystytuose iš komponentų susidedančiuose ir interneto technologija paremtuose servisuose:

Standartizuotas ir visiems prieinamas servisų aprašymo mechanizmas;

Paprastas serviso iškvietimo mechanizmas;

Lengvai pasiekiamas pagrindinis servisų informacijos registras.

UDDI specifikacija sudaryta iš kelių susijusių dokumentų ir WSDL kalba parašytos aprašomosios schemos, kur yra apibrėžiamos SOAP protokolu paremtos web servisų registracijos ir atradimo taisyklės.

UDDI registras yra logiškai centralizuotas ir fiziškai paskirstytas servisas, su daugybe šakninių elementų, kurie tarpusavyje apsikeičia duomenimis. Kai tik verslo elementas užregistruoja savo informaciją registre, ji yra automatiškai paskirstoma tarp atskirų šakninių elementų ir tampa visiems plačiai prieinama. UDDI registruose saugoma informacija susideda iš trijų pagrindinių komponentų:

Baltieji puslapiai – pateikiama kompanijos kontaktinė informacija;

Geltonieji puslapiai – kuriuose kompanijų veikla yra suskirstyta pagal standartinę sistematiką;

Žalieji puslapiai – pateikiama konkrečių servisų techninė dokumentacija.

3.2. XML

XML (angl. „Extensible Markup Language“) yra „žymėjimo“ kalba dokumentams, kuriuose saugoma struktūrizuota tekstinė informacija [11]. W3C patvirtino XML standartą dar 1998 metais. Iš esmės tai - seniai pramonėje vartojamos kalbos SGML poaibis, specialiai pritaikytas naudojimui internete. Todėl kiekvienas XML dokumentas yra ir SGML dokumentas.

XML duomenų modelis yra medis, sudarytas iš elementų, atributų ir kitokio tipo mazgų. Norima struktūra išgaunama naudojantis elementais, kurie tekstine XML sintakse atvaizduojami kaip žymės. XML standartas nustato bendras jų sudarymo taisykles, bet neapibrėžia konkretaus elementų rinkinio ar jų reikšmės.

Kiekviena žymė prasideda simboliu „<“ ir baigiasi simboliu „>“. Jų viduje užrašomas elemento vardas ir (galbūt) papildoma informacija, pavyzdžiui, atributai ar vardų sritys. Elementų vardai yra „case-sensitive“ (raidžių dydis svarbus). Yra pradžios bei pabaigos žymės. Pabaigos žymėje prieš jos vardą rašomas simbolis „/“. Elemento viduje taip pat gali būti kiti elementai.

Pavyzdys:

```
<elementas>
  tekstas 1
  <vidinis_elementas>
        tekstas 2
  </vidinis_elementas>
</elementas>
```

3.3. Programinės įrangos testavimo metodai

Programinės įrangos testavimas – procesas, apjungiantis statinius ir dinaminius veiksmus, susijusius su programinės įrangos planavimu, pasiruošimu ir analize. Šie veiksmai leidžia nustatyti, ar programinė įranga atitinka konkrečius jai keliamus reikalavimus, ar atitinka savo paskirtį bei

leidžia išsiaiškinti programinės įrangos defektus. Kitaip tariant programinės įrangos testavimas apima priemones, skirtas programinės įrangos kokybei įvertinti ir užtikrinti [12] [13].

Egzistuoja du pagrindiniai programinės įrangos testavimo būdai:

- Rankinis testavimas (Manual Scripted Testing) – tai seniausiai taikomas testavimo būdas, kada testavimo atvejus planuoja ir aptaria žmonių komanda prieš vykdant testavimą.
- Automatizuotas testavimas (Automated Testing) – kur tam tikri testavimo atvejai atliekami automatizuotai, vedamas klaidų taisymo registras.

Automatizavimo nauda:

- Padidina programinės įrangos patikimumą ir sumažina kainą
- Sumažina žmogiškųjų resursų poreikį
- Sumažina monotoniško darbo kiekį
- Lengvai pakartotinai panaudojamas

3.3.1. Testavimų rūšys

Testavimo rūšys gali būti skirstomos pagal testuojamą objektą arba pagal testavimo tikslą.

Testavimo rūšys pagal objektą:

- Vienetų testavimas
- Integracijos testavimas
- Visos sistemos testavimas

Testavimo rūšys pagal tikslą:

- Priėmimo/parengtumo testavimas (acceptance/qualification)
- Diegimo testavimas (installation)
- Alfa ir beta testavimas
- Atitikimo/funkcionalumo/korektiškumo testavimas
- Regresinis testavimas
- Našumo (performance)
- Stresinis testavimas (stress)
- Sulyginimo testavimas (Back-to-back)
- Atstatymo testavimas (recovery)
- Konfigūracijos testavimas (configuration)
- Naudojamumo testavimas (usability)

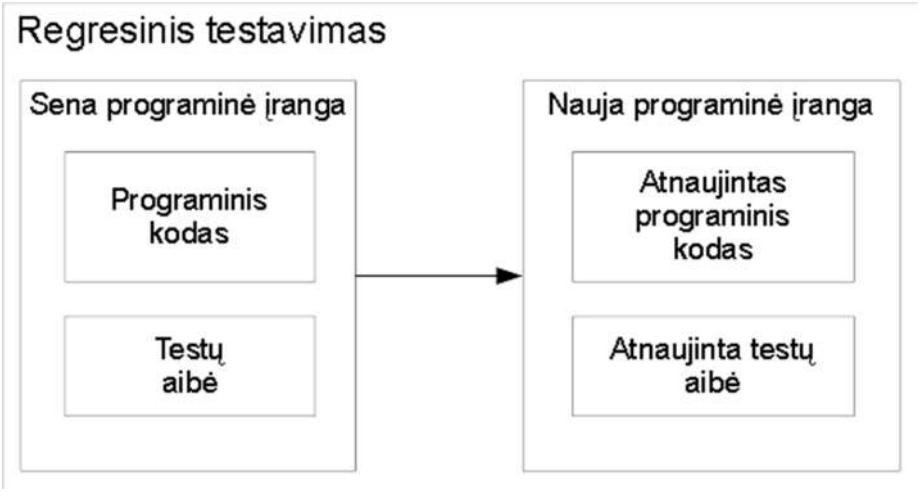
3.3.2. Regresinis testavimas

Regresinis testavimas yra programinės įrangos testavimo tipas, kuriuo siekiama rasti naujos programinės įrangos klaidas po sistemos pakeitimų, atnaujinimų ar konfigūracijos pasikeitimų [2].

Regresinio testavimo pagrindinis tikslas yra patikrinti, ar nauja programinė įranga išlaiko ankstesnėse versijose veikusį funkcionalumą ne tik pasikeitusiose dalyse bei moduliuose, bet ir visą bendrą web serviso funkcionalumą.

Viena iš pagrindinių regresijų atsiradimo priežasčių yra ta, jog programinio kodo dalys yra glaudžiai susiję, todėl programuotojui keičiant vieną modulį, gali būti pažeistas kitas susijęs programinės dalies kodas.

Regresinis testavimas gali būti taikomas bet kuriame testavimo lygyje. Atliekant visus jau taikytus testus, siekiama nustatyti, jog programinės įrangos funkcionalumas nepasikeitė net ir nepakitusiose ir neatnaujintose programos dalyse. Principas gaunasi toks, kad jau sukauptų testų aibę panaudojame pakeistos programinės įrangos testavimui (5 pav.).



5 pav. Regresinis testavimas

3.4. Web servisų klaidos

Nesutampantys tipai – situacija, kai klientas ir servisas skirtingai supranta metodo parametrus. Tokia situacija gali nutikti, jei perprojektuojant web servisą buvo pakeistas parametro tipas.

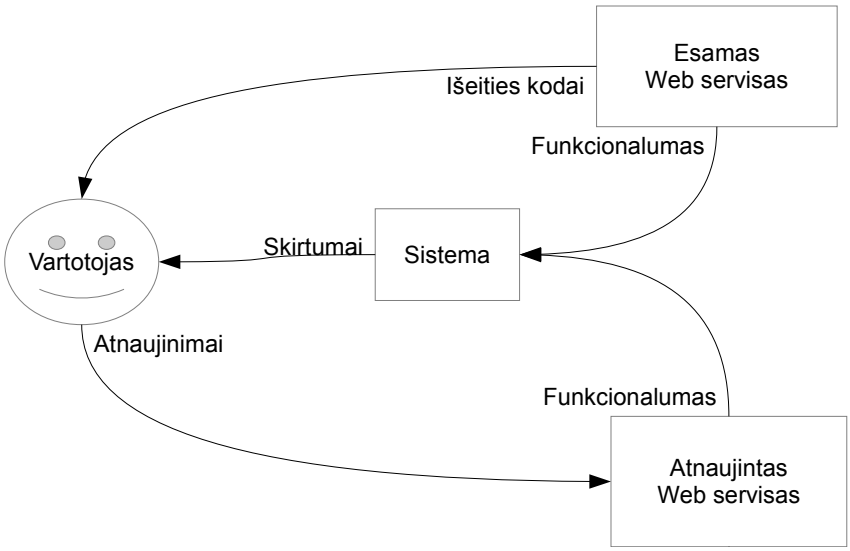
Web serviso metodas neatsako – situacija, kai kreipiantis į web serviso metodą, sistema pakimba ir nesulaukiamas atsakas per nustatytą laiką. Tokia situacija gali susiklostyti, jei perprojektuojant sistemą buvo įvelta klaidų, dėl kurių metodo skaičiavimai yra pernelyg ilgi arba vidiniai skaičiavimai pateko į “amžiną” ciklą.

Pašalintas funkcionalumas – situacija, kai klientas kreipiasi į serviso nebeegzistuojantį metodą.

4. PROJEKTINĖ DALIS

4.1. Veiklos kontekstas

Konteksto diagrama pateikta (6 pav.).



6 pav. Konteksto diagrama

4.2. Veiklos padalinimas

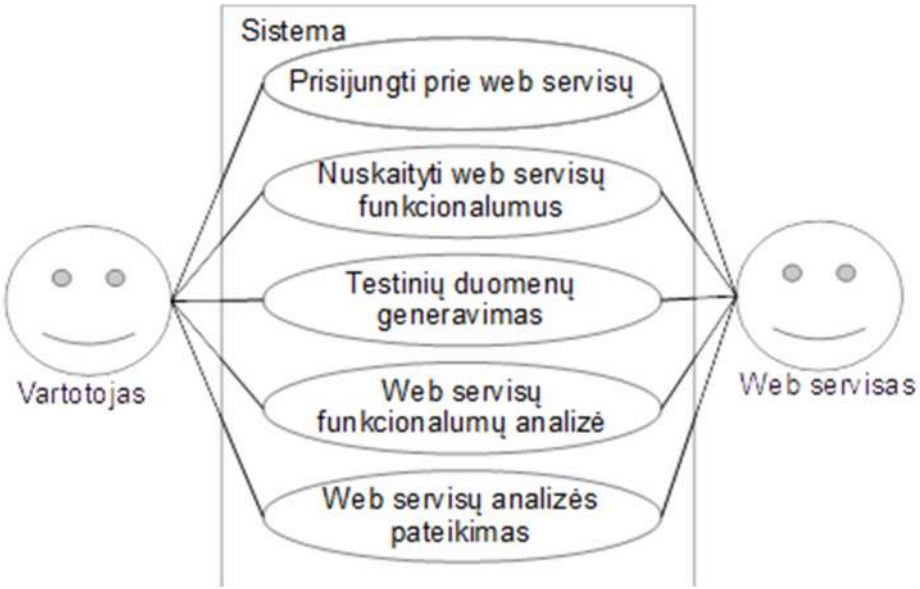
Lentelėje aprašomas veiklos įvykių sąrašas (lentelė 2).

Lentelė 2. Veiklos įvykių sąrašas

Eil. nr.	**Įvykio pavadinimas**	**Įeinantys/išeinantys informacijos srautai**
1	Prisijungimas prie web servisų	Vartotojas suveda prisijungimo duomenis (in)
2	Nuskaitomas web servisų funkcionalumas	Sistema nuskaito web servisų funkcionalumus (in)
3	Testinių duomenų generavimas	Sistema generuoja testinius duomenis (in)
4	Web servisų funcionalumų analizė	Sistema analizuoja web servisus (in)
5	Web servisų analizės rezultatų pateikimas	Sistema pateikia web servisų analizės rezultatus (out)

4.3. Panaudos atvejai

Šiame skyriuje išvardinami panaudos atvejai. Pateikiama panaudos atvejų diagrama (7 pav.)



7 pav. Panaudos atvejų diagrama

Panaudojimo atvejis	**1. Prisijungti prie web servisų**
Tikslas	Prisijungti prie web servisų
Aktoriai	Vartotojas, web servisas
Ryšiai su kitais PA	Nėra
Nefunkciniai reikalavimai	Turi būti paprasta ir suprantama ne tik IT specialistui
Prieš-sąlygos	Web servisai pasiekiami per tinklą/internetą/lokaliai, priklausomai nuo to kur randasi. Ugniasienės, antivirusinės sukonfigūruotos.
Sužadinimo sąlyga	Vartotojas paleido programą, suvedė web servisų adresus tinkle ir paspaudė mygtuką „prisijunti“.
Po-sąlyga	Programa praneša apie sėkmingą prisijungimą prie web servisų
Pagrindinis scenarijus	1. Vartotojas paleidžia programą 2. Vartotojas suvedą web servisų adresus 3. Vartotojas paspaudžia mygtuką „prisijungti“

Alternatyvūs scenarijai	Nepavykus prisijungti prie web servisų, vartotojui siūloma patikrinti ar teisingai nurodyti web servisų adresai. Jei nurodyta teisingai, jis turėtų patikrinti ugniasienę ir antivirusinę ar jos neblokuoja prisijungimo.

Panaudojimo atvejis	**2. Nuskaityti web servisų funkcionalumus**
Tikslas	Nuskaityti web servisų atliekamas funkcijas
Aktoriai	Vartotojos, web servisas
Ryšiai su kitais PA	Prisijungti prie web servisų
Nefunkciniai reikalavimai	Pateikti web servisų funkcijas lengvai suprantama ir skaitoma forma
Prieš-sąlygos	Turi būti prisijungta prie web servisų
Sužadinimo sąlyga	Po sėkmingo prisijungimo atliekamas funkcijų nuskaitymas
Po-sąlyga	Vartotojui pateikiamos web servisų funkcijos
Pagrindinis scenarijus	1. Vartotojas paspaudžia mygtuką „prisijungti“, po sėkmingo prisijungimo nuskaitomos web servisų funkcijos 2. Web servisų funkcijos išvedamos į sąrašą
Alternatyvūs scenarijai	Nėra

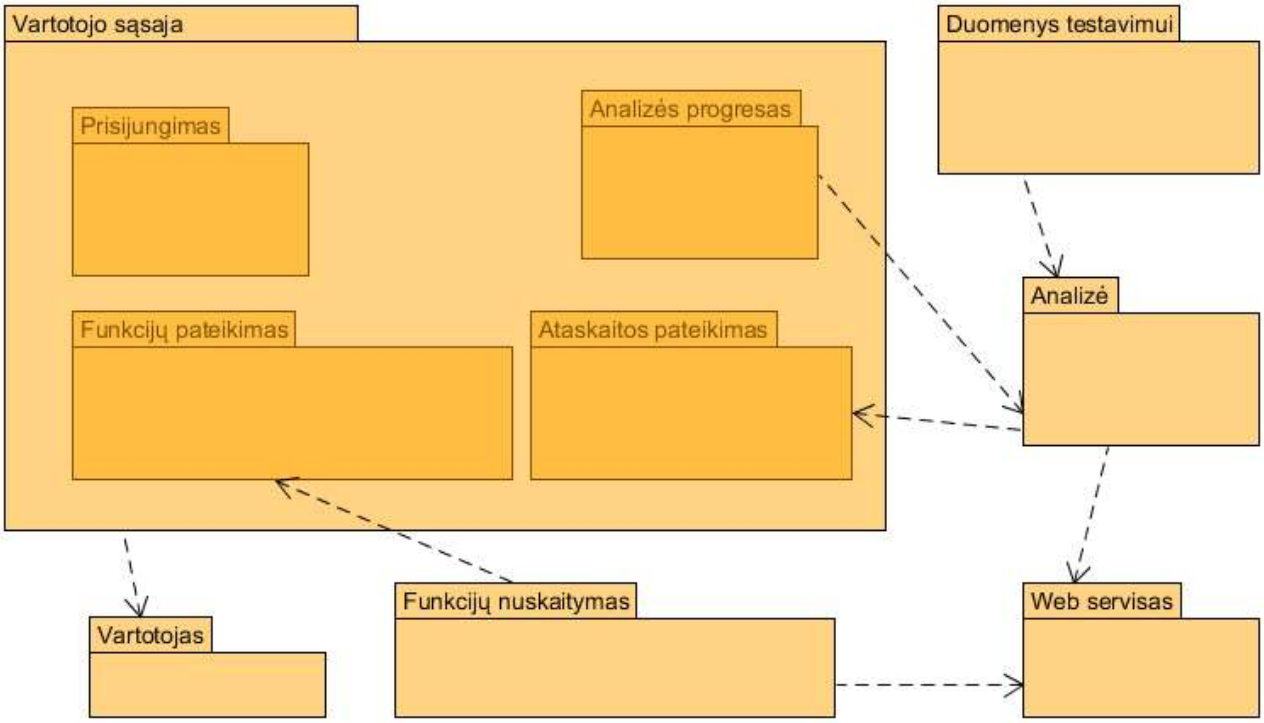
Panaudojimo atvejis	**3. Testinių duomenų generavimas**
Tikslas	Sugeneruoti fiktyvius duomenis web serviso funkcionalumų analizei
Aktoriai	Vartotojas, web servisas
Ryšiai su kitais PA	Prisijungti prie web servisų, nuskaityti web servisų funkcionalumus
Nefunkciniai reikalavimai	Turi greitai sugeneruoti testinius duomenis
Prieš-sąlygos	Turi būti prisijungta prie web serviso ir nuskaitytos web serviso funkcijos
Sužadinimo sąlyga	Vartotojas paspaudė mygtuką „Generuoti“
Po-salyga	Pradedama web servisui generuoti testinius

	duomenis.
Pagrindinis scenarijus	1. Vartotojas spaudžia mygtuką „generuoti“ 2. Generuojami testiniai duomenys
Alternatyvūs scenarijai	Nėra

Panaudojimo atvejis	**4. Web servisų funkcionalumų analizė**
Tikslas	Atlikti pilną web servisų funkcionalumų analizę
Aktoriai	Vartotojas, web servisas
Ryšiai su kitais PA	Prisijungti prie web servisų, nuskaityti web servisų funkcionalumus, testinių duomenų generavimas
Nefunkciniai reikalavimai	Turi greitai išanalizuoti
Prieš-sąlygos	Turi būti prisijungta prie web serviso, nuskaitytos web serviso funkcijos, sugeneruoti testiniai duomenys
Sužadinimo sąlyga	Vartotojas paspaudė mygtuką „analizuoti“
Po-sąlyga	Analizės duomenys pateikiami.
Pagrindinis scenarijus	1. Vartotojas spaudžia mygtuką analizuoti 2. Analizės duomenys pateikiami
Alternatyvūs scenarijai	Nėra

Panaudojimo atvejis	**5. Web servisų analizės pateikimas**
Tikslas	Pateikti web servisų analizės ataskaitą
Aktoriai	Vartotojas
Ryšiai su kitais PA	Web servisų funkcionalumų analizė
Nefunkciniai reikalavimai	Pateikta ataskaita lengvai suprantama ir analizuojama
Prieš-sąlygos	Turi būti atlikta web servisų funkcijų analizė
Sužadinimo sąlyga	Vartotojas paspaudžia mygtuką „Pateikti analizę“
Po-sąlyga	Vartotojui pateikama analizės ataskaita
Pagrindinis scenarijus	1. Vartotojui pateikiama analizė
Alternatyvūs scenarijai	Nėra

4.4. Statinis vaizdas



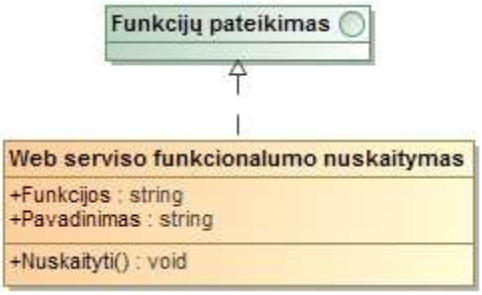
8 pav. Paketų diagrama

4.4.1. Paketų detalizavimas

Paketas “Prisijungimas” atsako už prisijungimą prie web serviso bei ryšį su juo. Su realizuota grąfine vartotojo sąsaja, prisijungimo duomenims pateikti.

9 pav. Paketo “Prisijungimas” klasių diagrama

Paketas “Funkcijų nuskaitymas” atlieka web serviso funkcijų nuskaitymą ir jų sąrašo sudarymą bei pateikimą vartotojui.

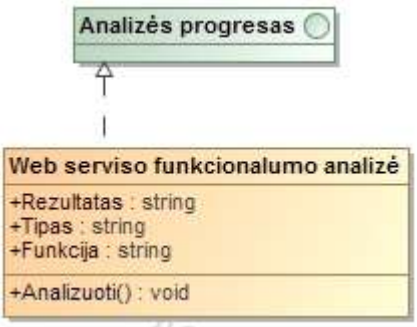


10 pav. Paketo “Funkcijų nuskaitymas” klasių diagrama

Paketas “Duomenys testavimui” atsakinga už testinių duomenų paruošimą.

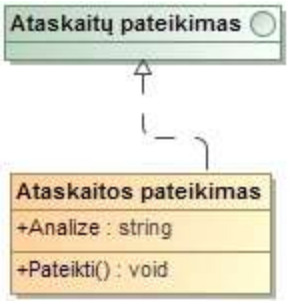
11 pav. Paketo “duomenys testavimui” klasių diagrama

Paketas “Analizė” atlieka web serviso analizę pagal testavimo duomenis.



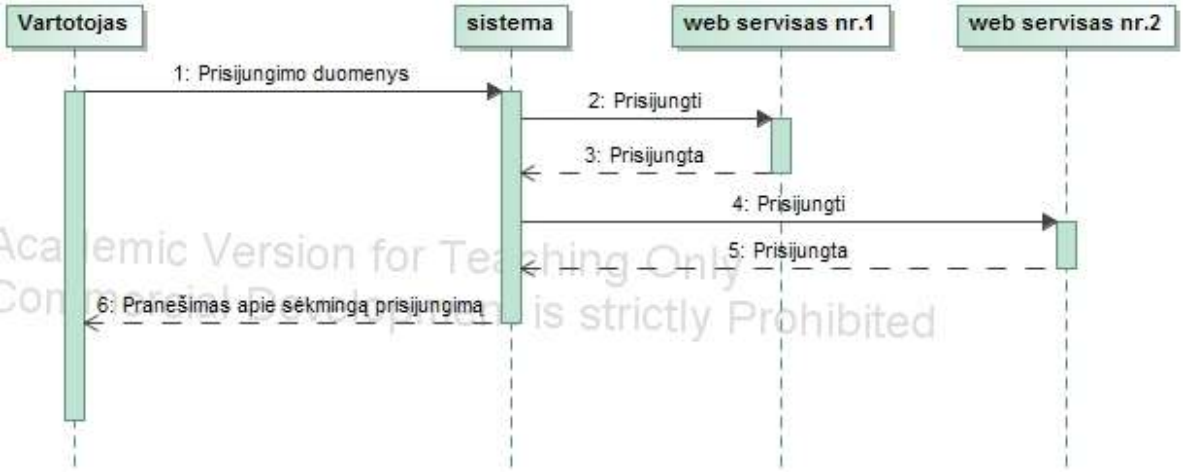
12 pav. Paketo “Analizė” klasių diagrama

Paketas “Ataskaitų pateikimas” atlieka ataskaitų pateikimą iš analizės rezultatų

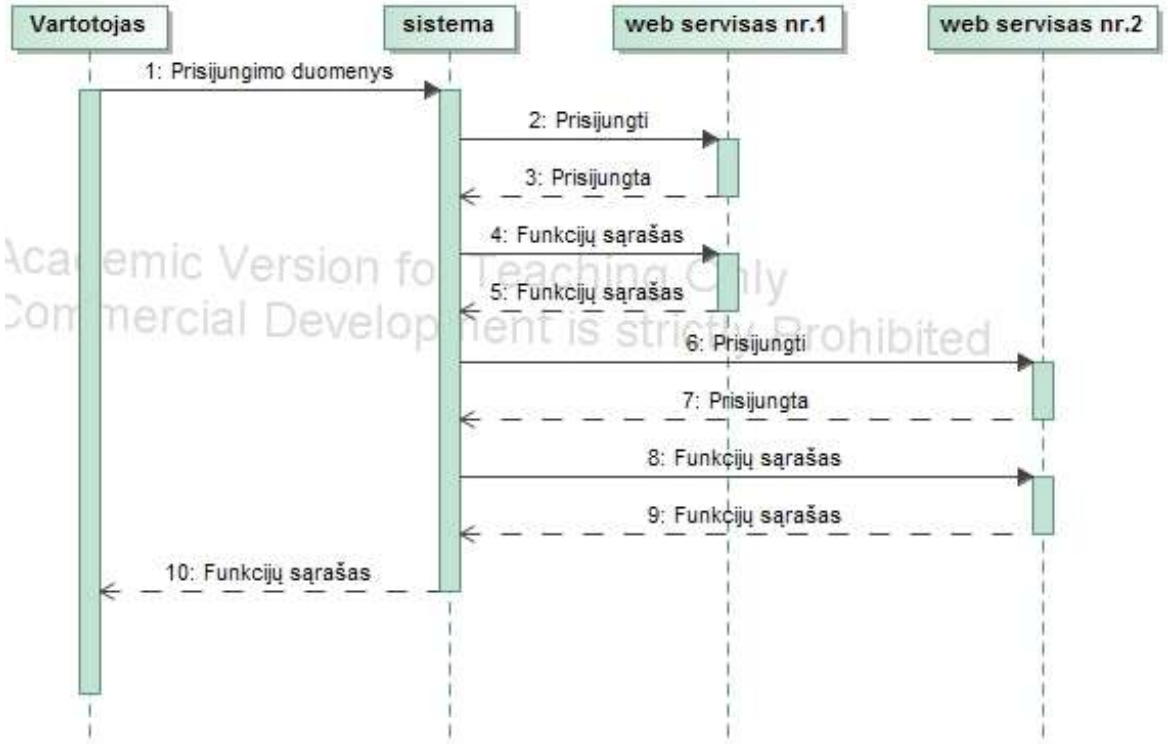


13 pav. Paketo “Ataskaitų pateikimas” klasių diagrama

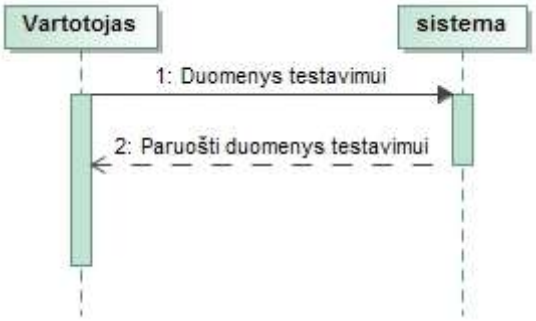
4.5. Dinaminis vaizdas



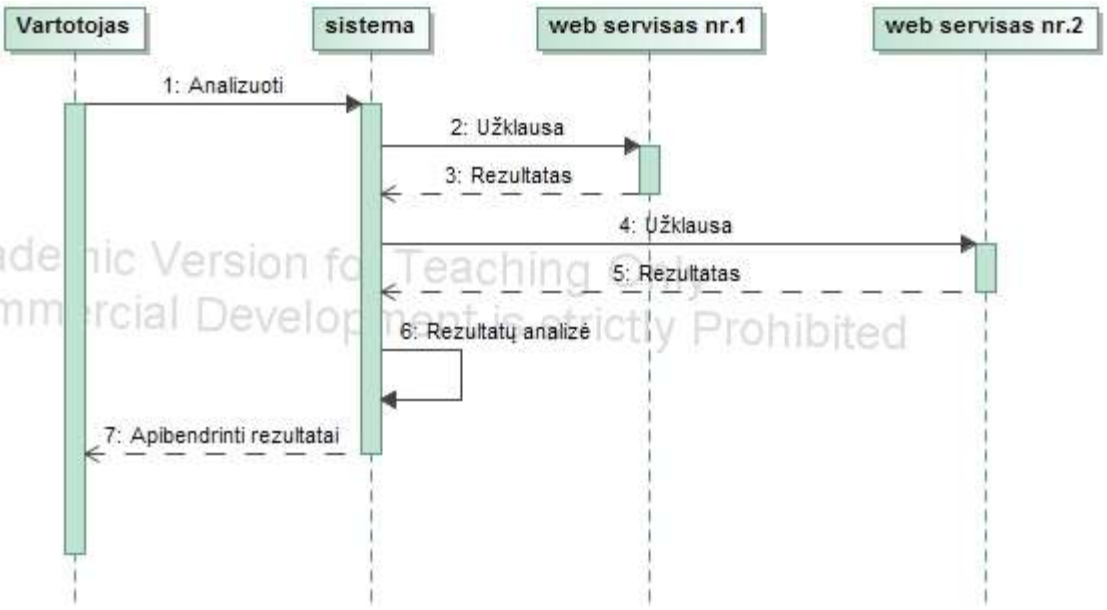
14 pav. Sekų diagrama prisijungimui prie web servisų



15 pav. Sekų diagrama web servisų funkcijų nuskaitymui



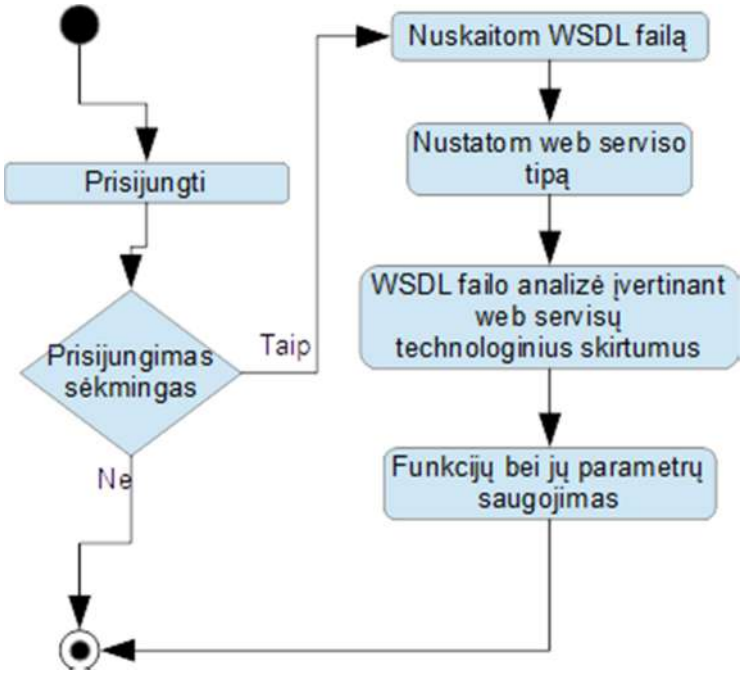
16 pav. Sekų diagrama testinių duomenų paruošimui



17 pav. Sekų diagrama web servisų funkcionalumų analizei

4.6. Prisijungimas prie web serviso

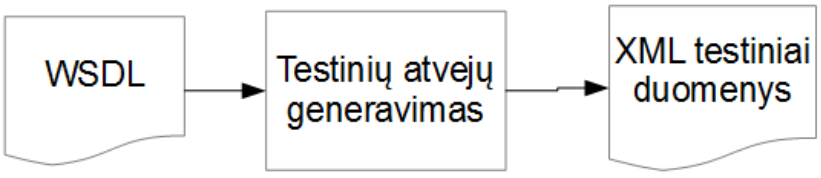
Prisijungus prie web serviso yra nuskaitomas WSDL failas ir nustatomas web serviso tipas. Tuomet atliekama WSDL failo analizė įvertinant web servisų technologinius skirtumus ir funkcijų sąrašas saugomas tolesnių etapų vykdymui (18 pav.).



18 pav. Prisijungimo prie web serviso veiklos diagrama

4.7. Automatinis testų generavimas

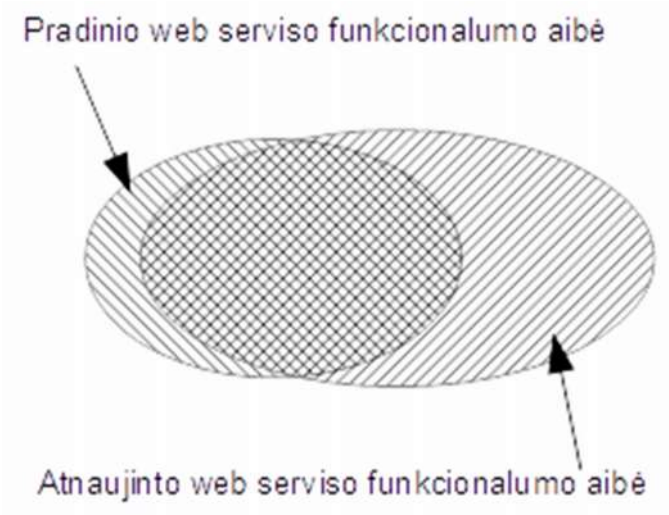
Prisijungus prie web serviso nuskaitomas WSDL failas, juo pasinaudojus ir išanalizavus, web servisui kuriami testiniai atvejai. Testinių atvejų kiekis priklauso nuo web serviso funkcijų skaičiaus bei pasirinkto testinių atvejų kiekio tenkančio vienai web serviso funkcijai (19 pav.). Kokį sugeneruoti testinių atvejų kiekį, turi nuspręsti pats vartotojas.



19 pav. Automatinis testų generavimas pasinaudojus WSDL

Susisteminti testiniai duomenys saugomi XML formatu. Kadangi testinių atvejų skaičius priklauso ir nuo pasirinkto testinių atvejų kiekio tenkančio vienai web serviso funkcijai, testiniai duomenys yra grupuojami ir išskaidomi į grupes.

Generuojant testinius atvejus rekomenduojama juos kurti pagal atnaujintą web servisą, kad būtų padengtas ir naujas atsiradęs funkcionalumas (20 pav.)



20 pav. Pradinio ir atnaujinto web servisų funkcionalumų aibės

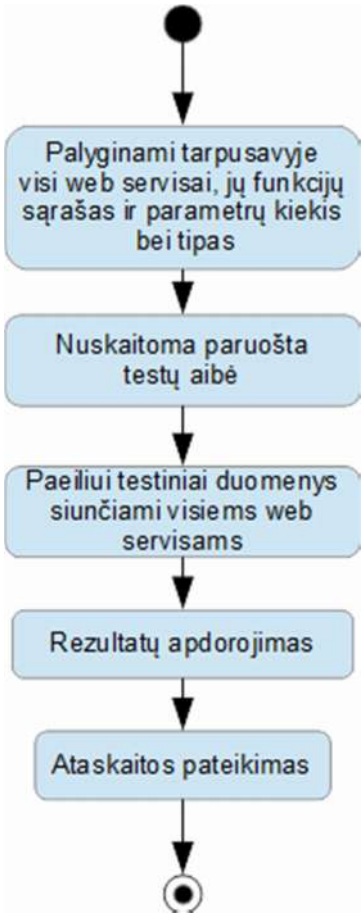
4.8. Automatinis testavimas

Automatinis testavimas vykdomas lygiagrečiai su abiem web servisais.

Pradiniame etape yra palyginamos visos web servisų funkcijos siekiant rasti esminius pasikeitimus. Yra ištraukiami abiejų web servisų WSDL failai ir yra palyginami ar jų metodai tarpusavyje atitinka. Kai kurie pasikeitimai gali būti randami vien iš WSDL failų palyginimo.

Sekančiame etape nuskaitoma paruošta testų aibė. XML failas nuskaitomas kaip duomenų lentelė. Pagal duomenų lentelėje pateiktą metodų sąrašą tikrinama ar web servisai turi aprašytą metodą su pateiktais parametrais. Visiems web servisams, kuriems rastas atitikmuo yra siunčiamos užklausos. Dėl neesminių pasikeitimų, kurie turėjo būti rasti pradiniame etape, tarkim pasikeitė duomenų tipas. Vistiek bandoma siųsti užklausas.

Gauti rezultatai analizuojami, ieškoma skirtumų tarp web servisų funkcijų grąžintų rezultatų (21 pav.).



21 pav. Automatinio testavimo veiklos diagrama

5. EKSPERIMENTAS

Sukuriamas visiškai naujas internetinis servisas “Nr1” su dviem funkcijomis:

- Add
- Subtract

Web serviso „Nr1“ padaroma įdentiška kopija „Nr2“ ir pridedama papildoma nauja funkcija „concat“:

- Add
- Subtract
- Concat

Pasinaudoję gautu web serviso “Nr2” funkcijų sąrašu, generuojame atsitiktines reikšmes ir bandome testuoti. Pagal ataskaitą (lentelė 3) matome, jog lygindamas web servisų skirtumus rado, kad servisas “Nr1” neturi funkcijos “concat”. Bandant web servisų funkcijoms paduoti reikšmes, gauti atsakymai sutampa.

Lentelė 3:

Nr.	Servisas	Metodas	Testas	Rezultatas
0	Nr2	concat	Skirtumai	Nr2 concat - yra, Nr1 concat - nėra
1	Nr1	Add	1,6	7
2	Nr2	Add	1,6	7
3	Nr1	Add	5,-3	2
4	Nr2	Add	5,-3	2
5	Nr1	Subtract	98,15	83
6	Nr2	Subtract	98,15	83
7	Nr1	Subtract	42,87	-45
8	Nr2	Subtract	42,87	-45
9	Nr1	concat	Labas,Rytas	Servisas neturi šio metodo.
10	Nr2	concat	Labas,Rytas	LabasRytas
11	Nr2	concat	Viso,Gero	VisoGero

Web servisui “Nr2” pakeičiame funkciją “Add”, įmituojame pakeitimus, web serviso atnaujinimą.

Bandome analizuoti su ta pačia testinių atvejų aibę. Pagal ataskaitą (lentelė 4) matome, kad serviso “Nr2” funkcija “Add” grąžina skirtingas reikšmes, negu serviso “Nr1”.

Lentelė 4:

Nr.	Servisas	Metodas	Testas	Rezultatas
0	Nr2	concat	Skirtumai	Nr2 concat - yra, Nr1 concat - nėra
1	Nr1	Add	1,6	7
2	Nr2	Add	1,6	0
3	Nr1	Add	5,-3	2
4	Nr2	Add	5,-3	-5
5	Nr1	Subtract	98,15	83
6	Nr2	Subtract	98,15	83
7	Nr1	Subtract	42,87	-45
8	Nr2	Subtract	42,87	-45
9	Nr1	concat	Labas,Rytas	Servisas neturi šio metodo.
10	Nr2	concat	Labas,Rytas	LabasRytas
11	Nr2	concat	Viso,Gero	VisoGero

6. APIBENDRINIMAS IR IŠVADOS

Mano siūlomo metodo pranašumai yra:

- Vienu metu lygiagrečiai testuojami du web servisai, sena ir nauja versijos.
- Web servisų palyginimas tarpusavyje.
- Sumažina žmogiškųjų resursų poreikį.
- Sumažina monotoniško darbo kiekį.
- Lengvai pakartotinai panaudojamas.

Mano programos trūkumai:

- Ataskaitos pateikimas sunkiai skaitomas, reikia ieškoti skirtumų.
- Testinių atvejų generavimas atsitiktinėmis reikšmėmis reikalauja didelio testinių atvejų kiekio, siekiant kuo labiau ištestuoti web servisus ir apimti visus kritinius rėžius. Todėl, kiek bu testinių atvejų paliekama apsispręsti vartotojui.

Siūlomo metodo pats dydžiausias pranašumas, jog vienu metu yra testuojami du web servisai, senas ir atnaujintas web servisas.

7. TERMINŲ IR SANTRAUKŲ ŽODYNAS

WSDL (Web Services Definition/Description Language) – web servisų aprašymo kalba;

SOAP (Simple Object Access Protocol) – paprastas objektų sujungimo protokolas;

UDDI (Universal Description, Discovery and Integration) – universali web servisų aprašymo, suradimo ir integravimo technologija;

XML (angl. „Extensible Markup Language“) - „žymėjimo“ kalba dokumentams, kuriuose saugoma struktūrizuota tekstinė informacija.

Regresinis testavimas - programinės įrangos testavimo tipas, kuriuo siekiama rasti naujos programinės įrangos klaidas po sistemos pakeitimų, atnaujinimų ar konfigūracijos pasikeitimų.

8. LITERATŪRA

[1] Yijun Yu ; Open Univ., London ; Jianguo Lu ; Fernandez-Ramil, J. ; Yuan, P. „Comparing Web Services with other Software Components“, Web Services, 2007. ICWS 2007. IEEE International Conference on, pp. 48-56, Sept.-Oct. 2010.

[2] Chaturvedi, A. ; Dept. of Comput. Sci. Eng., IIITDM-J, Jabalpur, India ; Gupta, A., „A tool supported approach to perform efficient regression testing of web services“, Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2013 IEEE 7th International Symposium on the, pp.50-55, 23-23 Sept. 2013.

[3] Vanderveen, P. ; Dept. of Comput. Sci., K ing's Univ. Edmonton, Edmonton, AB, Canada ; Janzen, M. ; Tappenden, A.F., „A Web Service Test Generator“, Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on, pp.516-520, Sept. 29 2014-Oct. 3 2014.

[4] Kai Ma ; Coll. of Comput. Sci., Beijing Univ. of Technol., Beijing, China ; Jin Wang ; Hongli Yang ; Jun Yan, „Choreography Scenario-Based Test Data Generation“, Theoretical Aspects of Software Engineering Conference (TASE), 2014, pp.70-73, 1-3 Sept. 2014.

[5] Chunyan Ma ; Coll. of Software & Microelectron., Northwestern Polytech. Univ., Xian ; Chenglie Du ; Tao Zhang ; Fei Hu, „WSDL-Based Automated Test Data Generation for Web Service“, Computer Science and Software Engineering, 2008 International Conference on, vol.2, pp.731-737, 12-14 Dec. 2008.

[6] Tekli, J.M. ; Dept. of Comput. Sci. & Stat., Univ. of Sao Paulo, Sao Carlos, Brazil ; Damiani, E. ; Chbeir, R. ; Gianini, G., „SOAP Processing Performance and Enhancement“, Services Computing, IEEE Transactions on, vol.5, no.3, pp. 387-403, 24 February 2011.

[7] Crasso, M. ; Univ. Nat. del Centro de la provincia de Buenos Aires, Buenos Aires, Argentina ; Rodriguez, J.M. ; Zunino, A. ; Campo, M., „Revising WSDL Documents: Why and How“, Internet Computing, IEEE , vol.14, no.5, pp.48-56, Sept.-Oct. 2010.

[8] Erik Christensen, Microsoft; Francisco Curbera, IBM Research; Greg Meredith, Microsoft; Sanjiva Weerawarana, IBM Research; „Web Services Description Language (WSDL) 1.1“, http://www.w3.org/TR/wsdl [2015-03-29].

[9] Tom Bellwood, IBM; Steve Capell; Luc Clement, Systinet; John Colgrave, IBM; Matthew J. Dovey; Daniel Feygin, UnitSpace; Andrew Hately, IBM; Rob Kochman, Microsoft; Paul Macias, LMI; Mirek Novotny, Systinet; Massimo Paolucci; Claus von Riegen, SAP AG; Tony Rogers, Computer Associates; Katia Sycara; Pete Wenzel, SeeBeyond Technology; Zhe Wu, Oracle; „UDDI Spec Technical Committee Draft“, http://www.uddi.org/pubs/uddi_v3.htm [2015-03-29]

[10] Don Box, DevelopMentor; David Ehnebuske, IBM; Gopal Kakivaya, Microsoft; Andrew Layman, Microsoft; Noah Mendelsohn, Lotus Development Corp.; Henrik Frystyk Nielsen, Microsoft; Satish Thatte, Microsoft; Dave Winer, UserLand Software, Inc.; "Simple Object Access Protocol (SOAP) 1.1",http://www.w3.org/TR/2000/NOTE-SOAP-20000508/ [2015-02-17]

[11] Tim Bray, Textuality and Netscape; Jean Paoli, Microsoft; C. M. Sperberg-McQueen, W3C; Eve Maler, Sun Microsystems, Inc.; François Yergeau; "Extensible Markup Language (XML) 1.0 (Fifth Edition)", http://www.w3.org/TR/REC-xml/ [2015-04-01]

[12] Mustafa, K.M.; Al-Zaytoonah Univ. of Jordan, Amman, Jordan; Al-Qutaish, R.E.; Muhairat, M.I.; „Classification of Software Testing Tools Based on the Software Testing Methods", Computer and Electrical Engineering, 2009. ICCEE '09. Second International Conference on (Volume:1), pp.229-233, 28-30 Dec. 2009

[13] Godefroid, P. ; Microsoft Res., Redmond, WA ; de Halleux, P. ; Nori, A.V. ; Rajamani, S.K.; "Automating Software Testing Using Program Analysis",Software, IEEE (Volume:25 , Issue: 5), pp.30-37, 19 August 2008

[14] Wu Chou ; Avaya Labs. Res. ; Li Li ; Feng Liu; „Web Services for Service-Oriented Communication", Collaborative Computing: Networking, Applications and Worksharing, 2006. CollaborateCom 2006. International Conference on, pp.1-8, 17-20 Nov. 2006

[15] Giri, K. ; Dept. of Comput. Sci. & Eng., Motilal Nehru Nat. Inst. of Technol., Allahabad, India ; Yadav, D.K.; „Modeling and verifying Web service Broker based architecture using CCS", Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on, pp.889-894, 30-31 March 2012

[16] Ching-Seh Wu ; Dept. of Comput. Sci. & Eng., Oakland Univ., Rochester, NY, USA ; Khoury, I.; „E-Healthcare Web Service Broker Infrastructure in Cloud Environment", Services (SERVICES), 2012 IEEE Eighth World Congress on, pp.317-322, 24-29 June 2012

[17] Tsai, W.T. ; Dept. of Comput. Sci. & Eng., Arizona State Univ., Tempe, AZ, USA ; Paul, R. ; Weiwei Song ; Zhibin Cao; „Coyote: an XML-based framework for Web services testinG", High Assurance Systems Engineering, 2002. Proceedings. 7th IEEE International Symposium on, pp.173-174, 2002

9. PRIEDAI

9.1. Straipsnis pateiktas “IT2015“

Pateiktas “INFORMACINĖS TECHNOLOGIJOS 2015”, 20-OJI TARPUNIVERSITETINĖ MAGISTRANTŲ IR DOKTORANTŲ KONFERENCIJA.

AUTOMATINIS WEB SERVISŲ TESTAVIMAS

Vaidas Nacius[1], **Šarūnas Stanskis**[2]

[1]*Kauno technologijos universitetas, Programų inžinerijos katedra, Studentų g. 50-406, Kaunas, Lietuva, vaidas.nacius@ktu.edu*

[2]*Kauno technologijos universitetas, Programų inžinerijos katedra, Studentų g. 50-406, Kaunas, Lietuva, sarunas.stanskis@ktu.edu*

Santrauka (abstract). Automatinis web servisų testavimas susideda iš WSDL failų apdorojimo, testų generavimo ir užklausų vykdymo bei rezultatų išvedimo. Prisijungiant prie web serviso atliekama WSDL failo analizė, išanalizavus kūriami testavimo atvejai. Su tais pačiais testiniais duomenimis testuojami visi pasirinkti web servisai ir pagal gautus rezultatus įvertinamas atnaujintasis web servisas.

Raktiniai žodžiai: web servisai, web service testing tool, SOAP, WSDL, UDDI, Web Services Architectures, regression testing, Performance Testing, XML, program testing, testing, web service test generator, test generator.

1 Įžanga

Web servisai yra interneto taikomųjų programų rūšis, kurie gali vykdyti tiek paprastas užklausas, tiek sudėtingus procesus. Šiuo metu internetas yra visuotinai paplitęs, o web servisai veikia interneto naršyklėse, todėl galima sakyti, kad web servisais galima naudotis visur, kur tik yra interneto prieiga. Web servisams sparčiai plintant ir juos panaudojant tiek paprastuose, tiek sudėtinguose verslo procesuose atsiranda patikimų web servisų poreikis. Yra įvairių web servisų rūšių ir jie kūriami ant įvairiausių platformų: Net frameworks, Apache Axis java2wsdl, PHP, Oracle ir t.t. [1].

Atnaujinant, perprojektuojant web servisus atsiranda poreikis lyginti seną versiją su nauja, norint užtikrinti naujojo web proceso efektyvumą ir patikimumą [2]. Vienas iš galimų testavimo būdų yra regresinis testavimas, kuris yra skirtas būtent atnaujinamų bei perprojektuojamų sistemų testavimui.

Praktikoje dėl įvairių priežasčių dažnai tenka perprojektuoti lygtines sistemas. Padėtį apsunkina tai, kad senojoje sistemoje naudotos technologijos gali būti nebepalaikomos, o naujoje sistemos versijoje gali būti naudojamos kitos technologijos, gali pasikeisti duomenų formatai ar atsirasti papildomi funkcionalumai. Tačiau, naujosios sistemos teikiamo funkcionalumo dalis, kuri atitinka seniau veikusios sistemos funkcionalumą turi veikti taip pat.

Perprojektuojant lygtines sistemas gali būti pakeistas vidinių metodų veikimas, perkurta sistemos architektūra, ar net panaudotos visiškai naujos technologijos. Dėl tokių pakeitimų dažnai pasitaiko naujų klaidų, ko pasekoje atsiranda poreikis abiejų sistemų teikiamo funkcionalumo palyginimui, kurį galima atlikti naudojant regresinį testavimą.

Nagrinėjant ir gilinantis į kitų tyrėjų atliktus darbus apie web servisų testavimą galima sakyti, kad dauguma iš jų susikoncentruoja į web serviso kaip vieneto (vienos sistemos) testavimą ir testų automatizavimą [3] [4] [5]. Tačiau yra svarbu pažvelgti į web servisų testavimą ir platesniu kampu, neapsibrėžiant vieno web serviso testavimu, sukurti metodiką, kuri leistų testuoti vienu metu kelis web servisus, dažniausiai du: seną web serviso versiją ir atnaujintą.

2 Prisijungimas prie web serviso

Pagrindinis reikalavimas web servisų tiekėjui ir serviso užsakovui yra gebėjimas kurti ir sintaksiškai analizuoti SOAP pranešimus bei gebėjimas komunikuoti tinklo pagalba, kitaip sakant apsikeisti XML pranešimais [6]. Paprastai SOAP serveris, veikiantis web aplikacijų serveryje, atlieka minėtas funkcijas. Gali būti naudojama tam tikros programinės kalbos biblioteka, užtikrinanti minėtas funkcijas.

WSDL yra pagrįsta XML technologija ir leidžia katalogizuoti bei aprašyti web servisus [7]. WSDL yra naudojama komunikavimo proceso tarp web servisų automatizavimui. WSDL aprašo web servisų interfeisą, prisijungimo prie web servisų protokolą (pvz. SOAP) ir galutinę web servisų būvimo vietą (t.y. nuorodą į web servisą).

WSDL dokumentai web servisų aprašymui naudoja šiuos elementus [8]:

types – konteineriai, skirti duomenų tipų aprašymams, naudojant tam tikrą sistemą (pavyzdžiui XSD).

message – abstraktus komunikuojančių duomenų aprašymas.

operation – abstraktus web serviso atliekamų veiksmų aprašymas (įėjimo išėjimo parametrai).

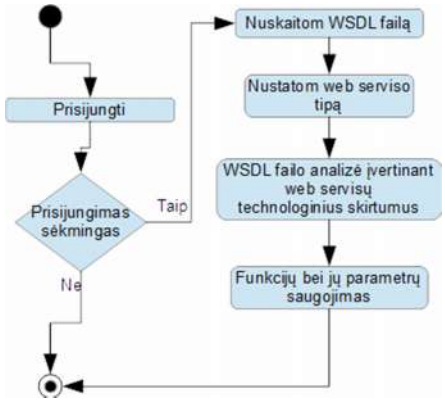
portType – operacijų, kurias sugeba atlikti vienas ar kitas baigtinis serviso komponentas, abstrakčių aprašymų aibė.

binding – konkreti protokolo ar duomenų formato specifikacija, tam tikram porto tipui (portType).

port – atskiras baigtinis serviso komponentas, apibūdinamas kaip sąryšio ir tinklo adreso kombinacija.

service – aprašo web servisą kaip port ar endpoint elementų kolekciją.

Prisijungiant prie web serviso atliekama WSDL failo analizė ir funkcijų sąrašas saugomas tolesnių etapų vykdymui (1 pav.).



1 pav. Prisijungimo prie web serviso veiklos diagrama

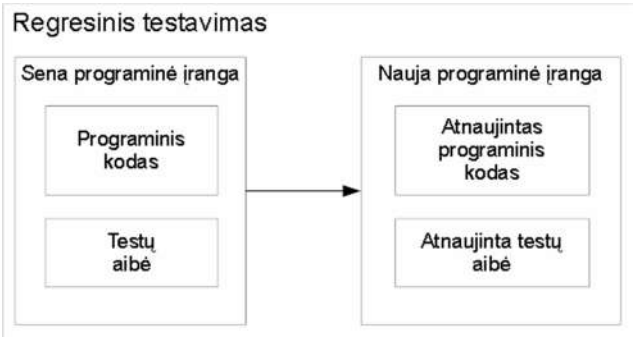
3 Regresinis testavimas

Regresinis testavimas yra programinės įrangos testavimo tipas, kuriuo siekiama rasti naujos programinės įrangos klaidas po sistemos pakeitimų, atnaujinimų ar konfigūracijos pasikeitimų [2].

Regresinio testavimo pagrindinis tikslas yra patikrinti, ar nauja programinė įranga išlaiko ankstesnėse versijose veikusį funkcionalumą ne tik pasikeitusiose dalyse bei moduliuose, bet ir visą bendrą web serviso funkcionalumą.

Viena iš pagrindinių regresijų atsiradimo priežasčių yra ta, jog programinio kodo dalys yra glaudžiai susiję, todėl programuotojui keičiant vieną modulį, gali būti pažeistas kitas susijęs programinės dalies kodas.

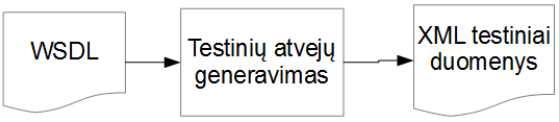
Regresinis testavimas gali būti taikomas bet kuriame testavimo lygyje. Atliekant visus jau taikytus testus, siekiama nustatyti, jog programinės įrangos funkcionalumas nepasikeitė net ir nepakitusiose ir neatnaujintose programos dalyse. Principas gaunasi toks, kad jau sukauptų testų aibę panaudojame pakeistos programinės įrangos testavimui (2 pav.).



2 pav. Regresinis testavimas

4 Automatinis testų generavimas

Prisijungus prie web serviso nuskaitomas WSDL failas, juo pasinaudojus ir išanalizavus, web servisui kuriami testiniai atvejai. Testinių atvejų kiekis priklauso nuo web serviso funkcijų skaičiaus bei pasirinkto testinių atvejų kiekio tenkančio vienai web serviso funkcijai (3 pav.).



3 pav. Automatinis testų generavimas pasinaudojus WSDL

Susisteminti testiniai duomenys saugomi XML formatu. Grupuojami duomenų lentelės pavidalu, aprašomi keturiais laukais: „Funkcijos pavadinimas“, „Parametro pavadinimas“, „Parametro reikšmė“, „Grupės nr.“ (4 pav.)

```
<_x0030_>
    <functionname>getChangedSubscribers</functionname>
    <parmname>api_key</parmname>
    <parmvalue>RWIFOLZDOGEFVRXOBLAEIDOYQBEWEOVMMAOPVI
    <groupid>9</groupid>
</_x0030_>
<_x0030_>
    <functionname>getActiveSubscribersByGroup</functionname>
    <parmname>api_key</parmname>
    <parmvalue>R</parmvalue>
    <groupid>0</groupid>
</_x0030_>
<_x0030_>
<_x0030_>
    <functionname>getActiveSubscribersByGroup</functionname>
    <parmname>from</parmname>
    <parmvalue>-309517455</parmvalue>
    <groupid>0</groupid>
</_x0030_>
<_x0030_>
    <functionname>getActiveSubscribersByGroup</functionname>
    <parmname>limit</parmname>
    <parmvalue>-679149188</parmvalue>
    <groupid>0</groupid>
</_x0030_>
<_x0030_>
    <functionname>getActiveSubscribersByGroup</functionname>
    <parmname>api_key</parmname>
    <parmvalue>RWIFOLZDOGEFVRXOBLAEIDOYQBEWEOVMMAOPVI
    <groupid>1</groupid>
```

4 pav. Susisteminti testiniai duomenys

Kadangi testinių atvejų skaičius priklauso ir nuo pasirinkto testinių atvejų kiekio tenkančio vienai web serviso funkcijai, testiniai duomenys yra grupuojami ir išskaidomi į grupes.

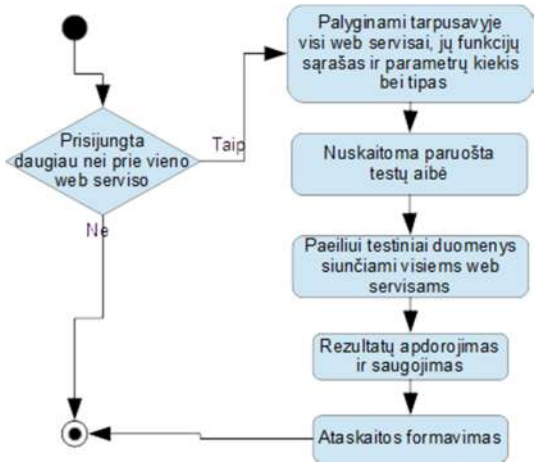
5 Automatinis testavimas

Automatinis testavimas vykdomas su pasirinktais web servisais.

Pradiniame etape yra palyginamos visos web servisų funkcijos siekiant rasti esminius pasikeitimus. Yra ištraukiami visų web servisų WSDL failai ir yra palyginami ar jų metodai tarpusavyje atitinka. Kai kurie pasikeitimai gali būti randami vien iš WSDL failų palyginimo.

Sekančiame etape nuskaitoma paruošta testų aibė. XML failas nuskaitomas kaip duomenų lentelė. Pagal duomenų lentelėje pateiktą metodų sąrašą tikrinama ar web servisai turi aprašytą metodą su pateiktais parametrais. Visiems web servisams, kuriems rastas atitikmuo yra siunčiamos užklausos. Dėl neesminių pasikeitimų, kurie turėjo būti rasti pradiniame etape, tarkim pasikeitė duomenų tipas. Vistiek bandoma siųsti užklausas.

Gauti rezultatai analizuojami, ieškoma skirtumų tarp web servisų funkcijų grąžintų rezultatų bei lyginama greitaveika (5 pav.).



5 pav. Automatinio testavimo veiklos diagrama

6 Išvados / Pastabos / Ateities darbai / Padėkos

Siūlomo metodo pranašumas yra tas, jog vienu metu galima testuoti daugiau negu vieną web servisą. Pasinaudojus automatinio testavimo įrankiu galima testuoti ir įvertinti web servisus. Analizuoti ar atnaujintasis web servisas atlieka ir grąžina tuos pačius rezultatus kaip ir senasis, bei palyginti jų greitaveikas.

Literatūros sąrašas

[1] Yijun Yu ; Open Univ., London ; Jianguo Lu ; Fernandez-Ramil, J. ; Yuan, P. „Comparing Web Services with other Software Components“, Web Services, 2007. ICWS 2007. IEEE International Conference on, pp. 48-56, Sept.-Oct. 2010.

[2] Chaturvedi, A. ; Dept. of Comput. Sci. Eng., IIITDM-J, Jabalpur, India ; Gupta, A., „A tool supported approach to perform efficient regression testing of web services“, Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2013 IEEE 7th International Symposium on the, pp.50-55, 23-23 Sept. 2013.

[3] Vanderveen, P. ; Dept. of Comput. Sci., K ing's Univ. Edmonton, Edmonton, AB, Canada ; Janzen, M. ; Tappenden, A.F., „A Web Service Test Generator“, Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on, pp.516-520, Sept. 29 2014-Oct. 3 2014.

[4] Kai Ma ; Coll. of Comput. Sci., Beijing Univ. of Technol., Beijing, China ; Jin Wang ; Hongli Yang ; Jun Yan, „Choreography Scenario-Based Test Data Generation“, Theoretical Aspects of Software Engineering Conference (TASE), 2014, pp.70-73, 1-3 Sept. 2014.

[5] Chunyan Ma ; Coll. of Software & Microelectron., Northwestern Polytech. Univ., Xian ; Chenglie Du ; Tao Zhang ; Fei Hu, „WSDL-Based Automated Test Data Generation for Web Service“, Computer Science and Software Engineering, 2008 International Conference on, vol.2, pp.731-737, 12-14 Dec. 2008.

[6] Tekli, J.M. ; Dept. of Comput. Sci. & Stat., Univ. of Sao Paulo, Sao Carlos, Brazil ; Damiani, E. ; Chbeir, R. ; Gianini, G., „SOAP Processing Performance and Enhancement“, Services Computing, IEEE Transactions on, vol.5, no.3, pp. 387-403, 24 February 2011.

[7] Crasso, M. ; Univ. Nat. del Centro de la provincia de Buenos Aires, Buenos Aires, Argentina ; Rodriguez, J.M. ; Zunino, A. ; Campo, M., „Revising WSDL Documents: Why and How“, Internet Computing, IEEE , vol.14, no.5, pp.48-56, Sept.-Oct. 2010.

[8] Erik Christensen, Microsoft; Francisco Curbera, IBM Research; Greg Meredith, Microsoft; Sanjiva Weerawarana, IBM Research; „Web Services Description Language (WSDL) 1.1“, http://www.w3.org/TR/wsdl [2015-03-29].

Automated web service testing

Automated web service testing consists of a WSDL file processing, test generation, execution of queries and the derivation of results. Connecting to the web service WSDL file analysis is performed and test cases are created after the analysis. All selected web services are tested with the same test data and according to the results obtained, a new updated web service is assessed.