VILNIAUS UNIVERSITETAS

SANDRA SVANIDZAITĖ

SPIRALINIS PROCESO MODELIS PASLAUGŲ STILIAUS ARCHITEKTŪROS ĮMONIŲ SISTEMŲ NEFUNKCINIAMS REIKALAVIMAMS IŠGAUTI IR ANALIZUOTI

Daktaro disertacijos santrauka

Fiziniai mokslai, informatika (09 P)

Vilnius, 2015

Disertacija rengta 2010–2014 m. Vilniaus universiteto Matematikos ir informatikos institute.

Mokslinis vadovas – prof. dr. Albertas Čaplinskas (Vilniaus universitetas, fiziniai mokslai, informatika – 09 P).

Disertacija ginama Vilniaus universiteto Informatikos mokslo krypties taryboje:

Pirmininkas – prof. habil. dr. Antanas Žilinskas (Vilniaus universitetas, fiziniai mokslai, informatika – 09 P).

Nariai:

prof. dr. Rimantas Butleris (Kauno technologijos universitetas, fiziniai mokslai, informatika – 09 P),
prof. dr. Saulius Gudas (Vilniaus universitetas, fiziniai mokslai, informatika – 09 P),
doc. dr. Olga Kurasova (Vilniaus universitetas, fiziniai mokslai, informatika – 09 P),
prof. dr. Raimundas Matulevičius (Tartu universitetas, fiziniai mokslai, informatika – 09 P).

Disertacija ginama viešame Vilniaus universiteto Informatikos mokslo krypties tarybos posėdyje 2015 m. rugsėjo 28 d. 13 val. Vilniaus universiteto Matematikos ir informatikos instituto 203 auditorijoje.
Adresas: Akademijos g. 4, LT-08663, Vilnius, Lietuva.

Disertacijos santrauka išsiųsta 2015 m. rugpjūčio 28 d.
Disertaciją galima peržiūrėti Vilniaus universiteto bibliotekoje ir VU interneto svetainėje adresu: www.vu.lt/lt/naujienos/ivykiu-kalendorius

VILNIUS UNIVERSITY

SANDRA SVANIDZAITĖ

SPIRAL PROCESS MODEL FOR CAPTURE AND ANALYSIS OF NON-FUNCTIONAL REQUIREMENTS OF SERVICE-ORIENTED ENTERPRISE SYSTEMS

Summary of Doctoral Dissertation
Physical Sciences, Informatics (09 P)

Vilnius, 2015

Dissertation was written during the period 2010 – 2014 at the Vilnius University Institute of Mathematics and Informatics.

Scientific Supervisor

Prof. Dr. Albertas Čaplinskas (Vilnius University, Physical Sciences, Informatics – 09 P).

The dissertation will be defended at the Council of the Scientific Field of Informatics of Vilnius University:

Chairman

Prof. Habil. Dr. Antanas Žilinskas (Vilnius University, Physical Sciences, Informatics – 09 P).

Members:

Prof. Dr. Rimantas Butleris (Kaunas University of Technology, Physical Sciences, Informatics – 09 P),
Prof. Dr. Saulius Gudas (Vilnius University, Physical Sciences, Informatics – 09 P),
Doc. Dr. Olga Kurasova (Vilnius University, Physical Sciences, Informatics – 09 P),
Prof. Dr. Raimundas Matulevičius (University of Tartu, Physical Sciences, Informatics – 09 P).

The dissertation will be defended at the public meeting of the Council of the Scientific Field of Informatics Sciences of Vilnius University in the auditorium number 203 at the Vilnius University Institute of Mathematics and Informatics, at 1 p. m. on the 28th of September 2015.
Address: Akademijos str. 4, LT – 08663, Vilnius, Lithuania.

The summary of the doctoral dissertation was sent out on 28th of August 2015.
A copy of the doctoral dissertation is available for review at the Library of Vilnius University or on this website: www.vu.lt/lt/naujienos/ivykiu-kalendorius

Įvadas

Mokslo problemos aktualumas

Paslaugų stiliaus paradigma – nauja programų sistemų kūrimo paradigma, atsiradusi praeito šimtmečio paskutiniame dešimtmetyje. Ši paradigma perėmė daugumą sąvokų ir principų iš ankstesnių paradigmų, tokių kaip objektinis programavimas (angl. *OOP – object-oriented paradigm*), komponentinis programavimas (angl. *COP – component-oriented programming*) ir atvirieji išskirstytieji skaičiavimai (angl. *ODB – open distributed processing*). Paslaugų paradigma nuo savo pirmtakių skiriasi tobulesniu turinių atskyrimo principu, nes įvedamas papildomas verslo logikos sluoksnis. Ji skirta išskirstytiesiems pajėgumams, kurie gali būti skirtingose nuosavybės kontrolės srityse, organizuoti ir panaudoti (SOA RM, 2006). Žmonės ir organizacijos kuria programinį funkcionalumą problemoms, su kuriomis susiduria vykdydami savo veiklą, spręsti, todėl natūralu galvoti, kad vieno asmens (taip pat ir programų sistemos) poreikius gali patenkinti kito asmens (programų sistemos) siūlomas funkcionalumas. Matomumas, sąveika ir poveikis yra pagrindinės sąvokos, apibūdinančios paslaugų stiliaus paradigmą. Matomumas padeda atrasti poreikį atitinkantį funkcionalumą. Šiuo tikslu paprastai pateikiami sistemos funkcijų (paslaugų) aprašai, jų naudojimo apribojimai ir galimi prieigos prie funkcionalumo mechanizmai. Paslaugų aprašų sintaksė ir semantika turi būti plačiai prieinama ir suprantama. Savaime aišku, kad neįmanoma apibūdinti kiekvieno poveikio, kurį gali sukelti naudojimasis trečiųjų šalių paslaugomis, todėl paslaugų stiliaus paradigmos kertine ypatybe yra laikoma galimybė pasinaudoti kitų sukurtomis paslaugomis nežinant visų jų realizacijos detalių. Paslauga yra pagrindinis paslaugų stiliaus paradigmos konceptas. Matomumo, sąveikos ir poveikio sąvokomis taip pat apibūdinama ir kiekviena paslaugų stiliaus paradigmos paslauga. Paslaugų stiliaus architektūra (angl. *SOA – Service-oriented architecture*) yra paslaugoms kurti skirtas architektūros stilius. Pats šis architektūros stilius nesprendžia jokių dalykinės srities problemų. Jis veikiau padeda organizuoti ir efektyviausiai išnaudoti savas ir trečiųjų šalių paslaugas.

Daugelis didelių paslaugų stiliaus architektūros programų sistemų projektų yra blogai apibrėžti, nes yra labai sudėtingi. Pagal (SOUP; Svanidzaite, 2014b), paslaugų stiliaus architektūros programų sistemų projektai turi vieną ar keletą iš šių trūkumų:

- Šie projektai yra daug sudėtingesni nei tradiciniai programų sistemų projektai. Jie reikalauja didesnės, turinčios skirtingas kompetencijas komandos, taip pat ir sudėtingesnių tarpusavio komunikacijos procesų.
- Paprastai projekto pradžioje sunku apibrėžti jo apimtį, ribas ir norimą rezultatą.
- Paslaugų stiliaus architektūra gali labai teigiamai paveikti ją realizuojančią įmonę, tačiau, kita vertus, liktinių sistemų (angl. legacy software) keitimas gali labai brangiai kainuoti.

Paslaugų stiliaus architektūra susiduria su naujais iššūkiais ir problemomis programų sistemų reikalavimų inžinerijoje, todėl atsiranda nauja reikalavimų inžinerijos atšaka – paslaugų stiliaus programų sistemų reikalavimų inžinerija (angl. *SORE – service-oriented requirement engineering*). Ši reikalavimų inžinerijos atšaka skirta identifikuoti ir modeliuoti paslaugoms, jų darbų sekoms ir pakartotiniam panaudojamumui užtikrinti.

Paslaugų stiliaus programų sistemų reikalavimų inžinerija sprendžia paslaugų specifikavimo (angl. *service specification*), paslaugų suradimo (angl. *service discovery*), žinių apie paslaugas valdymo (angl. *service knowledge management*) ir paslaugų komponavimo (angl. *service composition*) uždavinius.

Nepaisant išvardytų problemų, paslaugų stiliaus architektūros programų sistemų projektai tampa vis populiaresni ir vis dažniau įgyvendinami. Todėl paslaugų stiliaus architektūrai kurti reikalingi reikalavimų rinkimo metodai ir procesai. Esti gerai ištirti, stabilūs reikalavimų inžinerijos proceso modeliai ir technikos senesnėms programų sistemų paradigmoms, tačiau paslaugų stiliaus programų sistemų reikalavimų inžinerijos procesų modeliai ir technikos vis dar yra kuriami ir vystomi (Flores, et al., 2009; Flores, et al., 2010). Jau yra pasiūlyta keletas paslaugų stiliaus architektūros programų sistemų kūrimo metodologijų ir metodų, tačiau jie nėra skirti struktūrizuoti paslaugų stiliaus reikalavimų inžinerijos procesui. Taikant šiuos metodus trūksta informacijos ir procedūrų reikalavimams išgauti ir analizuoti, todėl yra reikalingi tolesni moksliniai tyrimai.

Šioje disertacijoje gilinamasi į paslaugų specifikavimo uždavinį tiriant paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimą, analizę ir konfliktinių situacijų sprendimą. Pradedama nuo įmonės strategijos lygmens, kur verslo tikslai įvardijami, apibrėžiami ir detalizuojami, iš jų išvedami sistemos nefunkciniai reikalavimai ir jie detalizuojami tol, kol nefunkciniai reikalavimai gaunami kiekvienai įmonės sistemos paslaugai. Paprastai kiekviena suinteresuotų šalių grupė (angl. *stakeholder*), dalyvaujanti paslaugų stiliaus architektūros įmonės sistemos kūrimo projekte, turi interesų skirtingoms sistemos kokybės charakteristikoms, todėl atsiranda būtinybė derėtis ir ieškoti kompromiso sprendžiant konfliktines situacijas.

Programų sistemų reikalavimai paprastai skirstomi į funkcinius ir nefunkcinius. Funkciniai reikalavimai aprašo programų sistemos funkcionalumą, o nefunkciniai – programų sistemos funkcionalumo apribojimus. Pastarieji yra ne tokie akivaizdūs, sunkiau identifikuojami ir aptinkami. Dėl šių priežasčių nefunkciniai reikalavimai sulaukia mažiau dėmesio ir tampa kritiniai, nes paprastai programų sistemų architektūra turi atitikti įmonės verslo tikslus atitinkančius nefunkcinius reikalavimus.

Disertacijoje pateikiamas spiralinis proceso modelis, skirtas paslaugų stiliaus architektūros įmonių (angl. *ESOA – Enterprise service-oriented architecture,* SOA postilis, veikiantis ne tokioje atviroje aplinkoje (įmonės ribose) kaip SOA ir atitinkantis įmonės strategiją, misiją ir verslo tikslus) sistemų nefunkciniams reikalavimams išgauti ir analizuoti. Pateikiamas proceso modelis yra paremtas tradiciniais ir paslaugų stiliaus architektūros reikalavimų inžinerijos proceso modeliais, įmonių architektūros standartais ir karkasais (angl. *Enterprise Architecture – EA Standards and Frameworks*) ir paslaugų stiliaus architektūros lygmenimis (angl. *service-oriented architecture layers*). Proceso modelis gali būti taikomas kartu su paslaugų stiliaus architektūros programų sistemų kūrimo metodologijomis (angl. *service-oriented architecture development methodologies*).

Mokslininkų (Leite & Freeman, 1991; Sommerville & Sawyer, 1997; Russo, et al., 1999; Nuseibeh & Easterbrook, 2000) nuomone, programų sistemos reikalavimai turėtų būti renkami ir analizuojami iš skirtingų požiūrių. Analizuojant iš kurio nors vieno požiūrio daug programų sistemos aspektų nematyti, o detaliai atsispindi tik tie, kurie tiesiogiai susiję su analizuojamu požiūriu. Todėl galima daryti išvadą, kad programų sistemos

skirtingų požiūrių analizė padeda objektyviai suprasti ir specifikuoti kuriamą programų sistemą. Be to, skirtingi požiūriai gali būti taikomi ir siekiant pagerinti sistemos reikalavimų išgavimo, analizės ir konfliktų sprendimo procesus.
i* (Yu, 2009) yra karkasas (angl. *framework*), taikomas programų sistemoms modeliuoti pačioje proceso pradžioje, kai siekiama apibrėžti analizuojamą probleminę sritį. i* paremtos modeliavimo kalbos gali būti taikomos identifikuoti ir analizuoti programų sistemos reikalavimams, jiems atvaizduoti skirtinguose programų sistemos požiūriuose, be to, šios kalbos leidžia modeliuoti verslo modelius, apibrėžiančius esamą (angl. *as-is*) ir pageidaujamą, būsimą (angl. *to-be*), verslo būklę. Taip pat i* paremtomis modeliavimo kalbomis galima kurti tiek į aktorius (angl. *actors*), tiek į verslo tikslus (angl. *business goals*) orientuotus modelius. i* yra vartotojo reikalavimų notacijos (angl. *URN – User Requirements Notation*) standarto dalis. Šis standartas apibrėžia dvi vartotojo reikalavimų notacijos kalbas (Amyot & Mussbacher, 2011): tikslų reikalavimų kalbą (angl. *GRL – Goal-oriented Requirement Language*) ir panaudos atvejų žemėlapius (angl. *UCM – Use Case Map*). Tai pirmasis tarptautinis standartas, parodantis, kaip grafiškai galima sieti verslo tikslus ir programų sistemos panaudos scenarijus.
Pateikiamo paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimo ir analizės proceso modelio pagrįstumui įvertinti atliktas eksperimentinis tyrimas – atvejo analizė (angl. *case study*). Ji parodė, kad modelis tinka paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimui, analizei ir konfliktinių situacijų sprendimui.

Darbo tikslas ir uždaviniai

Atliekamo tyrimo tikslas – sukurti proceso modelį, programų sistemų analitiką įgalinantį identifikuoti, specifikuoti ir analizuoti paslaugų stiliaus architektūros įmonių sistemų nefunkcinius reikalavimus. Šis modelis paremtas tradicinės ir paslaugų stiliaus reikalavimų inžinerijos proceso modeliais, reikalavimų konfliktų sprendimo praktikomis, įmonių architektūros standartais ir karkasais. Tikslui įgyvendinti įvardyti šie tyrimo uždaviniai:

1. Įvertinti paslaugų stiliaus reikalavimų inžinerijos dabartinę būklę, kartu ir visų artimų sričių, tokių kaip: paslaugų stiliaus architektūros programų sistemų kūrimo metodologijų, įmonių architektūros standartų ir karkasų. Šis žingsnis būtinas, nes pastarieji elementai galėtų būti panaudoti nefunkciniams reikalavimams identifikuoti, apibrėžti ir reikalavimų konfliktams analizuoti.
2. Pateikti paslaugų stiliaus architektūros įmonės sistema suinteresuotų šalių sąrašą ir parodyti, kaip skiriasi šių šalių atsakomybės nuo tradicinėmis programų sistemomis suinteresuotų šalių atsakomybių.
3. Pateikti nefunkcinių reikalavimų paslaugų stiliaus architektūros įmonių sistemoms sąrašą ir parodyti, kaip skiriasi šie nefunkciniai reikalavimai nuo nefunkcinių reikalavimų tradicinėms programų sistemoms.
4. Sukurti paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimo ir analizės proceso modelį, kuris apibrėžtų nefunkcinių reikalavimų derybų procesą, ir ištirti, kaip ir kada (kokiomis pakopomis), pateiktas modelis gali būti taikomas kartu su paslaugų stiliaus architektūros programų sistemų kūrimo metodologijomis.

Tyrimo klausimai ir hipotezės

Disertacijoje keliami šie klausimai:

- Kokia yra dabartinė paslaugų stiliaus reikalavimų inžinerijos padėtis? Kokios pagrindinės problemos ir iššūkiai kyla? Kokie paslaugų stiliaus reikalavimų inžinerijos proceso modeliai jau sukurti? Ar jų branda yra pakankama siekiant užtikrinti sėkmingą paslaugų stiliaus architektūros programų sistemų kūrimą?
- Kokios paslaugų stiliaus programų sistemų kūrimo metodologijos jau yra pasiūlytos? Ar jų branda yra pakankama siekiant užtikrinti sėkmingą paslaugų stiliaus architektūros programų sistemų kūrimą? Ar gali šių metodologijų analizės ir projektavimo stadijos būti sėkmingai panaudotos paslaugų stiliaus reikalavimų inžinerijos problemoms ir iššūkiams spręsti?
- Kuo skiriasi paslaugų stiliaus architektūros programų sistemų nefunkciniai reikalavimai nuo su tradicinių programų sistemų nefunkcinių reikalavimų? Ar paslaugų stiliaus reikalavimų inžinerija ir paslaugų stiliaus architektūros programų sistemų kūrimo metodologijos pateikia sprendimus, kaip išgauti ir analizuoti nefunkcinius reikalavimus?
- Kaip tradicinės reikalavimų inžinerijos procesai ir jų modeliai gali būti taikomi paslaugų stiliaus reikalavimų inžinerijos problemoms spręsti?
- Ar galima taikyti i* paremtas modeliavimo kalbas ir požiūrius paslaugų stiliaus architektūros įmonių sistemų nefunkciniams reikalavimams analizuoti?

Norint atsakyti į suformuluotus klausimus iškeltos šios hipotezės:

1 HIPOTEZĖ. Esti paslaugų stiliaus architektūros programų sistemų kūrimo metodologijos, kurios gali būti taikomos paslaugų stiliaus reikalavimų inžinerijos proceso modeliams kurti.

2 HIPOTEZĖ. Įprasti reikalavimų inžinerijos reikalavimų rinkimo, analizės ir konfliktų sprendimo būdai iš dalies gali būti taikomi paslaugų stiliaus architektūros įmonių sistemų nefunkciniams reikalavimams rinkti, analizuoti ir konfliktams spręsti.

3 HIPOTEZĖ. i* paremtos modeliavimo kalbos ir požiūriai, plačiai taikomi įmonių architektūros standartams ir karkasams, gali būti naudojami konfliktuojantiems nefunkciniams reikalavimams rasti ir konfliktams paslaugų stiliaus architektūros įmonių sistemose spręsti.

4 HIPOTEZĖ. Paslaugų stiliaus architektūros įmonių sistemos nefunkcinių reikalavimų gavimo ir analizės modelį galima sukurti taikant tradicinės reikalavimų inžinerijos reikalavimų rinkimo ir konfliktų sprendimo būdus, technikas, i* paremtas modeliavimo kalbas ir požiūrius.

Tyrimų metodika

Šis disertacinis tyrimas yra teorinio ir empirinio pobūdžio (angl. *theoretical and empirical nature*). Paslaugų stiliaus reikalavimų inžinerija yra gana nauja programų sistemų reikalavimų inžinerijos atšaka, tad dauguma atliekamų šios srities tyrimų dar yra paviršutiniški. Tai reiškia, kad siekiant tiksliai sustruktūruoti temos problematiką ir geriau suvokti jos kontekstą, reikia atlikti nuodugnią bibliografinių šaltinių analizę. Teorinio ir empirinio pobūdžio problemai spręsti tinkamiausias yra tyrimo konstravimu

(angl. *constructive research*) metodas (Mingers, 2001). Be to, disertacijos tyrimo apimtis visuomet yra ribota tiek finansiškai, tiek laiko sąnaudų prasme. Tai reiškia, kad yra per brangu ir iš esmės neįmanoma užtikrinti tokio tyrimo statistinio patikimumo ir kiekybinių matavimų statistinio reikšmingumo, jei matavimai atliekami. Todėl nepaisant galimų paklaidų, atskiro atvejo analizė (angl. *case study*) yra vienintelis tinkamas tyrimo metodas eksperimentiškai patvirtinti šio tyrimo rezultatams.

Atsižvelgiant į pateiktus motyvus tyrimas išskaidomas į tris etapus: susijusių darbų koncepcinę analizę (angl. *conceptual analysis*) (Laurence & Margolis, 2003), tyrimą konstravimu, kurio tikslas – sukurti paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimo ir analizės proceso modelį, ir atvejo analizę, patvirtinančią ir pagrindžiančią sukurto metodo funkcionalumą.

Pirmasis etapas, koncepcinė analizė, – konceptų, terminų, kintamųjų, konstrukcijų, apibrėžimų, teiginių, hipotezių ir teorijų analizė. Koncepcinės analizės metu nagrinėjami visi šie išvardyti dalykai,siekiama aiškumo ir rišlumo, kritiškai vertinami jų loginiai ryšiai, nustatomos prielaidos ir implikacijos (Machado & Silva, 2007). Koncepcinės analizės tikslas – pagerinti nagrinėjamos srities koncepcinį aiškumą. Pagrindinis koncepcinės analizės privalumas yra tas, kad ji padeda įvertinti tiriamos srities padėtį, tad galima sistemiškai planuoti tolesnį darbą (Penrod & Hupcey, 2005). Šiame darbe koncepcinė susijusių darbų analizė atlikta siekiant suformuluoti svarbias teorines konstrukcijas ir sudaryti teorinį pagrindą tolesniems tyrimams, taip pat kad nereikėtų atlikinėti pakartotinių tyrimų (Hart, 1998). Pagrindinės sritys, kuriose koncepcinė analizė atlikta, yra šios: paslaugų stiliaus architektūra, įmonių paslaugų stiliaus architektūra, paslaugų stiliaus reikalavimų inžinerija, paslaugų stiliaus architektūros programų sistemos kūrimo metodologijos, įmonių architektūros karkasai ir standartai.

Apskritai koncepcinė analizė padėjo atsakyti į tokius klausimus kaip: kokia yra paslaugų stiliaus reikalavimų inžinerijos proceso branda, su kokiais pagrindiniais iššūkiais susiduriama šioje reikalavimų inžinerijos šakoje, ar yra sukurti kokie nors paslaugų stiliaus reikalavimų inžinerijos proceso modeliai, ar jie yra brandūs, ar gali paslaugų stiliaus architektūros programų sistemų kūrimo metodologijos drauge su įmonių architektūros karkasais ir standartais būti naudojamos spręsti pasirinktam paslaugų specifikavimo uždaviniui.

Toliau pasitelkiamas tyrimo konstravimu metodas – tyrimo procedūra, generuojanti naujų konstrukcijų, skirtų spręsti realaus pasaulio problemoms ir įnešti tam tikrą įnašą į nagrinėjamos disciplinos teoriją (Lukka, 2003; Crnkovic, 2010). Pagrindinė šio metodo notacija, nauja konstrukcija, yra abstrakti ir gali turėti daug potencialių realizacijų. Kaip konstrukcijos yra suprantami modeliai, diagramos, metodai, algoritmai ir daugelis kitų artefaktų. Tai reiškia, kad jie yra išrasti arba sukurti, o ne atrasti. Tyrimo konstravimu metodas paremtas įsitikinimu, kad kai nuodugniai analizuojama tai, kas veikia (arba neveikia) praktiškai, galima įnešti svarbų mokslinį įnašą į teoriją. Šioje disertacijoje taikant tyrimo konstravimu metodą siekiama sukurti paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimo ir analizės metodą. Nuodugnios probleminės srities analizės metu pastebėta, kad kuriamas nefunkcinių reikalavimų išgavimo ir analizės modelis gali būti grindžiamas paslaugų stiliaus architektūroje išskiriamais lygmenimis, įmonių architektūros standartais ir karkasais ir apimti 5 požiūrius: įmonės strategijos, įmonės verslo proceso, vartotojų, verslo procesų ir paslaugų.

Tyrimo konstravimu metodas taip pat naudojamas ir darbinėms hipotezėms, suformuluotoms būtent šiai disertacijai, testuoti. Vienas iš šio tyrimo metodo privalumų yra tas, kad jis suteikia galimybę ne tik testuoti ir analizuoti naujos konstrukcijos ypatybes, bet ir tyrinėti patį konstravimo procesą. Kita vertus, tyrimas konstravimu, kaip ir kiti eksperimentinio tyrimo metodai, gali būti suprantamas kaip tam tikra atskiro atvejo analizės tyrimo metodo rūšis. Pagal įprastą požiurį atskiro atvejo analizė taikytina tik hipotezėms paneigti. Ji netaikytina hipotezėms įrodyti ir priskirtina prie hipotetinio dedukcinio aiškinamojo modelio. Vis dėlto atskiro atvejo analizės atitiktis realaus pasaulio situacijoms ir jos detalumas rodo, kad šis požiūris yra tik iš dalies teisingas (Flyvbjerg, 2004). Remiantis šiuo argumentu ir faktu, kad disertacijos tyrimas yra ribotos apimties tiek finansiškai, tiek laiko sąnaudų prasme, atskiro atvejo analizės tyrimo metodas pasirinktas kaip pagrindinis hipotezėms patikrinti. Apskritai atskiro atvejo analizė yra empirinio tyrimo metodas, taikomas tirti tam tikram fenomenui ir jo kontekstui (Runeson & Höst, 2009). Šios disertacijos tikslas – pasiūlyti paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimo ir analizės proceso modelį ir ištirti nefunkcinių reikalavimų rinkimo ir analizės galimybes pagal realų pavyzdį – paslaugų stiliaus architektūros įmonės draudimo sistemą.

Rezultatai

Disertacinio tyrimo rezultatai yra šie:

- Paneigta ***1 HIPOTEZĖ,*** teigianti, kad paslaugų stiliaus architektūros programų sistemų kūrimo metodologijos gali būti taikomos paslaugų stiliaus reikalavimų inžinerijos proceso modeliams kurti. Tyrimo metu paaiškėjo, kad paslaugų stiliaus reikalavimų inžinerijos proceso modeliai gali būti taikomi kartu su paslaugų stiliaus architektūros programų sistemų kūrimo metodologijomis.
- Patvirtinta ***2 HIPOTEZĖ,*** teigianti, kad tradiciniai reikalavimų inžinerijos reikalavimų rinkimo, analizės ir konfliktų sprendimo būdai iš dalies gali būti taikomi paslaugų stiliaus architektūros įmonių sistemų nefunkciniams reikalavimams rinkti ir konfliktų spręsti. Mūsų pateikiamas paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimo ir analizės proceso modelis yra paremtas spiraliniu reikalavimų derybų proceso modeliu, priskiriamu prie tradicinės reikalavimų inžinerijos.
- Patvirtinta ***3 HIPOTEZĖ,*** teigianti, kad i* paremtos modeliavimo kalbos ir požiūriai, plačiai taikomi įmonių architektūros standartams ir karkasams, gali būti pritaikomi konfliktuojantiems nefunkciniams reikalavimams rasti bei konfliktams paslaugų stiliaus architektūros įmonių sistemose spręsti.
- Patvirtinta ***4 HIPOTEZĖ,*** teigianti, kad paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimo ir analizės modelis gali būti sukurtas taikant tradicinės reikalavimų inžinerijos reikalavimų rinkimo ir konfliktų sprendimo būdus, technikas, i* paremtas modeliavimo kalbas ir požiūrius.
- Pateikti 5 požiūriai (įmonės strategijos, įmonės verslo proceso, vartotojo, verslo procesų, paslaugų), galimi taikyti kuriant tiek paslaugų stiliaus architektūros įmonių sistemas, tiek kokio nors kito potipio paslaugų stiliaus architektūros programų sistemas.

Praktinė vertė

Pagrindinis disertacijos praktinis įnašas – paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų rinkimo ir analizės proceso modelis. Jį galima rezultatyviai derinti su paslaugų stiliaus architektūros programų sistemų kūrimo metodologijomis ir taip pateikti nuoseklų ir objektyvų metodą paslaugų stiliaus architektūros įmonių sistemoms kurti, kai pradedama nuo sistemos planavimo, analizės ir projektavimo, o baigiama sistemos diegimu ir pokyčių valdymu. Disertacijoje taip pat pateikti 5 požiūriai, galimi taikyti kuriant tiek paslaugų stiliaus architektūros įmonių sistemas, tiek kokio nors kito potipio paslaugų stiliaus architektūros programų sistemas. Tam tikru būdu adaptuoti šie požiūriai gali būti naudojami taikomi ir funkciniams reikalavimams analizuoti.

Mokslinis naujumas

Ši disertacija yra vienas iš pirmųjų tyrimų, kurių objektas – nefunkcinių reikalavimų rinkimo ir analizės galimumas paslaugų stiliaus architektūros įmonių sistemų kontekste. Nors jau buvo pateikta keletas paslaugų stiliaus architektūros programų sistemų reikalavimų inžinerijos proceso modelių (Flores, et al., 2010; Flores, et al., 2009; Flores, et al., 2008) ir paslaugų stiliaus architektūros programų sistemų kūrimo metodologijų (Papazoglou, 2006; IBM RUP/SOMA; Erl, 2005; Erl, 2008; Erradi, et al, 2006; SOUP), nė viena iš šių iniciatyvų nėra pakankamai brandi užtikrinti sėkmingą paslaugų stiliaus architektūros įmonių sistemų kūrimą. Be to, tai pirmasis darbas, kuris kelia klausimą, ar galima rinkti ir analizuoti paslaugų stiliaus architektūros įmonių sistemų nefunkcinius reikalavimus, kai taikomi požiūriai ir kai jie modeliuojami i* paremtomis modeliavimo kalbomis. Galiausiai šioje disertacijoje taikomas atskiro atvejo analizės tyrimo metodas patvirtina iškeltas hipotezes ir pateikto nefunkcinių reikalavimų rinkimo ir analizės proceso modelio funkcionalumą.

Aprobavimas ir publikacijos

Pagrindiniai disertacijos rezultatai pristatyti ir aprobuoti šiose konferencijose:

- XIX tarpuniversitetinė magistrantų ir doktorantų konferencija „Informacinė visuomenė ir universitetinės studijos - IVUS 2014", 2014 m. balandžio 24 d., Kaunas, Lietuva.
- 4-oji jaunųjų mokslininkų konferencija „Fizinių ir technologijos mokslų tarpdalykiniai tyrimai", 2014 m. vasario 11 d., Vilnius, Lietuva.
- 10-oji tarptautinė Baltijos šalių duomenų bazių ir informacinių sistemų konferencija „Baltic DB&IS 2012", 2012 m. liepos 8–11 d., Vilnius, Lietuva.
- 15-oji mokslinė Lietuvos kompiuterininkų draugijos konferencija „Kompiuterininkų dienos 2011", 2011 m. rugsėjo 22–24 d., Klaipėda, Lietuva.

Darbo apimtis

Disertaciją sudaro įvadas, 5 skyriai, išvados, bibliografijos sąrašas ir paskelbtų publikacijų sąrašas. Visa disertacijos apimtis – 212 puslapių, 27 paveikslai, 19 lentelių.

Kiekvienas skyrius pradedamas skyriaus santrauka ir baigiamas skyriaus išvadomis (išskyrus I skyrių).
I skyriuje pateikiama paslaugų stiliaus architektūros sistemų, paslaugų stiliaus architektūros įmonių sistemų ir vartotojo reikalavimų notacijos standarto kalbų analitinė apžvalga.
II skyriuje aprašomi bibliografijos šaltinių analizės rezultatai, įtraukiama paslaugų stiliaus reikalavimų inžinerijos dabartinė padėtis ir kitų artimų sričių analizė. Be to, skyriuje aprašoma, kaip renkami nefunkciniai reikalavimai taikant požiūrius.
III skyriuje formuluojami ir aptariami pagrindiniai teoriniai tyrimo rezultatai. Pristatomas paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų gavimo ir analizės proceso modelis.
IV skyriuje detaliai aprašomas atvejo analizės tyrimas ir jo rezultatai. Skyriuje modeliuojami jau tyrime apibrėžti paslaugų stiliaus architektūros įmonių sistemų požiūriai.
V skyriuje aptariami tam tikri svarbūs, bet tiesiogiai su disertacijoje atlikto tyrimo tikslu nesusiję ir todėl nenagrinėti klausimai, kurie galėtų būti tolesnių tyrimų objektu.
Rezultatuose ir išvadose pateikiami darbo rezultatai ir išvados.

1. Analitinė apžvalga

Šioje disertacijos dalyje pateikiamas glaustas disertacijoje vartojamų terminų ir susijusių konceptų aprašas. Analitinėje apžvalgoje apibrėžiama paslaugų stiliaus architektūra, esminiai jos principai, vienas iš postilių – įmonių paslaugų architektūra, esminiai paslaugų stiliaus architektūros lygmenys. Taip pat aptariamos vartotojo reikalavimų notacijos standarto kalbos.
Paslaugų paradigma (angl. *service-orientation paradigm*) leidžia kurti programų sistemas, komponuojamas iš pavienių viena nuo kitos nepriklausomų paslaugų. Paslaugų paradigma yra kilusi iš programų inžinerijos turinių atskyrimo (angl. *separation of concerns*) teorijos (Erl, 2008). Ši teorija yra grindžiama teze, kad didelę problemą naudinga suskaidyti į atskirų mažesnių, individualių problemų eilę. Tuomet kiekviena problema gali būti sprendžiama atskirai, nes yra nepriklausoma nuo kitų. Ši teorija yra pritaikoma skirtingai skirtingoms programų inžinerijos paradigmoms, pavyzdžiui, objektinė ir komponentinė paradigmos turi objektus, klases ir komponentus. O paslaugų stiliaus paradigmoje turinių atskyrimo teoriją realizuoja paslaugų principai. Galima išskirti šiuos pagrindinius paslaugų stiliaus paradigmos principus: standartizuotą paslaugos aprašą (angl. *standardized service contract*), silpną paslaugų sankibą (angl. *service loose coupling*), paslaugų abstrakciją (angl. *service abstraction*), paslaugų pakartotinį panaudojamumą (angl. *service reusability*), paslaugų autonomiją (angl. *service autonomy*), paslaugos būklės nebuvimą (angl. *service statelessness*), paslaugų surandamumą (angl. *service discoverability*) ir paslaugų komponuojamumą (angl. *service composability*).
Norint įgyvendinti paslaugų architektūros principus paslaugų architektūra yra suskirstoma į atskirus abstrakcijos lygmenis. Kiekviename abstrakcijos lygmenyje sprendžiamos tam tikros rūšies problemos. (Erl, 2005) įvardija tris pagrindinius abstrakcijos lygmenis: aplikacijos paslaugų lygmenį (angl. *application service layer*), verslo paslaugų lygmenį (angl. *business service layer*) ir paslaugų orkestravimo lygmenį

(angl. *orchestration service layer*). Kitas šaltinis – (SOA-RA, 2011), pateikia devynis skirtingus lygmenis: operacinį sistemų lygmenį (angl. *operational systems layer*), paslaugų komponentų lygmenį (angl. *service component layer*), paslaugų lygmenį (angl. *services layer*), verslo procesų lygmenį (angl. *business process layer*), vartotojo lygmenį (angl. *consumer layer*), integracijos lygmenį (angl. *integration layer*), informacijos lygmenį (angl. *information layer*), paslaugų kokybės lygmenį (angl. *quality of service layer*), valdymo ir priežiūros lygmenį (angl. *governance layer*). Atliekamame tyrime formuluojant paslaugų stiliaus architektūros įmonių sistemų požiūrius esminiai yra šie lygmenys: paslaugų lygmuo (jame įvardijamos ir apibrėžiamos visos įmonės sistemos paslaugos), verslo procesų lygmuo (jame iš pavienių paslaugų yra komponuojamas verslo procesas) ir vartotojo lygmuo (jame aprašoma vartotojo ir sistemos sąsaja).
Paslauga yra pagrindinis ir pats svarbiausias paslaugų stiliaus architektūros elementas. (Erl, 2008) siūlo paslaugas tipizuoti pagal jų paskirtį ir išskirti tris esmines paslaugų grupes: esybių paslaugos (angl. *entity services*), skirtos apibrėžti verslo esybėms, užduočių paslaugos (angl. *task services*), skirtos apibrėžti verslo procesui ir komponuoti esybių paslaugoms į verslo procesą, pagalbinės paslaugos (angl. *utility services*), skirtos apibrėžti kitai programų sistemos logikai, nesusijusiai su esybėmis ar verslo procesais. Mūsų tyrimo eksperimentinėje dalyje yra įvardijamos esybių ir užduočių paslaugos, mat šios dvi paslaugų rūšys yra pačios svarbiausios apibrėžiant programų sistemos nefunkcinius reikalavimus įmonės strateginiame ir verslo procesų lygmenyse.
Atliekamame tyrime nagrinėjamas vienas iš paslaugų architektūros postilių – įmonės paslaugų architektūra (angl. *ESOA – Enterprise service-oriented architecture*). Šis postilis nuo bendrinio paslaugų architektūros stiliaus skiriasi tuo, kad yra skirtas kurti paslaugų architektūroms neperžengiant įmonės ribų. Daug dėmesio skiriama verslo strategijai ir tikslams įgyvendinti. Taip pat paslaugų stiliaus architektūros kūrimo metu yra įvardijami ir išskiriami globalūs įmonės duomenų tipai (Sambeth, 2006), lengvinantys paslaugų stiliaus architektūros įmonių sistemų kūrimą (Bichler & Lin, 2006). Verta pridurti, kad dalis paslaugų tiekėjų ir vartotojų gali būti už įmonės ribų, tačiau jų teikiamas funkcionalumas turi būti suderinamas su vidinėmis įmonės normomis ir standartais.

2. Susijusių darbų analizė

Paslaugų stiliaus programų sistemų inžinerija (angl. *SOSE – Service-oriented Software Engineering*) yra gana nauja ir sparčiai besiplečianti programų sistemų inžinerijos atšaka. Ši disciplina atsirado paskutiniame praėjusio amžiaus dešimtmetyje (Arsanjani, 1999; Layzell, et al., 2000) kaip atsakas į nevienalyčių (angl. *heterogeneous*), liktinių programų sistemų (angl. *legacy software*) integravimo sunkumus ir kaip siekis sumažinti atotrūkį tarp greitai kintančių verslo modelių ir programų sistemų. Iš pradžių paslaugų stiliaus reikalavimų inžinerija buvo laikoma RUP – Rational Unified Process arba (Rational Software, 1998) arba IBM's Global Service's Method (Arsanjani, 2001) metodų priedu. Pirmosiose publikacijose šia tema diskutuota apie šios disciplinos pobūdį, jos skirtumus nuo tradicinės reikalavimų inžinerijos, analizuota paslaugų stiliaus reikalavimų inžinerijos reikalavimų gyvavimo ciklo struktūra ir galimi būdai išgauti ir analizuoti funkcinius ir nefunkcinius reikalavimus (Van Eck & Wieringa, 2003; Trienekens, et at., 2004).

Darbuose (Flores, et al., 2010; Flores, et al., 2009; Flores, et al., 2008) pateikta keletas paslaugų stiliaus reikalavimų inžinerijos proceso modelių. Vienas iš jų, sistematinis paslaugų stiliaus reikalavimų inžinerijos procesas (angl. *Systematic Service-oriented Requirements Engineering – SORE Process*), paremtas paslaugos kaip aukšto abstrakcijos lygmens verslo esybės, verslo veiklos paslaugos ir operacinės verslo paslaugos sampratomis. Šis proceso modelis sudarytas iš trijų pagrindinių pakopų: verslo procesų modeliavimo, kai modeliuojami verslo tikslai ir juos atitinkantys verslo procesai, verslo procesų detalizavimo ir formalaus reikalavimų specifikavimo. Tobulesnis sistematinis paslaugų reikalavimų inžinerijos proceso modelis sudarytas iš 6 pakopų: kontekstinės verslo analizės, suinteresuotų šalių poreikių aiškinimosi, reikalavimų analizės, reikalavimų modeliavimo, komunikacijos ir derybų, reikalavimų validavimo ir galutinio specifikavimo ir pokyčių valdymo. Vis dėlto abu pateikti modeliai sprendžia tik paslaugų specifikavimo uždavinį. Jie nėra detalūs, todėl negali užtikrinti kokybiško paslaugų stiliaus programų sistemų reikalavimų rinkimo.

Be to, buvo pateikta keletas paslaugų stiliaus architektūros programų sistemų kūrimo metodologijų, kuriomis siekiama užtikrinti rezultatyvų šių sistemų kūrimą ir palaikymą, taip pat perduoti jau sukauptą sistemų kūrimo ir diegimo patirtį, kaip antai (Ramollari, et al., 2007): IBM Rational Unified Process Service-oriented Modelling and Architecture (IBM RUP/SOMA), Service-Oriented Architecture Framework (SOAF) (Erradi, et al, 2006), Service-oriented Unified Process (SOUP), paslaugų analizės ir projektavimo metodologija pagal Thomasą Erlą (Erl, 2005; Erl, 2008) ir paslaugų projektavimo ir kūrimo metodologija pagal Michaelį Papazoglou (Papazoglou, 2006). Paslaugų kūrimo procesas taikant šias metodologijas suskirstytas į devynis etapus: paslaugų planavimą (angl. *service-oriented planning / inception*), paslaugų analizę (angl. *service-oriented analysis*), paslaugų projektavimą (angl. *service-oriented design*), paslaugų kūrimą (angl. *service construction*), paslaugų testavimą (angl. *service testing*), paslaugų tiekimą (angl. *service provisioning*), paslaugų diegimą (angl. *service deployment*), paslaugų vykdymą (angl. *service execution*) ir paslaugų priežiūrą (angl. *service monitoring*). Visų nagrinėjamų metodologijų detalumas yra labai nevienodas. Vienos metodologijos detaliai aprašo paslaugų stiliaus programų sistemų kūrimo procesą ir pateikia visus reikiamus artefaktus, o kitos pateikia tik gaires, tad naudotojas turi pats spręsti dėl metodologijos taikymo detalių. Pati detaliausia yra IBM RUP/SOMA metodologija, plačiai taikoma paslaugų stiliaus programų sistemoms kurti. Nors šios metodologijos yra skirtos detalizuoti ir struktūrizuoti paslaugų stiliaus programų sistemų kūrimo procesui, tačiau jos nėra skirtos reikalavimams rinkti ir analizuoti, todėl poreikis apibrėžti paslaugų stiliaus reikalavimų inžinerijos procesą išlieka.

Dėl techninių ir žmogiškųjų priežasčių sistemos reikalavimų specifikacijos niekuomet nebus tobulos. Per daugelį metų suprasta, kad reikalavimų specifikacijų netobulumai yra pagrindinė programų sistemų projektų nepavykusių diegimų priežastis (Sommerville & Sawyer, 1997). Vis dėlto, nors tobulumo ir neįmanoma užtikrinti, neabejotina, kad galima stipriai pagerinti programų sistemų reikalavimų specifikacijų kokybę.

Reikalavimų specifikacijų kokybė gali būti pagerinta dviem būdais:

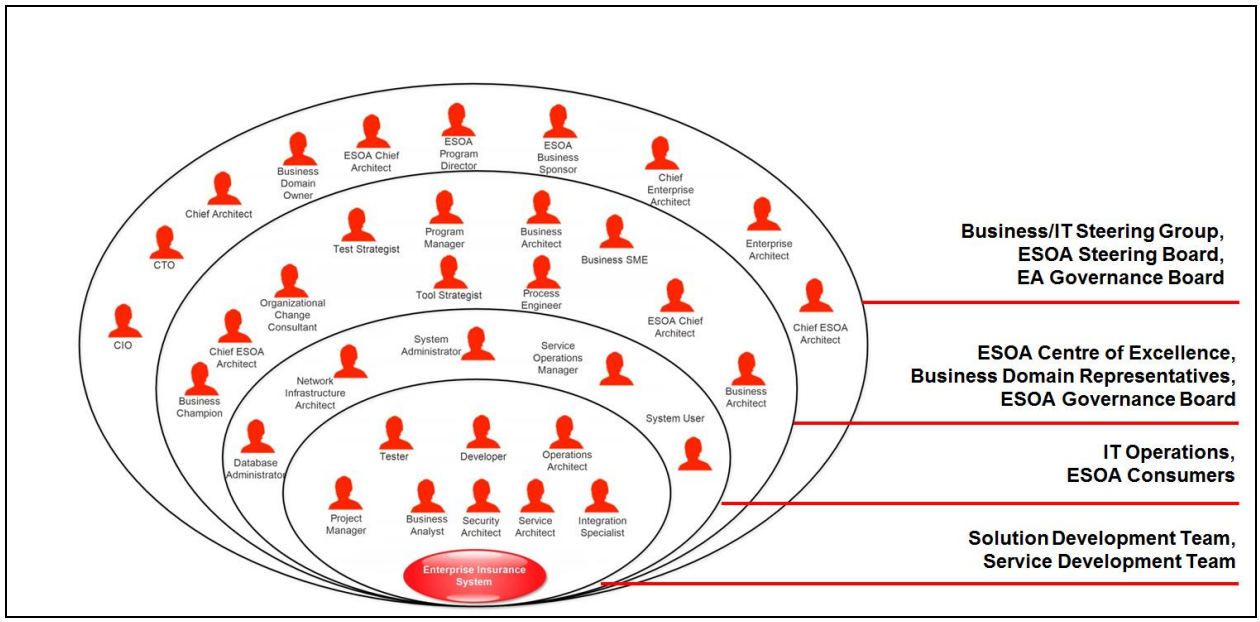
- Gerinant reikalavimų inžinerijos procesą, kad kuo mažiau klaidų atsidurtų reikalavimų specifikacijose.
- Gerinant pačios reikalavimų specifikacijos struktūrą, kad ją būtų galima lengviau skaityti, suprasti ir validuoti.

Galime pasiūlyti reikalavimų rinkimo ir analizės įrankį, kuris išnaudotų abu pateiktus būdus. Šis įrankis – požiūris (angl. *viewpoint*) – leidžia kokybiškai identifikuoti ir analizuoti skirtingus programų sistemų reikalavimus skirtinguose abstrakcijos lygmenyse. Požiūris yra esybė, plačiai taikoma programų sistemų architektūrai aprašyti, tačiau gali būti taikoma ir susisteminti programų sistemų funkciniams ir nefunkciniams reikalavimams. Skirtingi požiūriai gali būti taikomi turiniams atskirti, skirtingų suinteresuotų šalių interesams atskirti, tačiau sujungti visi suformuluoti požiūriai atskleidžia visuminį programų sistemos vaizdą. Pagal tarptautinį standartą (ISO/IEC/IEEE 42010:2011), kiekvienas požiūris turi nurodyti: vieną arba daugiau suinteresuotų šalių ir jų interesų, išreiškiamų per požiūrį, požiūrio modeliavimo būdus (angl. *model kinds*), tai gali būti įvairios kalbos, notacijos, susitarti žymenys ir nuorodas į požiūrio šaltinius.
Yra sukurta keletas tradicinės ir paslaugų stiliaus architektūros programų sistemų architektūros kūrimo karkasų (angl. *framework*), kurie programų sistemų architektūrai analizuoti taiko požiūrius, pvz: SOA-RAF – OASIS Reference Architecture Foundation for SOA (SOA-RAF, 2012), Zachman Enterprise Architecture Framework (Zachman), The Open Group Architecture Framework – TOGAF (TOGAF), Extended Enterprise Architecture Framework (E2AF), Department of Defence Architecture Framework (DoDAF), Kruchten's "4+1"/RUP's 4 + 1 View Model (Kruchten, 1995; Kroll, et al., 2003), Siemens 4 views method (S4V) (Hofmeister, et al., 2000; Soni, et al., 1995), Reference Model for Open Distributed Processing (RM-ODP). Šie architektūros karkasai gali praversti apibrėžiant reikalavimų rinkimo ir analizės požiūrius, vis dėlto analizė parodė, kad jų detalumas yra labai nevienodas. Pats detaliausias yra TOGAF (jo atžvilgiu atlikta visų kitų karkasų analizė). Keletas karkasų, tokių kaip: Kruchten's "4+1"/RUP's 4 + 1 View Model, Siemens 4 views yra veikiau techniniai, bet esama ir karkasų, labiau orientuotų į verslo strategijos ir tikslų analizę, pvz.: Zachman Enterprise Architecture Framework, Extended Enterprise Architecture Framework.

3. Spiralinis proceso modelis paslaugų stiliaus architektūros įmonių sistemų nefunkciniams reikalavimams išgauti ir analizuoti

Programų sistemų kūrimo kontekste suinteresuotosios šalys yra tie asmenys ar asmenų grupės, kurie turi daugiausia įtakos šių sistemų kūrimui. Kad verslo reikalavimai būtų įvykdyti efektyviai, kuriant paslaugų stiliaus architektūros įmonės sistemą visos suinteresuotos šalys turi būti identifikuotos nuo pat projekto pradžios. Paslaugų stiliaus architektūros įmonių sistemų suinteresuotos šalys skiriasi nuo tradicinių programų sistemų suinteresuotų šalių keliais aspektais: joms suteikiamos naujos rolės ir atsakomybės, be to, šiems projektams įgyvendinti reikia daugiau valdymo ir priežiūros pareigybių. Galima atskirti paslaugų kūrimą ir jų komponavimą: paslaugų kūrimą patikėti vienai grupei, o paslaugų komponavimą – kitai grupei darbuotojų. Šaltiniai (SOA GRM; TOGAF 9.1, 2011) siūlo tokias suinteresuotų šalių pagrindines grupes (žr. *1 paveikslas*): verslo / IT valdymo grupė (angl. *Business / IT Steering Group*), ESOA valdymo komisija (angl. *ESOA Steering Board, ESOA Governance Board*), įmonės architektūros valdymo komisija (angl. *EA Governance Board*), ESOA kompetencijos centras (angl. *ESOA Center of Excellence*), verslo srities atstovai (angl. *Business Domain*

Representatives), programinio sprendimo kūrimo grupė (angl. *Solution Development Team*), paslaugų kūrimo grupė (angl. *Service Development Team*), IT operacijų priežiūros grupė (angl. *IT Operations*), ESOA sistemų vartotojai (angl. *Consumers Group*).



1 paveikslas. Įmonių paslaugų architektūros suinteresuotų asmenų grupės

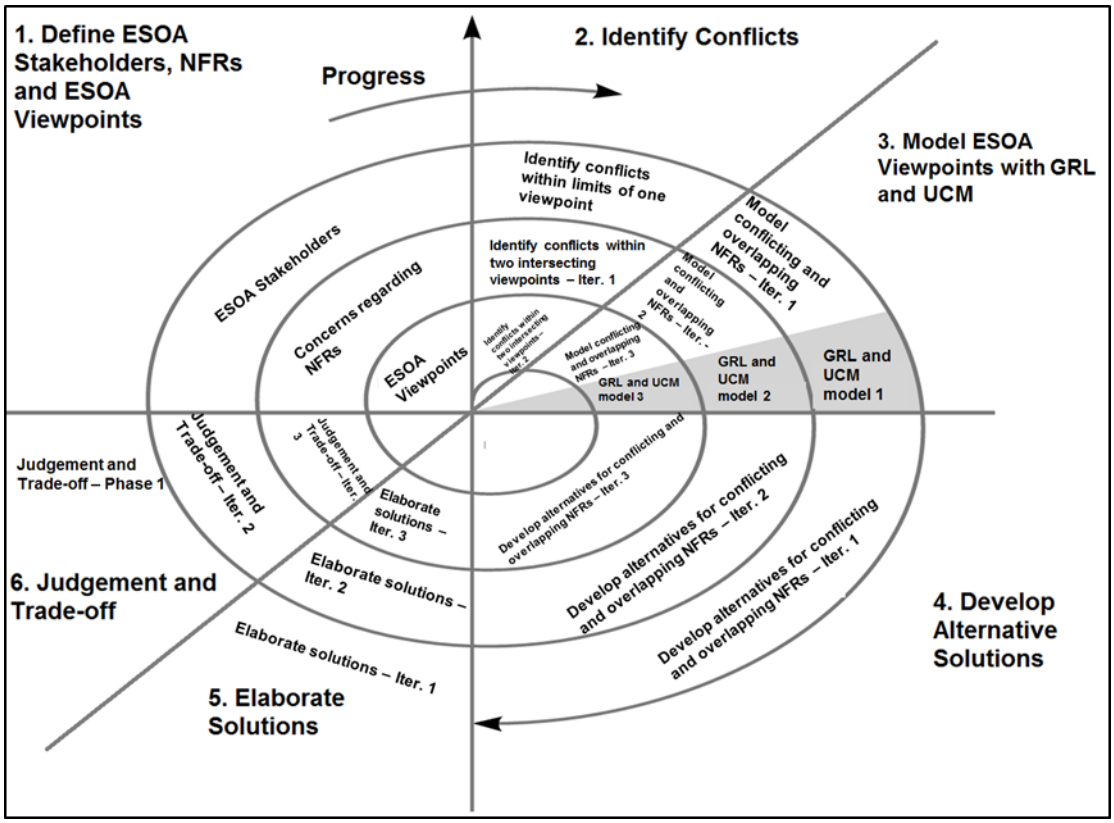
Paslaugų stiliaus architektūros programų sistemų nefunkciniai reikalavimai yra paveldėti iš tradicinių programų sistemų. Pagrindinis skirtumas yra pats nefunkcinio reikalavimo apibrėžimas ir jo matas (angl. *metric*). Tradicinėse programų sistemose nefunkcinis reikalavimas apibrėžia vieną visos programų sistemos kokybinę charakteristiką, o paslaugų stiliaus architektūroje nefunkcinis reikalavimas, visų pirma, apibrėžia paslaugą, kaip niekam nepriklausantį vienetą, charakteristiką ir tik paskui paslaugų kompozicijos kokybinę charakteristiką. Paslaugų stiliaus architektūroje nefunkciniai reikalavimai yra glaudžiai susiję su esminiais paslaugų stiliaus paradigmos principais (žr. skyrelį: *1 Analitinė apžvalga*). Kiekvienas principas realizuoja kokią nors paslaugų stiliaus programų sistemos kokybinę charakteristiką. Išskiriami tokie esminiai paslaugų stiliaus architektūros įmonių sistemų nefunkciniai reikalavimai (Choi, et al., 2007; O'Brien, et al., 2005): pasiekiamumas (angl. *availability*), našumas (angl. *performance*), patikimumas (angl. *reliability*), panaudojamumas (angl. *usability*), randamumas (angl. *discoverability*), adaptuojamumas (angl. *adaptability*), komponuojamumas (angl. *composability*), interoperabilumas (angl. *interoperability*), saugumas (angl. *security*), skaliuojamumas (angl. *scalability*), plečiamumas (angl. *extensibility*), testuojamumas (angl. *testability*), auditas (angl. *auditability*), keičiamumas (angl. *modifiability*), operabilumas (angl. *operability and deployability*). Šie reikalavimai mūsų formuluojamuose paslaugų stiliaus architektūros įmonių sistemų požiūriuose traktuojami kaip interesai (angl. *concerns*).

Remdamiesi paslaugų stiliaus reikalavimų inžinerijos analizės rezultatais, pristatytais ankstesniuose skyreliuose, pateikiame spiralinį proceso modelį paslaugų stiliaus architektūros įmonių sistemų nefunkciniams reikalavimams išgauti ir analizuoti. Šis modelis yra paremtas pagrindiniu paslaugų stiliaus paradigmos principu – kurti

programų sistemas, kurios atitiktų įmonių verslo strategijas ir tikslus. Todėl siekiama rasti atsakymą, kodėl (modeliuojant verslo tikslus) vieni nefunkciniai reikalavimai yra svarbesni už kitus. Pristatome iteratyvų ir spiralinį reikalavimų derybų modelį (žr. *2 paveikslas*), kuris yra paremtas spiraliniu reikalavimų derybų modeliu pagal (Ahmad, 2008; Boehm, 1988; Boehm, 2000). Pateikiamas modelis išnaudoja reikalavimų derybų iteratyvumo privalumą ir leidžia kartotines derybas. Kiekviena reikalavimų derybų iteracija leidžia išspręsti vis sudėtingesnius reikalavimų konfliktus ir gauti geresnį derančių reikalavimų rinkinį.

Pirmoji reikalavimų derybų modelyje apibrėžta veikla yra įvardyti paslaugų stiliaus architektūros įmonės sistema suinteresuotus asmenis, jų keliamus nefunkcinius reikalavimus ir įmonės sistemos požiūrius. Veiklos įeigai pasirenkame suinteresuotas šalis, įvardytas šiame skyrelyje, suinteresuotų šalių interesus nefunkciniams reikalavimams. Veiklos išeiga yra aiškiai įvardyti ir apibrėžti paslaugų stiliaus architektūros įmonės sistemų požiūriai (jų aprašas sudarytas iš šių dalių: suinteresuotos šalies įvardijimo, suinteresuotos šalies intereso nefunkciniam reikalavimui, suformuluoto nefunkcinio reikalavimo ir jo prioriteto). Kita veikla yra konfliktuojančių reikalavimų identifikavimas. Ši veikla atliekama tokiomis iteracijomis:

- Jeigu paslaugų stiliaus architektūros įmonės sistemos požiūris turi daugiau negu vieną suinteresuotą šalį, visų pirma ieškome reikalavimų konflikto požiūrio ribose – lyginame skirtingų suinteresuotų šalių iškeltus nefunkcinius reikalavimus. Palyginimui naudojame į Quality Function Deployment – QFD metodą panašų lentelių metodą (Sommerville & Sawyer, 1997; Errikson & McFadden, 1993). Vienos suinteresuotos šalies nefunkciniai reikalavimai yra vaizduojami kaip eilutės, kitos – kaip stulpeliai. Įvertiname kiekvieną eilutės ir stulpelio sankirtą pažymėdami, ar reikalavimai susikerta, ar konfliktuojantys, ar nepriklausomi (žr. *3 paveikslas*). Jeigu randami susikertantys arba konfliktuojantys reikalavimai, tokiam požiūriui braižome GRL ir UCM diagramas.
- Kai skirtingų suinteresuotų šalių nefunkcinių reikalavimų konfliktai yra išspręsti požiūryje, ieškome reikalavimų konfliktų tarp skirtingų požiūrių. Paieškai taikome tą patį lentelių metodą, tačiau vieno požiūrio nefunkcinius reikalavimus vaizduojame kaip eilutes, kito – kaip stulpelius. Jeigu randami reikalavimų konfliktai braižomos GRL ir UCM diagramos, apimančios analizuojamus požiūrius.
- Šie žingsniai kartojami tol, kol išanalizuojami visi požiūriai.



2 paveikslas. Paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų derybų spiralinis modelis

		Stakeholder Group 1		
		REQ1	REQ2	REQ3
Stakeholder Group 2	REQ1	0	10	1
	REQ2	1	0	0
	REQ3	0	1	10

3 paveikslas. Lentelių metodas nefunkcinių reikalavimų sankirtoms analizuoti (nepriklausomi reikalavimai žymimi „0“, susikertantys – „10“, konfliktuojantys – „1“)

Kai reikiamos GRL ir UCM diagramos nubrėžiamos, suinteresuotos šalys ima ieškoti alternatyvių sprendimų (formuluoti alternatyvius nefunkcinius reikalavimus). Pateikti alternatyvūs reikalavimai yra tobulinami atsižvelgiant į kiekvienos suinteresuotos šalies nuomonę. Galiausiai nuprendžiama remiantis vienu ar keliais sprendimo priėmimo kriterijais: projekto grafiko, sistemos kainos, sistemos funkcionalumo, technologijų galimybių. Pavyzdžiui, gali būti remiamasi bendradarbiavimo strategija, apibrėžiančia, kad turi būti atsižvelgta į visus suinteresuotų šalių interesus nefunkciniams reikalavimams. Šiuo atveju randamas sprendimas (įmonės sistemos nefunkcinių reikalavimų rinkinys), kai atsižvelgiama į minimalų visų suinteresuotų šalių interesų nefunkciniams reikalavimams skaičių.

Paslaugų stiliaus architektūros įmonių sistemų požiūriai buvo suformuluoti remiantis paslaugų stiliaus architektūros sluoksniais, įmonių architektūros standartais ir karkasais, aptartais ankstesniuose skyreliuose. Pateikiame penkis požiūrius – vieną įmonės strategijos, vieną įmonės verslo proceso ir tris architektūrinius požiūrius. Šis požiūrių

rinkinys leidžia numatyti objektyvų kuriamos įmonės sistemos pavidalą: pradedama nuo verslo strategijos, ji detalizuojama atsižvelgiant į visuminį verslo procesą, o toliau detalizuojami abstraktūs verslo reikalavimai įmonės sistemos vartotojo, verslo procesų ir konkrečių paslaugų reikalavimams. Tai šie požiūriai:

- **Įmonės strategijos požiūris** (angl. *Enterprise Strategy Viewpoint*), apibrėžiantis įmonės misiją, viziją ir strategiją vizijai pasiekti. Šiuo požiūriu suinteresuotų šalių grupė – verslo / IT valdymo grupė (angl. *Business / IT Steering Group*). Šį požiūrį modeliuoti siūloma šiomis technikomis: įmonės misijos ir vizijos teiginiais, PEST ir SWOT analize. Požiūris yra verslo strategijos pobūdžio ir neapibrėžia jokių įmonės sistemos nefunkcinių reikalavimų.
- **Įmonės verslo proceso požiūris** (angl. *Enterprise Business Processes Viewpoint*) apibrėžia įmonės veiklos sritį ir verslo modelį, nesiejamą su kuriama sistema. Šiuo požiūriu suinteresuotų šalių grupė – verslo / IT valdymo grupė (angl. *Business / IT Steering Group*). Šį požiūrį modeliuoti siūloma šia technika – verslo procesų modeliavimo notacija (angl. *BPMN – Bussiness Process Modelling Notation*). Požiūris yra verslo strategijos pobūdžio ir neapibrėžia jokių programų sistemos nefunkcinių reikalavimų.
- **Vartotojo požiūris** (angl. *Consumer Viewpoint*) apibrėžia vartotojo ir įmonės sistemos santykį. Šiuo požiūriu suinteresuotų šalių grupės: ESOA sistemų vartotojai (angl. *Consumers Group*) ir verslo srities atstovai (angl. *Business Domain Representatives*). Šį požiūrį modeliuoti siūloma GRL ir UCM diagramomis. Šiuo požiūriu yra analizuojamos šios programų sistemų kokybinės charakteristikos: pasiekiamumas / prieinamumas, našumas, naudojamumas, patikimumas, saugumas, skaliuojamumas, audito galimybės.
- **Verslo procesų požiūris** (angl. *Business Process Viewpoint*) apibrėžia įmonės sistema skaitmenizuojamus verslo procesus. Paslaugų stiliaus architektūroje verslo procesas sudaromas iš paslaugos-kontrolerio ir orkestruojamų paslaugų (paslaugų orkestracija), arba iš tarpusavyje santykiaujančių paslaugų rinkinio (paslaugų choreografija). Šiuo požiūriu suinteresuotų šalių grupės: verslo srities atstovai (angl. *Business Domain Representatives*), įmonės architektūros valdymo komisija (angl. *EA Governance Board*), ESOA kompetencijos centras (angl. *ESOA Center of Excellence*), ESOA valdymo komisija (angl. *ESOA Steering Board, ESOA Governance Board*) ir programinio sprendimo kūrimo grupė (angl. *Solution Development Team*). Šį požiūrį modeliuoti siūloma GRL ir UCM diagramomis. Šiuo požiūriu analizuojamos šios programų sistemų kokybinės ypatybės: randamumas / ieškomumas, adaptuojamumas, komponuojamumas, interoperabilumas.
- **Paslaugų požiūris** (angl. *Service Viewpoint*) apibrėžia visų paslaugų stiliaus architektūros įmonės sistemą sudarančių paslaugų aprašus (angl. *service descriptions*). Šiuo požiūriu suinteresuotų šalių grupės: verslo srities atstovai (angl. *Business Domain Representatives*), įmonės architektūros valdymo komisija (angl. *EA Governance Board*), ESOA kompetencijos centras (angl. *ESOA Center of Excellence*), ESOA valdymo komisija (angl. *ESOA Steering Board, ESOA Governance Board*) ir paslaugų kūrimo grupė (angl. *Service Development Team*). Šį požiūrį modeliuoti siūloma GRL ir UCM diagramomis. Šiuo požiūriu analizuojamos šios programų sistemų kokybinės charakteristikos: pasiekiamumas

/ prieinamumas, adaptuojamumas, našumas, panaudojamumas, patikimumas, saugumas, komponuojamumas, modifikuojamumas, praplečiamumas, skaliuojamumas, testuojamumas.

Verta paminėti, kad mūsų pateikiamas paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų rinkimo ir analizės modelis gali būti taikomas kartu su paslaugų stiliaus architektūros programų sistemų kūrimo metodologijomis:

- Taikant IBM RUP/SOMA metodologiją modelis gali būti taikomas verslo transformacijos analizės (angl. *Business Transformation Analysis*) stadijoje.
- Taikant paslaugų analizės ir projektavimo metodologiją pagal Thomasą Erlą, pateikiamas modelis gali būti taikomas paslaugų analizės (angl. *Service-Oriented Analysis*) stadijoje.
- Taikant paslaugų projektavimo ir kūrimo metodologiją pagal Papazoglou, pateikiamas modelis gali būti taikomas paslaugų analizės (angl. *Service-Oriented Analysis*) stadijoje.
- Taikant paslaugų architektūros karkasą (SOAF), pateikiamas modelis gali būti taikomas informacijos išsiaiškinimo/išgryninimo (angl. *Information Elicitation*) stadijoje.
- Taikant paslaugų unifikuotą procesą (SOUP) pateikiamas modelis gali būti taikomas apibrėžimo / įvardijimo (angl. *Define*) stadijoje.

4. Atvejo analizė: paslaugų stiliaus architektūros įmonės draudimo sistema

Šiame disertacijos skyriuje pateiktas sukurto modelio eksperimentinis tyrimas – atvejo analizė (angl. *case study*). Siekdami iliustruoti sukurto paslaugų stiliaus architektūros įmonės sistemos nefunkcinių reikalavimų rinkimo ir analizės modelio veikimo principus, pasirinkome draudimo paslaugų verslo sritį. Projektuosime draudimo paslaugas (konkrečiau, transporto priemonių civilinės atsakomybės draudimo arba turto draudimo) teikiančios įmonės (įmonės pavadinimas – *Insurance4You*) paslaugų stiliaus architektūros sistemą. Įmonės *Insurance4You* paslaugų stiliaus architektūros sistema sudaryta iš keturių pagrindinių posistemių: klientų duomenų valdymo (angl. *CRM – Customer Relationship Management Sub-system*), draudimo polisų kūrimo ir valdymo (angl. *Policy Sub-system*), sąskaitų ir draudimo įmokų valdymo (angl. *Billing Sub-system*), draudiminių įvykių registravimo (angl. *Claims Sub-system*). Suformuluoti ir apibrėžti sukurtame modelyje pateikti paslaugų stiliaus architektūros įmonės sistemos požiūriai:

Įmonės strategijos požiūris (angl. *Enterprise Strategy Viewpoint)* įvardija įmonės misiją ir viziją. Jame atliekama verslo PEST ir SWOT analizė.

Įmonės misija: Insurance4You – globali vidutinio dydžio draudimo įmonė, teikianti vienas iš kokybiškiausių transporto priemonių civilinės atsakomybės draudimo ir gyventojų turto draudimo paslaugų. Įmonė turi daugiau nei 35 milijonus klientų visame pasaulyje, laikoma pačiu geriausiu draudiku Jungtinėse Amerikos Valstijose ir Kanadoje ir aktyviai vysto savo veiklą Lotynų Amerikoje, Japonijoje, Kinijoje, Europoje ir Afrikos žemyne.

Pateikiama įmonės išorės veiksnių PEST analizė (žr. *1 lentelė*).

1 lentelė. Įmonės Insurance4You išorės veiksnių PEST analizė

Politiniai veiksniai Nestabili politika Rusijoje ir Ukrainoje.	**Ekonominiai veiksniai** Ekonominis nuosmukis Europoje, Japonijoje ir Kinijoje. Ebolos virusas Afrikos žemyne. Rusijos susitraukusi ekonomika, rublio kurso svyravimai. Mažėjantis EUR / USD poros kursas ir galimas euro ir dolerio paritetas 2016–2017 metais ar net anksčiau. Galimas Jungtinių Amerikos Valstijų palūkanų normų augimas nuo antro ar trečio 2015 m. ketvirčio. Nulinė Euro zonos palūkanų norma bent iki 2017 m.
Socialiniai veiksniai Sumažėjusi gyventojų populiacija ir senstanti visuomenė Jungtinėse Amerikos Valstijose, Kanadoje ir Europoje. Didėjanti gyventojų populiacija Indijoje ir Afrikoje. Auganti vidurinioji gyventojų klasė Indijoje ir Kinijoje.	**Technologiniai veiksniai** Augantis išmaniųjų telefonų ir aplikacijų naudojimas. 3G, 4G LTE ir 4G mobiliųjų tinklų plėtra.

SWOT įmonės vidaus veiksnių analizė (žr. *2 lentelė*):

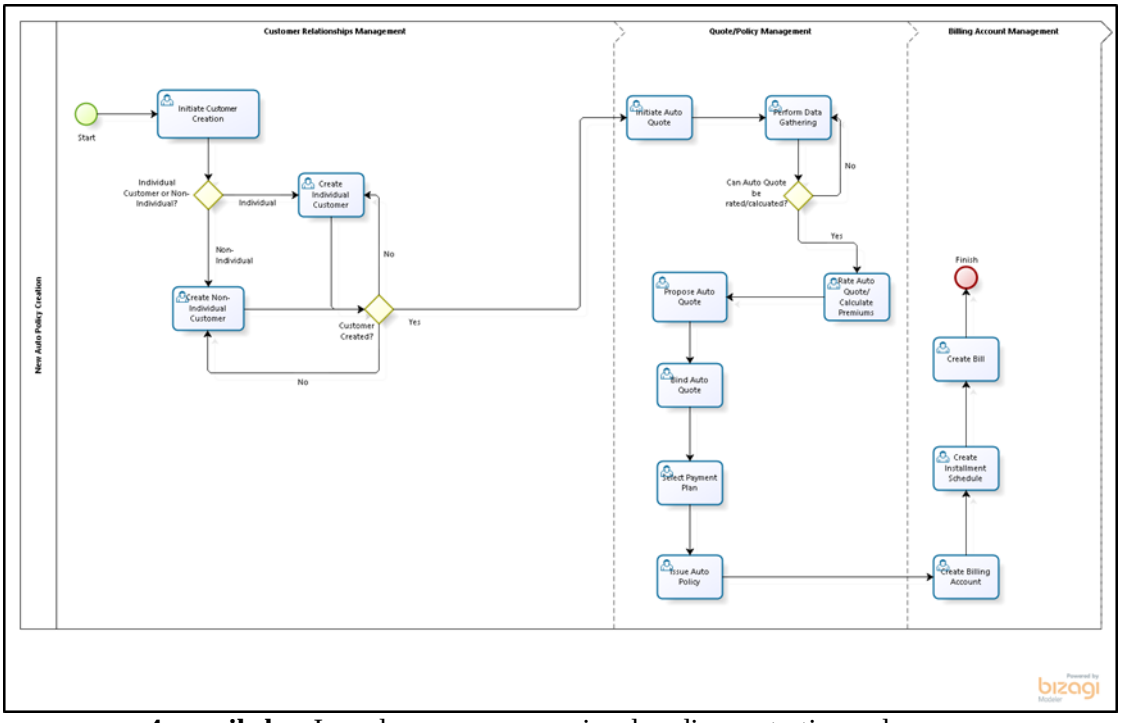
2 lentelė. Įmonės Insurance4You SWOT vidaus veiksnių analizė

Įmonės stiprybės Įmonė laikoma pačiu geriausiu draudiku, teikiančiu kokybiškiausias paslaugas Jungtinėse Amerikos Valstijose ir Kanadoje. Įmonė laikoma geriausiu darbdaviu, turinčiu geriausias darbuotojų motyvacijos ir kvalifikacijos kėlimo schemas Jungtinėse Amerikos Valstijose.	**Įmonės silpnybės** Sunku atnaujinti ir palaikyti liktinės įmonės programų sistemas. Nėra vartotojų savitarnos per internetinę svetainę ar mobiliąsias programėles.
Įmonės galimybės Įmonės plėtra Europoje, Azijoje, Lotynų Amerikoje ir Afrikoje. Geresnė teikiamų paslaugų kokybė ir geresnės klientų pasiekimo galimybės įdiegus naują paslaugų architektūros programų sistemą ir savitarnos taškus. Augantis klientų skaičius Indijoje dėl augančio gyventojų skaičiaus. Augantis klientų skaičius Kinijoje dėl augančios viduriniosios gyventojų klasės.	**Įmonės grėsmės** Kitos draudimo įmonės Jungtinėse Amerikos Valstijose, Europoje ir Azijoje, jau įdiegusios savitarną. Mažėjantis klientų skaičius Europoje ir Azijoje dėl įsisukančios defliacijos. Mažėjantis klientų skaičius Europoje, Jungtinėse Amerikos Valstijose, Kanadoje ir Kinijoje dėl senstančios visuomenės.

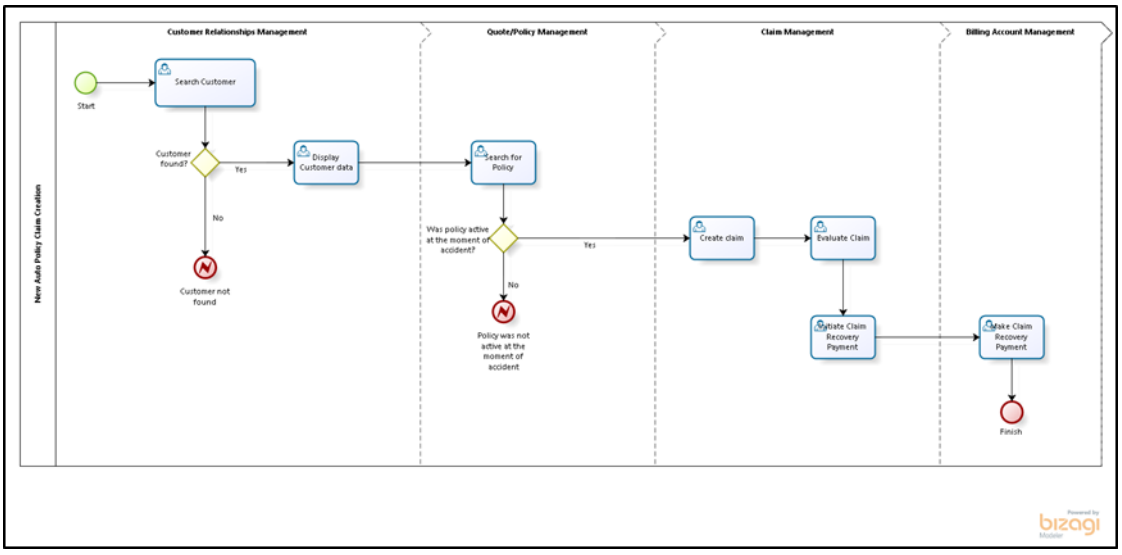
Įmonės vizija – Insurance4You yra draudimo paslaugų lyderis pasaulinėje rinkoje, teikiantis pačias kokybiškiausias paslaugas ir puikias klientų savitarnos galimybes, turintis 150 mln. klientų visame pasaulyje.

Įmonės verslo proceso požiūriu (angl. *Enterprise Business Processes Viewpoint*) apibrėžėme du pagrindinius įmonės vykdomus verslo procesus, apimančius visas kuriamos paslaugų stiliaus architektūros įmonės sistemos posistemes.

I Verslo procesas (žr. *4 paveikslas*) apibrėžia naujo kliento užregistravimą sistemoje, draudimo sutarties sudarymą, sutarties kainos ir draudimo įmokų suskaičiavimą.
II Verslo procesas (žr. *5 paveikslas*) apibrėžia kliento draudiminio įvykio užregistravimą sistemoje. Draudiminis įvykis įvertinamas ir išmokama draudimo išmoka.



4 paveikslas. I verslo procesas – naujos draudimo sutarties sudarymas



5 paveikslas. II verslo procesas – draudiminio įvykio registravimas

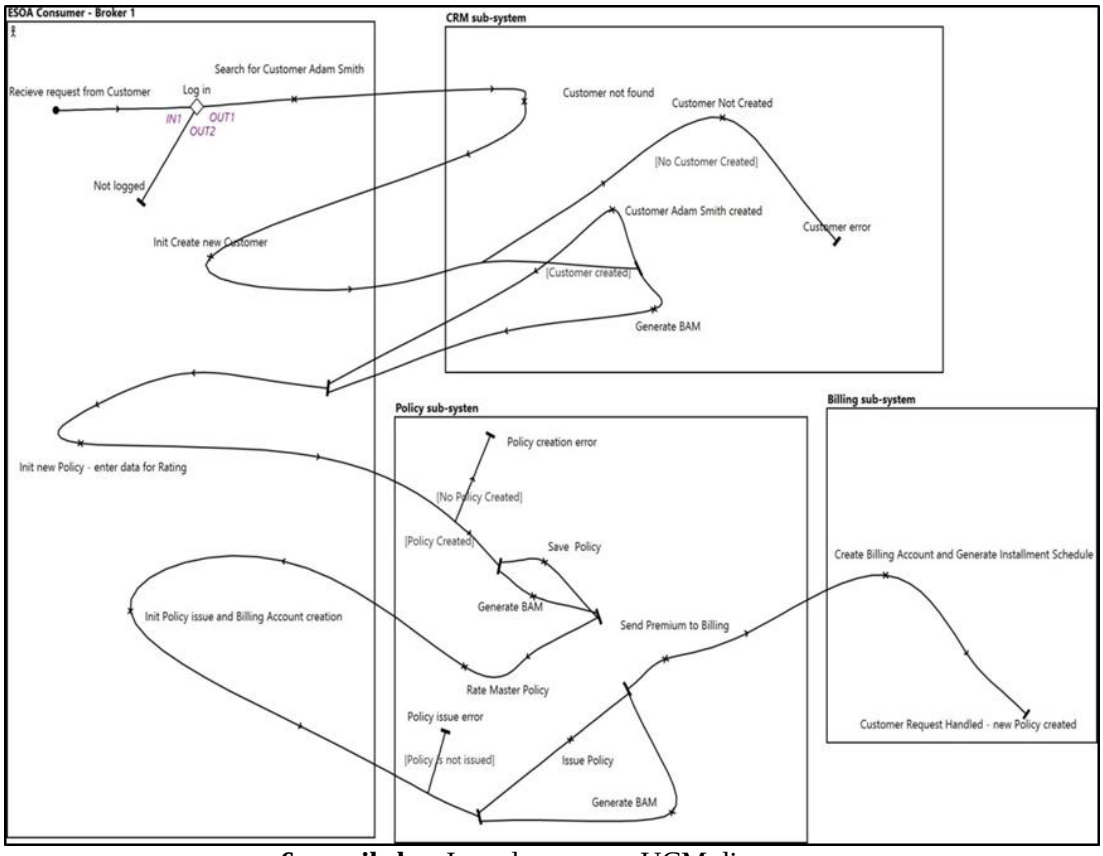
Vartotojo požiūriu (angl. *Consumer Viewpoint*) identifikuojami ir analizuojami dviejų suinteresuotų šalių: sistemos vartotojų ir verslo srities atstovų nefunkciniai reikalavimai įmonės sistemai. Nefunkcinių reikalavimų apibrėžimo pavyzdys pateiktas *3 lentelėje*.
Verslo procesų požiūriu (angl. *Business Process Viewpoint*) įvardijamos sistemos paslaugos, kuriomis skaitmenizuojami anksčiau apibrėžti verslo procesai. Šiuo požiūriu įvardijami suinteresuotų šalių nefunkciniai reikalavimai verslo procesams.

Paslaugų požiūriu (angl. *Service Viewpoint*) įvardijamos sistemos paslaugos, kuriomis skaitmenizuojami anksčiau apibrėžti verslo procesai. Šiuo požiūriu įvardijami suinteresuotų šalių nefunkciniai reikalavimai konkrečioms paslaugoms.

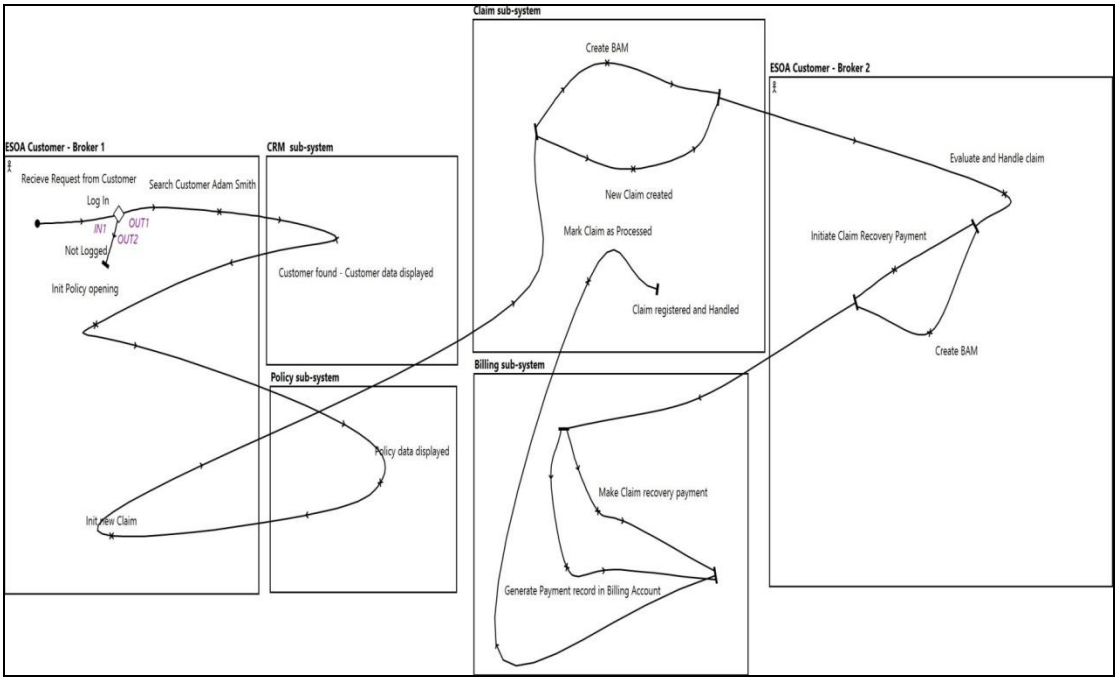
3 lentelė. Nefunkcinių reikalavimų apibrėžimo pavyzdys iš vartotojo požiūrio

Vartotojo interesas	**Reikalavimų kategorija**	**Reikalavimas**	**Prioritetas**
Suinteresuota šalių grupė: Paslaugų architektūros programų sistemų vartotojai			
Sistemos prieinamumas darbo ir nedarbo metu	Prieinamumas / Pasiekiamumas (angl. *Availability*)	**CUSTCA1**. Sistema turi būti pasiekiama darbo valandomis nuo 8 iki 23 valandos. **CUSTCA2**. Sistema turi būti pasiekiama nedarbo metu įvairiems automatiniams procesams (tokiems kaip ataskaitų generavimas, įmokų generavimas ir t. t.) vykdyti.	1

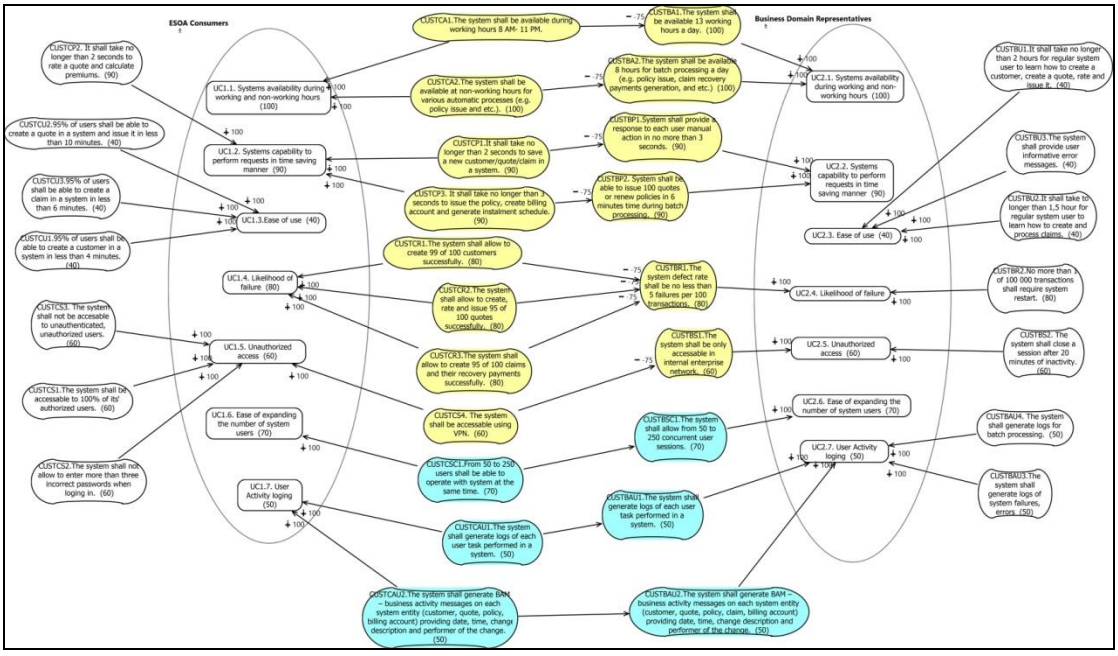
Apibrėžus išvardytus paslaugų stiliaus architektūros įmonės sistemų požiūrius, vykdyta antroji modelio veikla – konfliktuojančių nefunkcinių reikalavimų radimas. Reikalavimų konfliktams analizuoti sudarytos UCM (žr. *6, 7 paveikslas*) ir GRL diagramos (žr. *8 paveikslas*).



6 paveikslas. I verslo proceso UCM diagrama



7 paveikslas. II verslo proceso UCM diagrama



8 paveikslas. Vartotojo požiūrio GRL diagrama

Sudarius kiekvieno požiūrio GRL ir UCM diagramas, atliktos kitos sukurtame modelyje įvardytos veiklos. Apibendrinant, atliktas eksperimentas – atvejo analizė – parodė, kad sukurtas nefunkcinių reikalavimų rinkimo ir analizės modelis gali būti taikomas rinkti paslaugų stiliaus architektūros įmonių sistemų nefunkciniams reikalavimams. Kita vertus, modelis turi keletą trūkumų ir ribotumų, plačiau juos aptarsime kitame skyrelyje.

5. Neatsakyti klausimai ir tyrimo kontekstas

Atlikto disertacinio tyrimo tikslas – pateikti proceso modelį paslaugų stiliaus architektūros įmonių sistemų nefunkciniams reikalavimams rinkti ir analizuoti. Šis

modelis padėtų spręsti paslaugų stiliaus reikalavimų inžinerijoje keliamą paslaugų specifikavimo uždavinį. Mūsų pateikiamas proceso modelis reikalavimams išgauti ir analizuoti taiko požiūrius, įgalinančius pradėti reikalavimų analizę nuo įmonės strategijos lygmens ir detalizuoti reikalavimą iki sistemos paslaugos lygmens. Todėl siūloma konfliktuojančius reikalavimus analizuoti ir modeliuoti pasitelkiant GRL ir UCM kalbas, nes jos leidžia modeliuoti verslo tikslus, sieti juos su sistemos reikalavimais ir matyti, kaip skirtingos įmonės sistemos posistemės ar komponentai juos realizuos. Nors tyrimo rezultatai yra gana išsamūs, yra keletas proceso modelio taikymo realiems sistemų projektams apribojimų:

- Pateikiamas nefunkcinių reikalavimų rinkimo ir analizės proceso modelis gali būti sunkiai pritaikomas, jeigu verslo strategija, verslo tikslai nėra visiškai aiškūs ir jeigu nėra įmanoma tiesiogiai susieti verslo tikslo su sistemos funkcija. Mat pagrindinis proceso modelio tikslas ir yra atrinkti tas sistemos kokybines charakteristikas, kurios geriausiai atitiktų verslo tikslus.
- Metodo taikymas reikalauja daug rankų darbo: reikia aprašyti požiūrius, įtraukti kiekvienos požiūriu suinteresuotos šalies nefunkcinius reikalavimus, juos prioretizuoti, rasti nefunkcinių reikalavimų konfliktus, modeliuoti juos GRL ir UCM notacijomis, vykdyti reikalavimų derybas iteracijomis.

Šie proceso modelio taikymo ribotumai didina žmogiškųjų klaidų tikimybę. Todėl vertėtų pasvarstyti apie galimą pateikiamo modelio automatizavimą. Taip pat neištirta, ar siūlomas modelis galėtų būti taikomas tradicinių programų sistemų nefunkciniams reikalavimams rinkti ir analizuoti. Manoma, kad šiuo atveju reikėtų kiek pakeisti modelio požiūrius, adaptuoti suinteresuotų šalių ir jų atsakomybių aprašus, išmesti paslaugų stiliaus architektūros principus iš nefunkcinių reikalavimų ir paties modelio apibrėžimo. Be to, neištirta, ar siūlomas modelis galėtų būti taikomas įmonių sistemų funkciniams reikalavimams analizuoti. Manoma, kad visų pirma reikėtų pakeisti nefunkcinius reikalavimus ir suinteresuotų šalių interesus nefunkciniams reikalavimams į interesus funkciniams reikalavimams ir tuomet pateikti funkcinius reikalavimus. Tolesni modelio adaptavimo žingsniai nenumanomi. Tam reikia detalesnio tyrimo.

6. Išvados

Galime padaryti šias disertacinio tyrimo išvadas:

1. Paslaugų reikalavimų inžinerijai trūksta nuoseklių, detalių ir brandžių proceso modelių. Taip yra todėl, kad šiuo metu paslaugų inžinerijoje dar yra neišspręstų problemų ir uždavinių.
2. Paslaugų architektūros programų sistemos yra sudėtingesnės nei tradicines programų sistemos. Paprastai kuriant šias sistemas dalyvauja daug skirtingų suinteresuotų šalių, skirtingai įsivaizduojančių sistemos kokybines charakteristikas, tad anksčiau ar vėliau atsiranda nefunkcinių reikalavimų konfliktai.
3. Esti keletas paslaugų stiliaus architektūros programų sistemų kūrimo metodologijų: IBM RUP/SOMA, SOAF, SOUP, metodologija pagal Thomasą Erlą ar metodologija pagal Michaelį Papazoglou. Jos sukurtos, kad užtikrintų rezultatyvų paslaugų stiliaus architektūros programų sistemų kūrimą, tačiau nėra

skirtos paslaugų stiliaus reikalavimų inžinerijos procesui struktūrizuoti ir neteikia jokių pasiūlymų reikalavimų išgavimui ir analizei.

4. Pateiktas įmonių paslaugų stiliaus architektūros sistemų suinteresuotų šalių sąrašas. Jame išryškinti esminiai šių šalių skirtumai nuo tradicinės architektūros sistemų suinteresuotų šalių.
5. Pateiktas įmonių paslaugų stiliaus architektūros sistemų kokybinių charakteristikų (nefunkcinių reikalavimų) sąrašas. Jame išryškinti esminiai šių charakteristikų skirtumai nuo su tradicinės architektūros sistemų nefunkcinių reikalavimų.
6. Disertacinio tyrimo metu sukonstruotas ir pateiktas paslaugų stiliaus architektūros įmonių sistemų nefunkcinių reikalavimų išgavimo ir analizės spiralinis proceso modelis. Jis paremtas tradicinių ir paslaugų stiliaus programų sistemų reikalavimų rinkimo, analizės bei konfliktų sprendimo metodais, praktikomis ir įmonių architektūrų standartais bei karkasais.
7. Taikant proceso modelį naudotinos i* paremtos modeliavimo kalbos (GRL ir UCM) ir požiūriai, kurie yra rezultatyviai taikomi programų sistemų architektūrai analizuoti, specifikuoti ir modeliuoti. Taip pat rezultatyviai gali būti taikomi ir nefunkciniams reikalavimams identifikuoti ir analizuoti.
8. Proceso modelis taiko penkis nefunkcinių reikalavimų išgavimo ir analizės požiūrius: įmonės strategijos, įmonės verslo proceso, vartotojo, verslo procesų ir paslaugų. Atliktas eksperimentinis tyrimas – atvejo analizė, – kurios objektas – nefunkciniai reikalavimai paslaugų stiliaus architektūros įmonės draudimo sistemai, parodė, kad nefunkcinių reikalavimų analizė taikant požiūrius gali padėti efektyviai rasti konfliktuojančius nefunkcinius reikalavimus ir išspręsti konfliktus.
9. Proceso modelis yra paremtas iteratyviu reikalavimų derybų procesu ir leidžia pakartotines derybas. Reikalavimų derybų procesas paremtas spiraliniu proceso modeliu, kur kiekvienu reikalavimų derybų ciklu galima išspręsti vis sudėtingesnius reikalavimų konfliktus.
10. Pateikiamas proceso modelis pagrįstas pagrindiniu paslaugų stiliaus architektūros principu – kurti programų sistemas, kurios atitiktų verslo strategiją ir tikslus. Jis leidžia atskleisti, kodėl (modeliuojant verslo tikslus) vieni sistemos nefunkciniai reikalavimai yra svarbesni už kitus.
11. Pateikiamas proceso modelis gali būti taikomas kartu su paslaugų stiliaus architektūros programų sistemų kūrimo metodologijomis. Taip galima pagerinti reikalavimų išgavimo ir analizės galimybes ir pačios programų sistemos kokybę ir panaudojamumą.

Bibliografinių šaltinių, naudotų disertacijos santraukoje, sąrašas

Ahmad, S. (2008). Negotiation in the Requirements Elicitation and Analysis Process. 19th Australian Conference on Software Engineering. IEEE Computer Society Press, p. 683–689.

Amyot, D., Mussbacher, G. (2011). User Requirements Notation: The First Ten Years, The Next Ten Years. *Journal of Software,* Vol. 6, No. 5.

Arsanjani, A. (1999). Service Provider: A Meta-Domain Pattern and its Business Framework Implementation, In Online Proceedings of the Pattern Languages in Programming Conference.

Arsanjani, A. (2001). Enterprise Component: A compound pattern for building component architectures, IEEE Computer Society Press.

Bichler, M., Lin, K-J. (2006). Service-Oriented Computing. *Computer* (39-3), p. 99–101.

Boehm, B. (1988). A Spiral Model of Software Development and Enhancement. *Computer* (May 1988): 61–72.

Boehm, B. (2000). Spiral Development: Experience, Principles,and Refinements, Special Report CMU/SEI-2000-SR-008.

Choi, S. W., Her, J. S., Kim, S. D. (2007). Modeling QoS Attributes and Metrics for Evaluating Services in SOA Considering Consumers' Perspective as the First Class Requirement. APSCC, IEEE, p. 398–405.

Crnkovic G. D. (2010). Constructive Research and Info-Computational Knowledge Generation. Model-based reasoning in science and technology, Studies in Computational Intelligence, 2010, Volume 314/2010, p. 359–380.

DoDAF. Department of Defense Architecture Framework Version 2.02, 2010.

Erl, T. (2005). Service-Oriented Architecture: Concepts, Technology and Design. Prentice Hall PTR.

Erl, T. (2008). SOA Principles of Service Design. Prentice Hall PTR.

Erradi, A., Anand, S., Kulkarni, N. (2006). SOAF: An Architectural Framework for Service Definition and Realization. IEEE International Conference on Services Computing (SCC'06).

Errikson, I., McFadden, F. (1993). Quality Function Deployment: A Tool to Improve Software Quality. *Information and Software Technology* 35, 9.

E2AF. Extended Enterprise Architecture Framework Essentials Guide v1.5. (2006). Institute For Enterprise Architecture Developments.

Flores, F., Mora, M., Álvarez, F., O'Connor, R., Macias, J. (2008). Handbook of Research on Modern Systems Analysis and Design Technologies and Applications. In IGI Global (eds), p. 96–111.

Flores, F., Mora, M., Álvarez, F., Garza L., Durán, H. 2009. From Requirements Engineering to Service-oriented Requirements Engineering: An Analysis of Transition. Phoenix, AZ, USA, December 14, 2009, s.n., p. 1–13.

Flores, F., Mora, M., Álvarez, F., Garza L., Durán, H. (2010). Towards a Systematic Service-oriented Requirements Engineering Process (S-SoRE). *ENTERprise Information Systems Communications in Computer and Information Science*, Volume 109, p. 111–120.

Flyvbjerg, B. (2004). Five misunderstandings about case-study research. In C. Seale, G. Gobo, D. Silverman (eds.). Qualitative Research Practices. London and Thousand Oaks, CA: Sage, p. 420–434.

Hart, C. (1998). Doing a literature review: Releasing the social science research imagination. London, SAGE Publications.

Hofmeister, C., Nord, R., Soni, D. (1999). Applied Software Architecture. Addison-Wesley, Boston.

IBM RUP/SOMA. IBM Rational Unified Process for Service-Oriented Modelling and Architecture.

ISO/IEC/IEEE 42010:2011. Systems and software engineering – Architecture description.

Yu, E. (2009). Social Modeling and i*. In Conceptual Modeling: Foundations and Applications, Lecture Notes in Computer Science Volume 5600, p. 99–121.

Kroll, P., Kruchten, P., Booch, G. (2003) The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP: A Practitioner's Guide to the RUP. Addison-Wesley, ISBN-13: 978-0321166098 ISBN-10: 0321166094.

Kruchten, P. (1995). Architectural Blueprints — The "4+1" View Model of Software Architecture. IEEE Software 12 (6), p. 42–50.

Layzell, P. et. al. (2000). Service-based Software: The Future for Flexible Software, In Proceedings of Asia-Pacific Software Engineering Conference, 5-8 December, Singapore. IEEE Computer Society Press.

Laurence, S., Margolis, E. (2003). Concepts and conceptual analysis. Philosophy and Phenomenological Research, 67: 253–282.

Leite, P., Freeman, P. (1991). Requirements Validation Through Viewpoint Resolution. *IEEE Trans. Softw. Eng.* 17, 12.

Lukka, K. (2003). The constructive research approach. In: L. Ojala, O-P. Hilmola (eds.) Case study research in logistics. Publications of the Turku School of Economics and Business Administration, Series B 1: 2003, p. 83–101.

Machado, A., Silva, F. J. (2007). Toward a richer view of the scientific method. The role of conceptual analysis. American Psychologist, 62(7), p. 671–681.

Mingers, J., (2001). Combining IS Research Methods: Towards a Pluralist Methodology. Information Systems Research, Vol. 12, No. 3, p. 240–259.

Nuseibeh, B., Easterbrook, S. (2000). Requirements engineering: a roadmap. Proceedings of the Conference on The Future of Software Engineering (ICSE '00).

O'Brien, Liam., Bass, Len., Merson, P. F. (2005). Quality Attributes and Service-Oriented Architectures. Software Engineering Institute. Paper 449.

Papazoglou, M.P., 2006. Service-Oriented Design and Development Methodology. Int. J. of Web Engineering and Technology (IJWET).

Penrod, J., Hupcey, J. (2005). Enhancing methodological clarity: principle-based concept analysis. *Journal of Advanced Nursing* Vol. 50, No. 4, p. 403–409.

Ramollari, E., Dranidis, D., Simons, A.JH., 2007. A survey of service oriented development methodologies. The 2nd European Young Researchers Workshop on Service Oriented Computing.

Rational Software. (1998). Rational Unified Process: Best Practices for Software Development Teams.

RM-ODP. Reference Model of Open Distributed Processing.

Runeson, P., Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering, Vol. 14, Issue 2, 14: 131–164.

Russo, A., Nusbeibeh, B., Kramer, J. (1999). Restructuring Requirements Specifications. IEEE Proceedings.

Sambeth, M. (2006) Enterprise SOA. Mastering Future Business. Presentation slides, SAP AG.

SOA GRM, SOA Governance Technical Standard: SOA Governance Reference Model.

SOA-RM. (2006). Reference Model for Service-Oriented Architecture 1.0. s.l.: OASIS.

Soni, D., Nord, R., Hofmeister, C., 1995. Software architecture in industrial applications. In: Proceedings of 17th International Conference on Software Engineering (ICSE-17). ACM Press, p. 196–207.
Sommerville, I., Sawyer, P. (1997). Viewpoints: principles, problems and a practical approach to requirements engineering. Annals of Software Engineering, Vol. 3, p. 101–130.
SOUP. Service-Oriented Unified Process.
The Open Group SOA Reference Architecture (SOA-RA), 2011. Technical Standard.
Trienekens, J., Bouman, J.J., van der Zwan, M. (2004). Specification of Service Level Agreements: Problems, Principles and Practices, Software Quality Journal, 12(1), p. 43–57.
TOGAF 9.1 (2011). An Open Group standard.
Van Eck, P., Wieringa, R. (2003). Requirements Engineering for Service-Oriented Computing: a Position Paper, Proceedings of First International E-Services Workshop, ICEC 03, Pittsburgh, USA, p. 23–28.
Zachman International Enterprise Architecture v3.0 (2011).

Autorės mokslinių publikacijų disertacijos tema sąrašas

Straipsniai recenzuojamuose moksliniuose leidiniuose

Svanidzaitė, S. (2014c). A Methodology for Capturing and Managing Non-Functional Requirements for Enterprise Service-Oriented Systems. Baltic J. Modern Computing Vol. 2, No.3, p. 17–131.
Svanidzaitė, S (2014b). An Approach to SOA Development Methodology: SOUP Comparison with RUP and XP. Computational Science and Techniques Vol. 2, No.1 (2014), p. 238–252, ISSN: 2029-9966.

Straipsniai kituose leidiniuose

Svanidzaite, S. (2014a). Towards Service-oriented Requirement Engineering. Proceedings of IVUS 2014, XIX Interuniversity Conference Information Society and University Studies 2014, p.15–20.
Svanidzaitė, S (2012). A comparison of SOA methodologies Analysis & Design Phases. Baltic DB & IS 2012, Tenth International Baltic Conference on Databases and Information Systems, July 8–11, 2012, Vilnius, Lithuania.

Trumpos žinios apie autorę

Sandra Svanidzaitė gimė 1985 m. lapkričio 13 d. Marijampolėje. 2004 m. baigė Marijampolės „Šaltinio“ vidurinę mokyklą. 2008 m. Vilniaus universiteto Matematikos ir informatikos fakultete įgijo informatikos bakalauro laipsnį. 2010 m. Vilniaus universiteto Matematikos ir informatikos fakultete įgijo informatikos magistro laipsnį. 2010–2014 m. doktorantė Vilniaus universiteto Matematikos ir informatikos institute.

SPIRAL PROCESS MODEL FOR CAPTURE AND ANALYSIS NON-FUNCTIONAL REQUIREMENTS OF SERVICE-ORIENTED ENTERPRISE SYSTEMS

Research Context and Challenges

Service-Oriented Architecture (SOA) is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains (SOA-RM, 2006). In general, people and organizations create capabilities to solve the problems they face in the course of their business. It is natural to think of one person's needs being met by capabilities offered by someone else, or, in the world of distributed computing, one computer agent's requirements being met by a computer agent belonging to a different owner. The perceived value of SOA is that it provides a powerful framework for matching needs and capabilities and for combining capabilities to address those needs. Visibility, interaction, and effect are key concepts for describing the SOA paradigm. Visibility refers to the capacity for those with needs and those with capabilities to be able to see each other. This is typically done by providing descriptions for such aspects as functions and technical requirements, related constraints and policies, and mechanisms for access or response. The descriptions need to be in a form in which their syntax and semantics are widely accessible and understandable. Interaction is the activity of using a capability. Interaction proceeds through a series of information exchanges and invoked actions. At the interaction stage, the description of real world effects establishes the expectations of those using the capability. Needless to say, it is not possible to describe every effect from using a capability. A cornerstone of SOA is that capabilities can be used without the need to know all the details. The service is a central concept of SOA (Erl, 2005; Erl, 2008). It combines the following related ideas: the capability to perform work for another, the specification of the work offered for another, the offer to perform work for another. The concepts of visibility, interaction, and effect apply directly to services in the same manner as these were described for the general SOA paradigm.

While both needs and capabilities exist independently of SOA, in SOA, services are the mechanism by which needs and capabilities are brought together. SOA is a means of organizing solutions that promotes reuse, growth and interoperability. It is not itself a solution to domain problems but rather an organizing and delivery paradigm that enables one to get more value from use both of capabilities which are locally "owned" and those under the control of others. It also enables one to express solutions in a way that makes it easier to modify or evolve the identified solution or to try alternate solutions. In addition to this, SOA does not provide any domain elements of a solution that do not exist without SOA.

Problem Statement

The subject of the thesis research is capture and analysis of non-functional requirements in ESOA systems starting from the highest abstraction level – enterprise strategy – where business goals are elicited, deriving system non-functional requirements from business

goals and refining them until concrete non-functional requirements are produced for each service in the ESOA system.
Software system requirements are normally divided into functional and non-functional requirements. Functional requirements focus on to what extent the software system actually does what it is expected to do. Non-functional requirements, on the other hand, are said to be the constraints on the system functions and are less obvious and harder to identify. As a result, non-functional requirements receive less attention and thus become more critical. The choice to use an ESOA approach depends on several factors including the architecture's ultimate ability to meet functional and non-functional requirements. Usually, architecture needs to satisfy many non-functional requirements in order to achieve enterprise business goals.
The research aims to define a process model for capture and analysis of non-functional requirements as each stakeholder group involved in the ESOA initiative usually have different expectations regarding system quality characteristics and there is a necessity to be able to negotiate and trade-off these differences.

Motivation

Many large software projects are ill-defined as a result of the high level of complexity. It becomes difficult not only to fully specify system requirements but even to understand all aspects of the system.
According to (SOUP; Svanidzaite, 2014b) SOA/ESOA projects potentially suffer from one or more of the following problems:

- They are significantly more complex than typical software projects, because they require a larger, cross-functional team along with correspondingly more complex inter-team communication and logistics;
- Usually it is hard to define the scope and boundaries of a project. As a result, the vision for the final result is often not clear at the project inception;
- SOA can have a very positive impact on an enterprise, but, on the other hand, the development and replacement of legacy systems can be very expensive;
- SOA/ESOA project has a higher risk of failure than other traditional software development projects.

Despite these problems, SOA/ESOA approaches are gaining popularity and are used for more and more complex systems. Having this in mind, SOA/ESOA projects require much more sophisticated requirement gathering and analysis techniques.
While for previous paradigms we have well-researched and stable requirements engineering (RE) processes and techniques, in service-oriented requirement engineering (SORE) such processes and techniques still are under research (Flores et al, 2009; Flores et al, 2010). SORE like traditional requirement engineering, concerns with the specification and analysis of system requirements and constraints but its focus is on the identification of services and workflows used to modelling applications and on their reuse. Several service-oriented system development methodologies and approaches were proposed but they are not aimed at structuring SORE process, lack details and procedures for requirements gathering and analysis, as a result, further research is required.

Research by (Bano et al, 2010) suggests that SORE faces with four main categories of issues and challenges: service specification, service discovery, service knowledge management, and service composition issues (2.1 Service-Oriented Requirement Engineering).

Service specification issues are the ones of great importance for ESOA because the system is only as good as its requirements are. As a result, our research focuses on the resolution of the following concern: capturing and analysing non-functional requirements of ESOA systems, finding conflicting requirements and proposing an approach how to resolve them.

It is suggested by (Leite & Freeman, 1991; Sommerville & Sawyer, 1997; Russo et al, 1999; Nuseibeh & Easterbrook, 2000) that system requirements should be elicited and defined from different viewpoints. For any given viewpoint of the system many aspects will be hidden and only ones actual to the viewpoint will be depicted in details. As a consequence, multiple viewpoints need to be considered in order to fully understand and specify the system-of-interest. Viewpoints can be used to improve system requirements gathering, analysis and conflict resolution process.

Furthermore, i* (pronounced "i star") is a framework (Yu, 2009) suitable for an early phase system modelling in order to understand the problem domain. i*-based modelling language can be used to model viewpoints when specifying system requirements as it allows to model both as-is and to-be business models. It covers both actor-oriented and goal-oriented modelling. The i* models answer the question who (actor) and why (goal), not what (system function). The i* framework is a part of a User Requirements Notation (URN) international standard. The URN standard combines two sub-languages (Amyot & Mussbacher, 2011): Goal-oriented Requirement Language (GRL) and Use Case Maps (UCM) notation. URN is the first international standard that addresses business goals and scenarios and links between them in a graphical way.

As SORE has emerged recently, there are no works that deal with Service Specification issues employing viewpoints and User Requirements Notation (URN) standard languages directly. Having this in mind, further research is required.

Aims and Objectives of the Research

The research aims to develop a process model that allows a system analyst to capture and analyse non-functional requirements for enterprise service-oriented systems that are designed incorporating traditional and service-oriented requirement gathering process models, conflicts management approaches and techniques, EA standards and frameworks. In order to achieve this aim, the following research objectives have been stated:

1. To evaluate the state of affairs in SORE and all other interrelated enterprise and service-oriented architecture domain areas including service-oriented systems development methodologies, enterprise architecture standards and frameworks that could be used for non-functional requirements definition conflicts resolution in ESOA systems;
2. To propose a set of stakeholders for ESOA systems and highlight the main differences between stakeholders for traditional systems and these;

3. To define a set of quality attributes (non-functional requirements) for ESOA by drawing the main attention to their differences in respect of traditional systems non-functional requirements;
4. To develop a process model for non-functional requirements capturing and analysis that includes a process for conflict resolution between different stakeholder groups.

Research Questions and Hypotheses

Main questions that need to be answered in this research are:

- How mature is Service-Oriented Requirement Engineering (SORE) currently? What are the main issues and challenges of it? What SORE process models are already created? Are they mature enough to ensure successful SOA/ESOA systems development?
- What service-oriented systems development approaches and methodologies are created? Are they mature enough to ensure successful SOA/ESOA systems development? Can analysis and design creation phases of these methodologies and approaches be used to solve SORE issues and challenges successfully?
- How do SOA/ESOA systems non-functional requirements differ from traditional systems non-functional requirements? Do SORE or service-oriented systems development methodologies and approaches provide solutions for non-functional requirements capturing and analysis?
- How can traditional requirement engineering processes and their models be used to solve SORE issues and challenges? To what extent can traditional requirements conflicts negotiation approaches be used to solve enterprise systems non-functional requirements conflicts? How can enterprise service-oriented systems non-functional requirements conflicts be solved?
- Can viewpoints that are usually used when designing enterprise architectures be used to structure and analyse enterprise service-oriented systems non-functional requirements? Can i*-based modelling languages be used to model and negotiate SOA/ESOA non-functional requirements?

To answer these questions, the following hypotheses have been stated:

- **H1.** There exist service-oriented systems development methodologies such as IBM RUP/SOMA, SOAF, SOUP, service-oriented analysis and design methodology by Thomas Erl, service-oriented design and development methodology by Papazoglou that can be used to create service-oriented requirement engineering process models;
- **H2.** Traditional requirement gathering, conflicts management approaches and techniques can at least to some extent be used to capture, analyse and negotiate enterprise service-oriented systems non-functional requirements;
- **H3.** i*-based modelling languages and viewpoints that are widely used in Enterprise Architecture (EA) standards and frameworks can be used to solve conflicting non-functional requirements in SOA/ESOA systems;
- **H4**. A process model for enterprise service-oriented systems non-functional requirements capturing and analysis can be developed incorporating traditional

requirements gathering, conflicts management approaches and techniques, i*-based modelling languages and viewpoints.

Research Design and Research Methods

The research design of present thesis is of theoretical and empirical nature, as it is usual in the field of Informatics. Service-oriented requirement engineering is a relatively young research and development area. The research in this area is still in its infancy. It means that a relatively large amount of library research is required in order to define the exact structure of a problem, and to gain a better understanding of the environment within which the problem arises. In this context, the best way of solving the problem of theoretical and empirical nature is constructive research (Mingers, 2001). Furthermore, any dissertation research is a small-scale research from both financial and time points of view. It means that in such research it is too expensive and practically impossible to ensure high statistical reliability and high level statistical significance. Thus, despite its possible biases, the case study methodology is the only practically acceptable methodology to validate the research results.

Taking into account all that was discussed above, the research design provides three distinctive research phases: *conceptual analysis* (Laurence & Margolis, 2003) of related work, *constructive research* that aims to develop a process model for ESOA non-functional requirements capture and analysis and experimental investigation – *a case study* that validates the designed process model.

Conceptual analysis is the analysis of concepts, terms, variables, constructs, definitions, assertions, hypotheses, and theories. It involves examining these for clarity and coherence, critically scrutinizing their logical relations, and identifying assumptions and implications (Machado & Silva, 2007). The goal of conceptual analysis is to increase the conceptual clarity of the research subject. The primary utility of conceptual analysis is to determine the existing state of the research field so that further work may be strategically and appropriately planned (Penrod & Hupcey, 2005). The conceptual analysis of related works has been carried out to generate important theoretical constructs and to provide a theoretical basis for further research as well as to avoid performing research that has already been done by others (Hart, 1998). The main fields on which conceptual analysis has been performed includes service-oriented architecture (SOA), enterprise service-oriented architecture (ESOA), service-oriented requirement engineering (SORE), service-oriented systems development methodologies, enterprise architecture frameworks and standards.

Generally, conceptual analysis allowed us to answer the questions of how mature SORE is, what its main issues and challenges are, what the process models created for SORE process structuration are, whether they are mature enough, whether service-oriented systems development methodologies together with enterprise architecture frameworks and standards can be used to solve our selected service specification issue in SORE.

The constructive research approach is a research procedure for producing innovative constructions intended to solve the problems encountered in the real world and to make some contribution to the theory of the discipline in which it is applied (Lukka, 2003; Crnkovic, 2010). The central notion of this approach, the novel construction, is an abstract notion with a great variety of potential realizations. Models, designs, methods,

algorithms, and most other artefacts are considered as constructions. It means that they are invented and developed, not discovered. The constructive research approach is based on the belief that by an in-depth analysis of what works (or does not work) in practice one can make a significant contribution to theory. In the present thesis this approach is used to design a process model for ESOA non-functional requirements capturing and management. As a result of an in-depth analysis of the problem, it has been discovered that process model can be based on service-oriented architecture layers, EA standards and EA frameworks and include five viewpoints: Enterprise Strategy, Enterprise Business Processes, Consumer, Business Process and Service Viewpoints.
A constructive research methodology is also used to test working hypotheses that have been provisionally accepted in the present thesis. One of the advantages of this methodology is that it allows not only to test and investigate the properties of the innovative construction but also to study its development process. On the other hand, constructive research can be viewed as a kind of case study methodology. However, according to the conventional view, case studies should be used for falsification of the hypothesis only. Case study itself cannot prove any hypothesis and should be linked to some hypothetic-deductive model of explanation. However, the correspondence of case study to real-world situations and its multiple wealth of details state that this view is only partly correct (Flyvbjerg, 2004). Taking into account this argument and the fact that the research for the dissertation is a small-scale research from both financial and time points of view, the case study methodology has been approved as the main hypothesis testing methodology. Mainly, case study is an empirical research method that aims at investigating some phenomena in his context (Runeson & Höst, 2009). In the present thesis the aim is to test the applicability of the process model for ESOA non-functional requirement capture and analysis by choosing a simplified real life example in which we test the possibilities of capturing and analysing non-functional requirements for enterprise service-oriented insurance system.

Summary of Research Results

The results of the thesis research can be summarized as follows:

- Hypothesis **H1** that service-oriented systems development methodologies can be used to create service-oriented requirement engineering process models has been rejected. During the thesis research we have found out those service-oriented systems' development methodologies lack details about the capture and analysis of requirements. As a result, SORE process models can be used in conjunction with service-oriented systems development methodologies;
- Hypothesis **H2** that traditional requirement gathering, conflicts management approaches and techniques can be at least to some extent be used to capture, manage and negotiate enterprise service-oriented systems' non-functional requirements has been approved. Our proposed ESOA non-functional requirements capture and analysis process model is based on the spiral requirement negotiation model from traditional requirement engineering;
- Hypothesis **H3** has been validated and approved with case study that i*-based modelling languages and viewpoints that are widely used in Enterprise

Architecture (EA) standards and frameworks can be used to solve conflicting non-functional requirements in SOA/ESOA systems;

- Hypothesis **H4** has been constructively proven by developing and proposing a spiral process model for capture and analysis non-functional requirements of service-oriented enterprise systems that is designed incorporating traditional requirement gathering and conflicts management approaches and techniques, i*-based modelling languages and viewpoints.
- Viewpoints: Enterprise Strategy Viewpoint, Enterprise Business Processes Viewpoint, Consumer Viewpoint, Business Process Viewpoint, Service Viewpoint have been developed in the thesis research that can be applied for ESOA as well as SOA systems requirements capture and analysis.

Contributions of the Dissertation

The present thesis is one the first research works that aims to investigate non-functional requirements capturing and management techniques in the context of enterprise service-oriented architecture (ESOA) systems. Although, there has been several attempts to propose a service-oriented requirement engineering process models (Flores, et al, 2010; Flores, et al, 2009; Flores, et al, 2008) and service-oriented systems development methodologies (Papazoglou, 2006; IBM RUP/SOMA; Erl, 2005; Erl, 2008; Erradi, et al, 2006; SOUP) none of them are sufficient and mature enough to ensure ESOA systems development including sophisticated requirements capture, analysis and negotiation processes. Furthermore, it is also the first work that raises the question whether enterprise service-oriented architecture systems non-functional requirements can be captured using viewpoints and modelled using i*-based modelling languages and finally confirms it with case study.

The practical significance of the thesis is as follows:

- Spiral process model for ESOA non-functional requirements capture and analysis that have been developed in the thesis research can be applied developing ESOA systems. In addition to this, it can be successfully combined with service-oriented systems development approaches methodologies and provide a coherent and comprehensive solution for service-oriented enterprise systems development from planning, analysis and design to deployment and change management.
- Viewpoints that have been developed in the thesis research can be applied for ESOA as well as SOA systems. Furthermore, after some customization, they can also be used to model functional requirements.

Approbation

The main results of the thesis were presented and approved at the following conferences:

- 15th Conference of Lithuanian Computer Society "Computer Days – 2011", September 22–24, 2011, Klaipėda, Lithuania;
- 10th International Baltic Conference on Databases and Information Systems (Baltic DB&IS 2012), July 8-11, 2012, Vilnius, Lithuania

- Information Society and University Studies - IVUS 2014, April 24 2014, Kaunas, Lithuania.
- 4th Junior Scientists Conference of Physical and Technology Sciences Interdisciplinary Researches, February 11 2014, Vilnius Lithuania.

Outline of the Dissertation

The text of the thesis consists of introduction, 5 main chapters, conclusions, list of references and list of publications. Main chapters are provided with summary and (except Chapter 1) with conclusions.

Chapter 1 presents preliminaries on Service-Oriented Architecture (SOA) and one of its sub-types Enterprise Service-Oriented Architecture (ESOA) by highlighting main differences between them. Furthermore, chapter describes User Requirement Standard Notation Languages: Goal Requirement Language (GRL) and Use Case Map (UCM) that are used in research to model ESOA system non-functional characteristics.

Chapter 2 describes the results of critical analysis of related works. It presents the state-of-the-art of Service-oriented Requirement Engineering and all other interrelated enterprise and service-oriented architecture areas (including service-oriented system development methodologies and enterprise architecture frameworks) that could be used for non-functional requirements conflicts resolution in ESOA systems. Chapter also analyses the problematics of capturing non-functional requirements for ESOA systems using viewpoints.

Chapter 3 develops and discusses main theoretical results of doctoral research. Chapter analyses and outlines the possible stakeholders of ESOA systems and discusses the non-functional requirements (quality characteristics) that will be treated as concerns in our proposed ESOA viewpoints. Furthermore, chapter describes a spiral process model for ESOA non-functional requirements capturing and analysis. Chapter is summarized with discussion of process model viewpoints mapping to architecture domains and process models' applicability to use it in conjunction with service-oriented systems development methodologies.

Chapter 4 presents evaluation results. A case study was performed for this aim. Chapter starts with describing how ESOA viewpoints are modelled in a case study. The following three sections in chapter apply our proposed methodology on each of ESOA viewpoints.

Chapter 5 discusses some open questions and limitations.

Results and Conclusions present the main results and conclusions of the dissertation.

Results and Conclusions

The results of the thesis research can be summarized as follows:

1. Service-oriented requirement engineering (SORE) lacks coherent, comprehensive and mature requirement engineering process models. There exist issues and challenges in SORE that have not been solved yet.
2. ESOA systems are of high complexity, usually have many different stakeholder groups with different expectations of systems non-functional characteristics and conflicting expectations inevitably rise. In addition to this, ESOA systems non-

functional requirements differ from traditional systems non-functional requirements and there are no mature service-oriented requirement engineering process models targeted at them.

3. Several service-oriented system development methodologies such as IBM RUP/SOMA, SOAF, SOUP, the methodology by Tomas Erl, and methodology by Michael Papazoglou have been proposed to ensure successful service-oriented systems development by providing process guidance and proven best practices from already accomplished SOA projects. Although these methodologies help to structure service-oriented systems development processes, they are not aimed at defining SORE process and do not provide any approach to requirement capturing and analysis.
4. A set of typical stakeholder groups for ESOA systems has been proposed with the main differences between stakeholders for traditional systems and ESOA systems highlighted.
5. A set of quality attributes (non-functional requirements) for ESOA systems has been proposed by drawing the main attention to their differences with respect to traditional systems non-functional requirements.
6. Process model for capturing and analysis non-functional requirements of service-oriented enterprise systems has been proposed; it is designed incorporating traditional and service-oriented requirement gathering process models, conflicts management approaches and techniques, EA standards and frameworks.
7. Process model recommends using i*-based modelling languages (GRL and UCM) and viewpoints that are widely used in Enterprise Architecture (EA) standards and frameworks and have not been previously thoroughly researched for their applicability to solve issues and challenges of service specification for service-oriented enterprise systems.
8. Process model includes five viewpoints – Enterprise Strategy viewpoint, Enterprise Business Processes viewpoint, Consumer viewpoint, Business Process viewpoint, and Service viewpoint. A case study performed on an insurance domain revealed that non-functional requirements analysis using viewpoints can help to find conflicting requirements and resolve requirements conflicts, because service-oriented enterprise systems are complex ones and usually stakeholders have conflicting and overlapping requirements.
9. Process model is designed to benefit from the iterative requirement negotiation process and allows renegotiation. Requirement negotiation process is based on a spiral model to accommodate the dynamic requirements engineering. Each round of the cycle resolves more conflicted requirements and achieves better resolution. The model considers the win conditions of all stakeholders that participate in ESOA project as it identifies and evaluates alternative approaches for satisfying the win conditions. It also helps to identify and resolve risks that stem from the selected approach by performing elaboration, judgment and the trade-off of selected solution.
10. Process model is based on the main aim of service-orientation – to develop systems that support enterprise business strategy, objectives and goals and, as a result, is primarily concerned with exposing "why" (by modelling business goals) certain non-functional requirements are more important than the others.

11. Process model can be used in conjunction with service-oriented systems development methodologies to improve requirement capturing, analysis capabilities and, at the same time, increase system quality and usability.

About the Author

Sandra Svanidzaitė was born in Lithuania in Marijampolė on 13th of November 1985. In 2004, she finished Marijampolės "Šaltinis" secondary school. She graduated from Vilnius University Faculty of Mathematics and Informatics in 2008 acquiring Bachelor's Degree in Informatics. She gained hers Master's Degree in Informatics at Vilnius University Faculty of Mathematics and Informatics in 2010. From 2010 to 2014 she has been a doctorate at Vilnius University Institute of Mathematics and Informatics.

Sandra Svanidzaitė

SPIRALINIS PROCESO MODELIS PASLAUGŲ STILIAUS ARCHITEKTŪROS ĮMONIŲ SISTEMŲ NEFUNKCINIAMS REIKALAVIMAMS IŠGAUTI IR ANALIZUOTI

Daktaro disertacijos santrauka

Fiziniai mokslai, informatika (09 P)

Redaktorė Jorūnė Rimeisytė

Sandra Svanidzaitė

SPIRAL PROCESS MODEL FOR CAPTURE AND ANALYSIS NON-FUNCTIONAL REQUIREMENTS OF SERVICE-ORIENTED ENTERPRISE SYSTEMS

Summary of Doctoral Dissertation

Physical Sciences, Informatics (09 P)

Editor Zuzana Šiušaitė