KLAIPĖDOS UNIVERSITETAS
GAMTOS IR MATEMATIKOS MOKSLŲ FAKULTETAS
INFORMATIKOS KATEDRA

Semion Larin

Informatikos specialybės studentas

Informacijos apdorojimo taisykles inicijuojančių įvykių sistemos tyrimas

Magistro tezė

Mokslinis vadovas: prof. dr. Olegas Vasilecas
Konsultantas: Aidas Šmaižys

Klaipėda, 2007

Anotacija

Šiuolaikinėse informacinėse sistemose, kurios siekia maksimaliai patenkinti verslo poreikius, yra plačiai taikomos verslo taisyklės. Verslo taisyklių problematiką nagrinėja daug mokslinių grupių ir yra nemažai publikacijų, analizuojančių verslo taisyklių modeliavimą ir jų įgyvendinimą. Realizuojamos verslo taisyklės yra transformuojamos į informacinės sistemos, ir toliau į programų sistemos, taisykles. Tačiau ir taisykles inicijuojantys įvykiai taip pat turi būti transformuojami nuo verslo iki programų sistemos.

Šiame darbe atlikta susijusių publikacijų analizė rodo, kad įvykių, kurie aktyvuoja taisykles, analizei skiriamas nepakankamas dėmesys. Tinkama taisyklių ir įvykių sistemos analizė bei modeliavimas įgalins patobulinti informacinės sistemos kūrimo procesą. Taisyklių, įvykių ir procesų sistemų modelius apjungimas į vieningą karkasą sudarys galimybes kurti sistemas kurios geriau tenkins vartotojų poreikius bei reikalaus mažesnių išlaidų sistemos eksploatacijos ir palaikymo procese.

Reikšminiai žodžiai: įvykiai, įvykių sistema, verslo taisyklės.

Annotation

Business rules are widely applied in modern information systems, aimed to maximum satisfaction of business requests. Sphere of business rules problems involves a lot of scientific groups and there are a lot of publications with business rules modelling and implementation. While on realization, business rules are transformed to information systems, than to software systems, rules. But rules initiating events should be transformed from business to software systems as well.

The main idea of this work was that there is not enough attention dedicated to rules activating events analyzing, which was based on completed analyze of related publications. Suitable analyzing of rules and events as well as modelling will reinforce system formation process. Combining rules, events and process systems model into one framework allows to develop information systems that will be more adjusted for user's need and will give a chance to reduce expenses during system exploitation and maintenance process.

Keywords: Events, Event System, Business rules.

TURINYS

PAVEIKSLŲ SĄRAŠAS

LENTELIŲ SĄRAŠAS

ĮVADAS

Temos aktualumas

Šiuo metu informacinių sistemų kūrimui, kurios galėtų maksimaliai patenkinti verslo poreikius, yra naudojamos verslo taisyklės. Tą temą nagrinėja labai daug mokslinių grupių ir yra parašyta daug straipsnių ir publikacijų, aprašančių verslo taisykles, jų formalizavimą ir pritaikymą. Tinkamas įvykių sistemų realizavimas gali optimizuoti informacinės sistemos darbą.

Šiuolaikinėse informacinėse sistemose, kurios siekia maksimaliai patenkinti verslo poreikius, yra plačiai taikomos verslo taisyklės. Realizuojamos verslo taisyklės yra transformuojamos į informacinės sistemos, ir toliau į programų sistemos, taisykles. Tačiau ir taisykles inicijuojantys įvykiai taip pat turi būti transformuojami nuo verslo iki programų sistemos.

Šiame darbe atlikta susijusių publikacijų analizė rodo, kad įvykių, kurie aktyvuoja taisykles, analizei skiriamas nepakankamas dėmesys. Tinkama taisyklių ir įvykių sistemos analizė bei modeliavimas gali patobulinti informacinės sistemos kūrimo procesą ir tikėtina, kad tokiu būdu sukurtos sistemos geriau tenkins vartotojų poreikius bei reikalaus mažesnių išlaidų sistemos eksploatacijos procese.

Tyrimo objektas

Tyrimo objektas yra verslo taisyklėmis grindžiama informacinių sistemų kūrimo metodika, akcentuojant verslo ir informacinės sistemų įvykius.

Tyrimo tikslas

Išplėtoti informacinių sistemų kūrimo metodus įvertinant verslo ir informacinės sistemų įvykius ir jų realizaciją programų sistemoje.

Uždaviniai:

1. Atlikti susijusių su tyrimo objektu mokslinių darbų ir sąvokų analizę.
2. Išanalizuoti įvykių ir įvykių sistemų įtaką verslo, informacinėms ir programų sistemoms.
3. Ištirti ir pasiūlyti įvykių sistemos modeliavimo būdą ir įvykių modelio panaudojimu pagristą PS kūrimo metodą.
4. Sukurti IS prototipą įvykių transformavimui tarp VS, IS ir PS susiejimui ir kitimui tirti.
5. Patikrinti pasiūlytą metoda kuriant IS prototipą.

Temos naujumas

Tema yra nauja. Verslo taisyklių realizavimui ir vaizdavimui yra skirta pakankamai daug dėmesio straipsniuose ir kitoje literatūroje, tačiau įvykių sistemai yra skiriama mažai dėmesio.

Tyrimo metodai

Darbe bus naudojama literatūros analizė ir eksperimentinė bei lyginamoji analizė

1. Darbo motyvacija

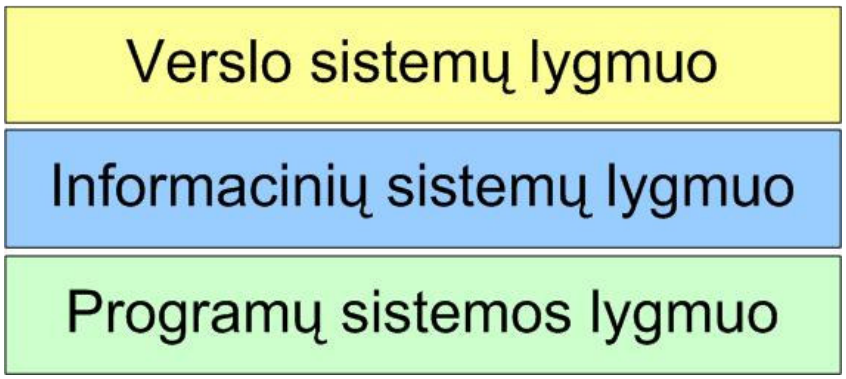
Verslo sistemų funkcionalumas remiasi atskirų verslo sričių taisyklėmis. Dinaminė prigimtis sąlygoja verslo aplinką dažnai keistis priklausomai nuo vidinių ar išorinių veiksnių, tokių kaip įstatymų pokytis, nauji rinkos poreikiai ir kiti [37]. Verslo taisyklių problematiką nagrinėja daug mokslinių ir praktinių grupių (tokių kaip: „The Business Rules Group" [33], "Business Rules Community" [4], "OMG" [27] ir t. t.) ir yra daug susijusių publikacijų, analizuojančių verslo taisyklių modeliavimą ir įgyvendinimą. Tam, kad šiuolaikinės informacinės sistemos maksimaliai tenkintų besikeičiančio verslo poreikius, privaloma atsižvelgti į verslo taisykles. Realizuojamos verslo taisyklės yra transformuojamos į informacinės sistemos, ir toliau į programų sistemos taisykles. Taisyklės negali savaime inicijuotis ir būti vykdomos, jas inicijuoja sistemoje vykstantis įvykiai. Tačiau įvykių, kurie aktyvuoja taisykles, analizei skiriamas nepakankamas dėmesys. Įvykiai, kaip ir taisyklės, taip pat turi būti transformuojami nuo verslo iki programų sistemos. Tinkamai išanalizavus įvykių sistemą ir sukūrus jos modelį, galima patobulinti informacinės sistemos kūrimo procesą. Tokios sistemos galės geriau tenkinti vartotojų poreikius. Be to, įvykių sistemos modelis padės geriau stebėti procesų vykdymo eiliškumą ir vietą verslo taisyklių sistemoje, o tai įgalins, pasikeitus taisyklėms ar jas inicijuojantiems įvykiams, be kardinalaus sistemos pertvarkymo efektyviai pakeisti susijusius procesus.

2. Susijusių darbų analizė

Šiame skyriuje, pasitelkiant literatūros analizę, yra nagrinėjamos esminės su darbų susijusios sąvokos ir įvykių bei įvykių sistemų įtaką verslo, informacinėms ir programų sistemoms.

2.1. Verslo, informacinių ir programų sistemų lygmenys

Istoriškai susiklostė, kad skirtingiems požiūriams į sistemas klasifikuoti buvo pasirinkti vadinamieji abstrakcijų lygmenys (angl. *layers*). Požiūrių skirstymas lygmenimis yra gana subjektyvus ir neapibrėžtas dalykas. Sunku griežtai apibrėžti ne tik ribas tarp skirtinų požiūrių lygmenų, bet ir pačius lygmenys. Šiame darbe bus nagrinėjami trys pagrindinai lygmenys, į kuriuos dažniausiai skirstomi skirtingi požiūriai: verslo sistemų lygmuo, informacinių sistemų lygmuo, programų sistemų lygmuo (1 pav.), kaip tai pasiūlyta kai kurių autorių [1, 5, 35, 36].



1 pav. ***Trys sistemų lygmenys***

Kaip jau buvo minėta, šie trys lygmenys nėra vienareikšmis ir galutinis sistemų sandaros variantas. Žemiau programų sistemų lygmens gali būti kompiuterinių sistemų lygmuo, aukščiau verslo įmonės (angl. *enterprise*) lygmuo. Informacinių sistemų lygmens tikslas yra sąvokinių informacinės sistemos specifikacijų, skirtų jos efektyviam vystymui, pavaizdavimas. Iš to galima padaryti išvadą, kad informacinių sistemų lygmuo turi būti nepriklausomas technologiniu ir organizaciniu aspektu [18]. Pagrindinis dėmesys šiame darbe yra skiriamas informacinių ir programų sistemų lygmenims.

2.2. Verslo taisyklė

Kiekvieno programinio produkto uždavinys yra paremti tam tikrą verslo sistemos veiklą (angl. *enterprise work item*). Verslo sistema aprašoma kaip tam tikrų apibrėžimų (nukreipiančių, varžančių ir t. t.) rinkinys. Tie deklaratyvūs apibrėžimai yra ir formalūs, o tai leidžia vadinti juos verslo taisyklėmis arba tiesiog taisyklėmis [12].

Verslo taisyklių tyrimo grupės „The Business Rules Group“ darbo ataskaitoje [15], taip pat „OMG“ grupės sukurtoje specifikacijoje [28] verslo taisyklės yra apibrėžiamos taip: verslo taisyklės – tai teiginiai, apibrėžiantys ar ribojantys tam tikrus verslo aspektus; jų paskirtis - daryti įtaką verslo struktūrai arba kontroliuoti verslą. Verslo požiūriu (Zachmano karkaso (angl. *framework*) antras lygmuo) taisyklės sieja visus ribojimus taikomus žmonių ir verslo elgesiui, nuo draudimo rūkyti iki užsakymo pildymo nurodymų [15].

Verslo taisyklės atvaizduoja nuolat besikeičiančią verslo tvarką ir gali kilti iš paties verslo arba būti perduotos iš verslo aplinkos. Paprasčiausiame lygyje, verslo taisyklė yra panaši į teiginį su JEI/TADA, kuris lygina kintamąjį su gautąją reikšmę ir pateikia komandą, kai jos (kintamasis ir vertė) yra sulyginamos. Pavyzdžiui:

JEI klientas yra pensinio amžiaus, TADA suteikti 5% nuolaidą.

Šiuo atveju kintamasis yra pensinis žmogaus amžius ir nuolaidos dydis.

JEI asmuo pravažiuoja daugiau nei 150 mylių per savaitę, TADA padidinti draudimo išmoką 25$.

Šioje taisyklėje kintamieji bus pravažiuotų mylių ir pinigų, pridėtų prie išmokos, kiekis [17].

2.3. ECA taisyklė

ECA (angl. *event/condition/action*) taisyklės buvo realizuotos aktyviose duomenų bazių sistemose. Reliacinėse duomenų bazių sistemose ECA taisyklės yra realizuojamos naudojant trigerius, objektinėse – naudojant objektų metodus.

Aktyvios duomenų bazių sistemos gali reaguoti į įvykius pagal realizuotas ECA taisykles. Reakcija į įvykį yra apibrėžiama kaip taisyklė. Taisyklės susideda iš trijų dalių: įvykis (angl. *event*), sąlyga (angl. *condition*), veiksmas (angl. *action*). Įvykis – tai atsitikimas tam tikru laiko momentu. Sąlygos – tai predikatai, susiję su duomenų baze. Jos apibrėžia, kokiom sąlygoms esant įvykis yra svarbus ir turi būti apdorojamas. Tačiau sąlygos yra nebūtina taisyklės dalis. Veiksmas yra tai, kas turi būti padaryta, jeigu įvykis ir sąlyga davė teigiamą (angl. *true*) rezultatą [16].

ECA taisyklių dėka programos veikimo funkcionalumas gali būti tiksliau nusakytas ir geriau tvarkomas, nei tai būtų užkoduota programos kode.

ECA taisyklės yra aukšto lygio, konstatuojamosios (angl. *declarative*) sintaksės ir pasiduoda analizei ir gerinimo technikai (angl. *optimization techniques*), kurios nelengvai pritaikomos, jei toks pat funkcionalumas išreikštas tiesiogiai programos kode.

ECA taisyklės yra bendras mechanizmas, kuris gali sutraukti plačią veikimo elgsenos įvairovę, priešingai nei programos kodas, kuris yra tipiškai specializuotas tam tikram veikimo scenarijui [40].

2.4. Įvykis

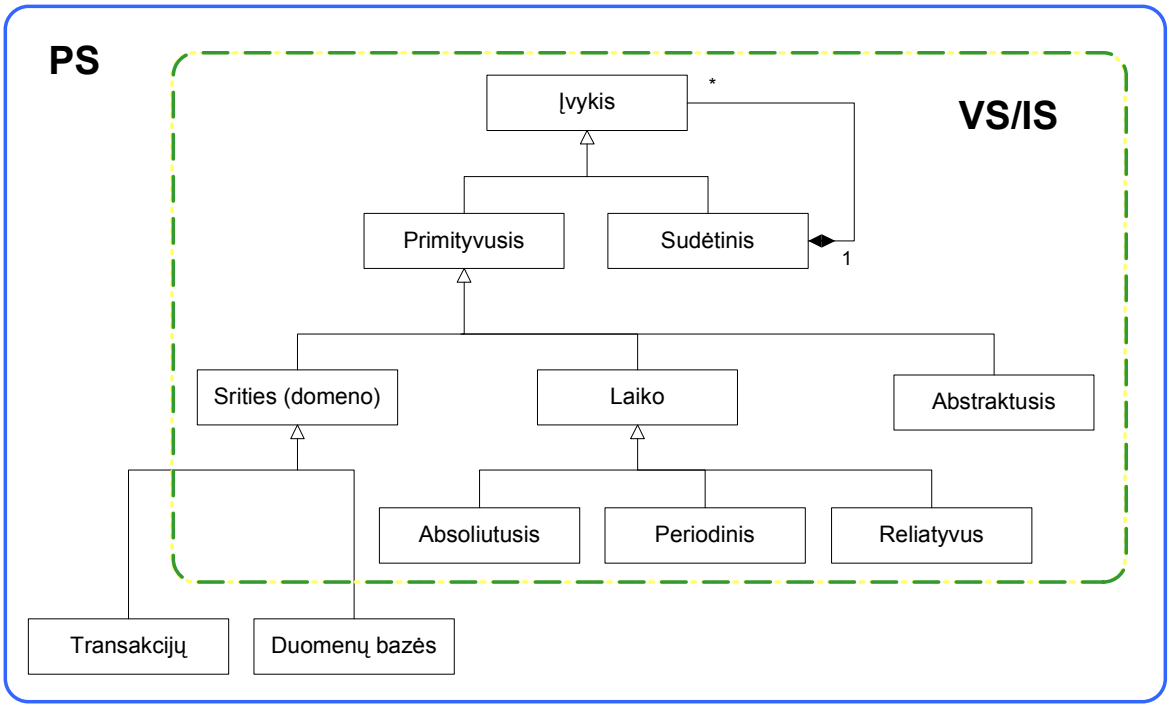
Įvykis (angl. event):

- *kas nors, kas atsitinka tam tikru laiku tam tikroje vietoj,*
- *specialus aplinkybių rinkinys,*
- *fenomenas, vykstantis tam tikroje laiko-erdvės vietoje [39].*

Šie įvykio apibrėžimai yra iš įvairių sričių. Šiam darbui arčiausias yra pirmas apibrėžimas. Tačiau mus domins truputį kitokie verslo ir informacinės sistemos lygmenų įvykių apibrėžimai. Įvykis – tai atsitikimas, pavyzdžiui, komponento būsenos pasikeitimas tam tikru laiko momentu dominančioje vietoje [8, 13]. Įvykis verslo sistemoje – tai verslo atsitikimas, į kurį reaguoja verslo dalyviai.

Verslo lygmens įvykiu gali būti atominė veiksmo, kuris atspindi kažką, kas vyksta realiame pasaulyje, pavyzdžiui, naujo užsakymo sudarymas, dalis. Be įvykių niekas nevyktų: jie atspindi, kaip atsiranda informacija ir objektai, kaip jie keičiasi ir kaip pranyksta [23]. Tai yra grynai verslo sistemos požiūris į įvykį. Skirtinguose lygmenyse įvykio apibrėžimas ir „paskirtis“ praktiškai nesiskiria. Taigi, gali būti situacija, kad įvykis verslo sistemoje gali inicijuoti veiksmą programų sistemoje ir atvirkščiai.

Programų sistemos įvykiai, pavyzdžiui, skirstomi į primityviuosius (angl. *primitive*) ir sudėtinius arba loginius (angl. *complex*). Iš pavadinimų aišku, kad sudėtiniai įvykiai susideda iš primityviųjų. Primityvieji įvykiai yra pavieniai, kurie niekaip nėra sąlygoti kitų įvykių. Primityvieji įvykiai gali būti skirstomi į daugelį klasių, tačiau dažniausiai yra skiriamą keletas: srities, arba domeno, įvykiai (angl. *domain*), laiko įvykiai (angl. *temporal*) ir abstraktieji įvykiai (angl. *abstract/explicit*). Verslo ir informacinės sistemų įvykių klasifikacija yra panaši į programų sistemos, tačiau, kadangi verslo ir informacinės sistemos turi būti nepriklausomos technologiniu ir organizaciniu aspektu [18], joje neturi figūruoti priklausomi nuo tokių aspektų įvykiai (2 pav.).



2 pav. ***Įvykių klasifikacija***

2.4.1. Duomenų bazių įvykiai

Duomenų bazių įvykiai inicijuojami duomenų pakeitimu. Duomenų bazės įvykiai skirstomi į duomenų modifikavimą ir duomenų peržiūrėjimą. Duomenų modifikavimas atitinka tas operacijas, kurios keičia duomenis duomenų bazėje (kaip SQL komandos INSERT, DELTE ir UPDATE reliacinėse duomenų bazėse). Duomenų peržiūrėjimas, pavyzdžiui, yra pasirinkimo tam tikroje reliacinės duomenų bazės lentelėje įvykdymas ar objekto gavimas objektinėse DBVS. Kiti vidiniai atvejai taip pat gali būti apdorojami, bet tai priklauso nuo konkrečios DBVS realizacijos [9, 34].

2.4.2. Transakcijų įvykiai

Transakcijų įvykiai iššaukiami duomenų bazių transakcijų vykdymo metu (SQL komandos COMMIT, BEGIN TRANSACTION it t.t.) [9].

2.4.3. Laiko įvykiai

Laiko įvykiai – įvykiai inicijuojami tam tikru laiko momentu. Laiko įvykiai skirstomi į absoliučiuosius, reliatyviuosius ir periodinius. Absoliutieji laiko įvykio pavyzdžiu galėtų būti tam tikras tikslus laikas tam tikrą dieną (2006-02-02 13:50). Reliatyvieji įvykiai įvyksta praėjus tam tikram laiko tarpui nuo konkretaus laiko taško. Periodiniai įvykiai yra atskiras reliatyviųjų įvykių atvejis, kai įvykis kartojasi kas tam tikrą apibrėžtą laiko tarpą [9].

2.4.4. Abstraktieji įvykiai

Abstraktieji įvykiai priskiriami įvykiams, kurie yra inicijuojami verslo sistemos objekto ar aprašomi programoje, tokie kaip vartotojo prisijungimas, veiksmo iniciavimas ir pan. Programų sistemos atveju, abstraktieji įvykiai gali būti inicijuojami kaip pačios sistemos, taip ir perduodami į sistemą iš išorės [8, 9].

2.4.5. Sudėtiniai įvykiai

Sudėtiniai įvykiai susideda iš primityviųjų įvykių, tarp kurių gali būti įvairiausių loginių ryšių, tokių kaip konjunkcijos, disjunkcijos, įvykių sekos ir t. t. [7, 32]. Specifikuojant sudėtinį įvykį, sistemos kūrėjas privalo nurodyti įvykių apdorojimo seką, tai yra nustatyti kokia tvarka iš įvykių dėklo (angl. *stack*) bus traukiami įvykiai sudėtinių įvykių sudarymui [2].

2.5. Įvykiais valdomos sistemos

Įvykiais valdomos (angl. *event-driven*) sistemos atlieka įvairius aplinkos stebėjimo ir kitus uždavinius. Daugumoje šių sistemų yra realizuotas įvykių sistemos karkasas, apdorojantis gaunamus įvykius [13, 40, 32, 22].

Įvykių sistemose gali būti realizuota pranešimų apie įvykį sistema, kai apie įvykį pranešama visiems suinteresuotiems objektams. Pranešimas yra žinutė, pranešanti apie įvykį besidomintiems klientams [10]. Pranešimas apie įvykį suteikia apibūdinančią informaciją, t. y., su programa susijusius parametrus ir administracinius duomenis, tokius kaip unikalų numerį, laiką (angl. *timestamp*), gyvavimo laiką (angl. *time-to-live*), įvykių šaltinį ir t.t.**.** Vienas iš tokių sistemų realizavimo būdų galėtų būti mechanizmo realizavimas, kai įvykius generuojantys objektai ir įvykiais suinteresuoti objektai registruojasi tam tikroje vietoje ir visi suinteresuoti

įvykiais objektai gauna reikiamą informaciją. Be to, dėl tarpininko įvedimo atsiranda anoniminis įvykių perdavimas ir asinchroninis bendravimas [10].

Įvykiais valdomų informacinių sistemų koncepcija yra vis dažniau naudojama kuriant informacines sistemas, nes sistemų užsakovai vis labiau stengiasi sumažinti informacinių sistemų kainą ir pagerinti sistemų reagavimo į vartotojų užklausas savybę.

Naujos kartos informacinės sistemos, sukurtos ateinančioms judrioms ir greitai dirbančioms verslo sistemoms, skirsis nuo šiuolaikinių. Vienas pagrindinių pokyčių bus įvykiais valdoma informacinių sistemų architektūra [30]. Įvykiais valdomi verslo procesai yra visų pirma verslo sistemos dalis, tačiau jie turi didelę įtaką informacinių sistemų architektūrai, nes informacinės sistemos privalo prisitaikyti prie besikeičiančios aplinkos. Kad architektūra būtų valdoma įvykiais, jai mažiausiai reikia turėti tam tikrą pranešimų apdorojimo dalį (angl. *message-oriented middleware*).

Įvykiai gali būti generuojami ne tik verslo, bet ir kompiuterine sistema. Geras tokios situacijos pavyzdys gali būti dažniniai radijo identifikatoriai (angl. *Radio Frequency Identification (RFID)*), kurie perduoda saugomą informaciją, kai patenka į specialaus radijo imtuvo veikimo sritį. Kai atsitinka toks įvykis, sudaromas laikinas ryšys ir paleidžiamos tam tikros procedūros. Tokios situacijos pavyzdžiu gali būti jūrinio konteinerio, pažymėto unikaliu dažniniu radijo identifikatoriumi, perėjimas per vartus, kuris generuoja *įėjimo* įvykį. Įvykis gali inicijuoti pranešimo, kuriame būtų visa informacija apie konteinerio pristatymo vietą, siuntimą vairuotojui. Be to, pranešimai apie konteinerio judėjimą gali būti siunčiami bet kokioms kitoms suinteresuotoms šalims [38].

2.5.1. Įvykiais valdomos verslo sistemos architektūros samprata

Įvykiais valdomos verslo sistemų architektūros tikslai yra pagreitinti ir „pajudrinti“ (angl. *agile*) verslo procesus. Čia „pagreitinti“ reiškia sumažinti laiką, skirtą procesui, pavyzdžiui, mašinai pagaminti ar draudimo išmokai išmokėti. Taigi verslo sistemos su greita reakcija turi konkurencingą privalumą, pavyzdžiui, įmonės gali parduoti daugiau prekių, teikti geresnes paslaugas ar greičiau už konkurentus pasiekti naujų rinkos segmentų [30]. Be prastovų veikiančios įmonės (angl. *Zero-Latency Enterprise (ZLE)* ar *Real-Time Enterprise (RTE)*) strategija turi savalaikiškumo tikslą. Tokių verslo sistemų tikslas – akimirksniu reaguoti ir tinkamai atsakyti į visus verslo sistemos įvykius, kurie vyksta sistemos viduje ir už jos ribų. Naudojant techninius terminus, prastova (angl. *latency*) yra laikas, reikalingas sistemai, kad ji atsakytų į įvesties duomenis [30].

Judrumas (angl. *agility*) skiriasi nuo greičio, bet yra jam artimas. Judrumas gali būti dviejų tipų: reikalavimų judrumas (angl. *instance agility*) ir procesų judrumas (angl. *process agility*). Taikant reikalavimų judrumo strategiją, skiriamas dėmesys atskirai kiekvienam procesui, pavyzdžiui, skirtingų spalvų ir nustatymų diegimas atskirai į kiekvieną naują automobilį surinkimo linijoje. Procesų judrumas iš esmes keičia procesus, kad šie atitiktų pasaulio pokyčių, pavyzdžiui, automobilių gaminimo linijos pertvarkymas gaminti naujus, atitinkančius rinkos poreikius, kėbulus vietoj prieš tai gamintų. Maksimalus judrumo lygis yra reaguoti į įvykius bet kuriuo laiku, netgi proceso vykdymo metu. Taisyklės, procedūros ir kitos sistemos sudedamosios dalys turi būti pakankamai lanksčios, kad prisitaikytų prie besikeičiančios aplinkos [30].

2.5.2. Įvykiais valdomų verslo procesų modelis

Kaip jau buvo minėta, įvykiai verslo sistemoje paprastai inicijuoja įvykius informacinėje sistemoje. Verslo sistemose buvo įvykių apdorojimo sistemos netgi prieš atsirandant kompiuteriams. Kai verslo sistema atpažįsta verslo įvykį, yra paleidžiamas verslo procesas, žmogiškųjų ir automatinių veiksmų seka. Nors procesas ir yra inicijuojamas įvykių, jis yra nebūtinai įvykiais valdomas nuo pradžios iki galo. Dažnai procesas nustoja būti valdomas įvykiais po pirmojo procesą inicijavusio įvykio. Įvykiais valdomos architektūros privalumas yra tai, kad į procesą siekiama įdiegti naujus įvykių aspektus.

Kaip pavyzdį galima paimti verslą kuriame yra užsiimama kompiuterių pardavimu ir gamyba. Šio verslo specifika yra gaminti kaupimui (angl. *build-to-stock*) Produkcija yra gaminama kaupimui, ir sandėlio ribos yra nustatomos eksperimentiniu būdu. Yra sudaroma produkto pardavimo mastų prognozė ir remiantis ja sudaromi gamybos planai. Yra ir kitas gamybos būdas – gamyba pagal užsakymą (angl. *build-to-order*). Tokiu būdu gamintojas gali per 24 valandas nuo užsakymo pateikimo pristatyti pagamintą pagal šį užsakymą kompiuterį. Kiekvienas verslo sistemos įvykis (užsakymas) inicijuoja gamybos procesą. Tai vadinama tiesioginiu inicijavimu (angl. *direct triggering*) . Tradicinis gamybos kaupimui metodas taip pat priklauso nuo įvykių, bet netiesiogiai. Veikiant pagal tokį gamybos būdą, praeitų įvykių istorija naudojama kaip vedlys tolimesniems veiksmams, bet automatiškai nereaguojama į kiekvieną naują įvykį ar informaciją. Gamybos planuotojai nemato verslo įvykių ir tuo yra atskirti nuo užsakymo vykdymo proceso. Abiems atvejams (gamybos kaupimui (angl. *build-to-stock*) ir gamybos pagal užsakymą (angl. *build-to-order*)) procesai yra vienaip ar kitaip valdomi įvykiais, - tik gamybos pagal užsakymą atveju įmonė gali greičiau reaguoti į pokyčius. Pavyzdžiui, jeigu padidėtų paklausa, gamybos pagal užsakymą strategija pasirinkusi kompanija iškart padidintų

gamybos mastus. Tuo tarpu gamybos kaupimui kompanijai gali pritrūkti saugomos produkcijos kiekio, ir tai gali sąlygoti neužplanuotas išlaidas. Ryšis tarp užsakymo įvykio ir reagavimo į jį (gamybos) gana lėtas ir netiesioginis įmonėse, orientuotose į gamybą kaupimui. Be to, gamybos pagal užsakymą atveju kiekvieno gaminio ypatybės gali būti pritaikytos individualiam vartotojui (masinė gamyba pagal užsakymą).

Įvykiais valdoma architektūra paremtas verslas turi labai mažas produkcijos transportavimo ir saugojimo išlaidas. Be to, kiekvienas produktas gali būti priderintas prie konkretaus užsakymo. Aišku, ne visos verslo sistemos yra pritaikytos gamybai pagal užsakymą. Dažnai naudojamas tarpinis variantas tarp gamybos kaupimui ir pagal užsakymą. Be to, tiekėjai taip pat turi pakeisti savo elgseną, kad galėtų veikti gamybos pagal užsakymą modeliu [30].

Įvykiais valdomas verslo procesas dažnai turi vykti greičiau negu paprastas procesas. Pavyzdžiui kompiuterių gamintojas turi pagaminti kompiuterį mažiau negu per 24 valandas, kad būtų įvykdytas užsakymas, kad tai būtų beveik taip pat greitai kaip gamybos kaupimui atveju, kai kompiuteris jau yra sandėlyje. Tai yra pagrindinė priežastis, kodėl įmonės anksčiau negalėjo pritaikyti gamybos pagal užsakymą koncepcijos. Kad sutrumpėtų verslo proceso vykdymo laikas, galėtų būti pritaikyti tokie veiksmai:

- Nereikalingų proceso žingsnių praleidimas
- Laiko proceso žingsniams pradėti ir tarpų tarp jų sumažinimas
- Proceso žingsnių laiko trumpinimas
- Keleto žingsnių sudėjimas į vieną
- Lygiagretus žingsnių vykdymas
- Kokybės gerinimas ir klaidų kieko mažinimas.

2.5.3. Įvykiais valdomų informacinių sistemų modelis

Įvykiais valdomas procesas nėra paprastas tiesiog pagreitintas procesas. Informacinių sistemų kūrėjai naudoja skirtingus modelius, kūrimo būdus ir įrankius tam, kad informacinės sistemos, sukurtos klasikiniams procesams, būtų paverstos įvykiais valdomomis sistemomis. Architektai ir programų inžinieriai kuria informacines sistemas remdamiesi šiais įvykiais valdomų architektūrų principais:

Apdoroti įvykius individualiai, ne grupėmis. Įvykiais paremta sistema reaguoja į įvykį tuo momentu, kai jis yra generuojamas, o ne laukia iš anksto nustatyto laiko, po kurio galima paimti įvykių grupę.

Bendravimui naudoti ne užklausos dėl įvykio būdą, o panešimą apie įvykį. Įvykio siuntėjas pats informuoja sistemą apie įvykį, o ne laukia, kol sistema jo paprašys. Taigi

informacija yra perduodama labai greitai (priklauso tik nuo siuntėjo) ir niekada nelaukia eilėje [30].

Siųsti pranešimus aktoriams išsirinktinai. Pranešimai yra siunčiami tik tada, kai kas nors įvyko ir reikalauja aktoriaus dėmesio. Tokiu būdu yra kuriama žymiai mažiau ar net visai nekuriama pranešimų lyginant su įprastomis sistemomis [30].

Iškelti verslo taisykles iš programų sistemų. Taisyklių iškėlimas iš programų kodo leidžia lengviau keisti skirtingus informacinių sistemų aspektus, pavyzdžiui, duomenų judėjimo ir procesų valdymo [30].

Modulinė struktūra. Didėles problemos yra skaidomos į mažesnes taip, kad atskiras programos komponentas galėtų būti pataisytas ar pakeistas netrikdant visos sistemos darbo. Moduliai yra slepiami (angl. *encapsulated*) nuo išorinio pasaulio. Prieiti prie modulių naudojama sąsajų ir pranešimų sistema [30].

Dinaminė architektūra. Sistemos moduliai kuriami veikti kaip įvykių šaltiniai (angl. *sender*) arba įvykių gavėjai (angl. *recipients*). Moduliai gali būti pridedami ir šalinami programos vykdymo metu visiškai neveikiant kitų modulių. Dinaminė architektūra dažnai naudojama žinutėmis valdomose mikroprogramose (ŽVM) (angl. *Message-Oriented Middleware (MOM)*), kur taikoma publikavimo ir užsakymo (angl. *pub-sub*) elgsena. Šaltinis siunčia (publikuoja) įvykį į ŽVM, komponavimo tarpininką ar įvykių administratorių, kuris, pažymėdamas įvykį pagal užsakymo taisykles, nulemia tam tikrą paskirties vietą. Suinteresuoti objektai programos vykdymo metu, gali dinamiškai registruotis įvykių gavimui. Dinaminė architektūra leidžia greitai keistis santykiams tarp programų ir verslo vienetų tuo metu, kai mažinama prastovos trukmė ir žmogaus įsikišimas [30].

Sudėtinis įvykių apdorojimas (angl. *Complex Event Processing (CEP)*). Sudėtinis įvykių apdorojimas paremtas aibe centralizuotų įvykių valdymo programų arba įvykių apdorojimo agentų, kurie naudoja filtrus, kad išrinktų reikiamus įvykius, surenka iš paprastų įvykių sudėtinius, atitinkančius tam tikras formas, ir stebi, ar įvykiai nepažeidžia taisyklių. Įvykių valdymo programos pačios gali generuoti ir siuntinėti naujus įvykius [30].

2.5.4. Servisams pritaikyta architektūra

Įvykiais valdoma architektūra dažnai yra siejama su servisams pritaikyta architektūra (SPA) (angl. *Service-Oriented Architecture (SOA)*) [30]. Servisams pritaikyta architektūra yra sistemų tarpusavio suderinimo paradigma, kai jos komponentės servisus kurie laukia kitų komponenčių iškvietimo. SPA sistemos paprastai remiasi asinchroniniu bendravimu tarp komponenčių vienas su vienu (angl. *one-to-one*) stiliumi. Pranešimų siuntimui yra naudojami

tokie standartai kaip WSDL ir SOAP. Tikėtina, kad SPA gali tapti viena dominuojančių IS kūrimo rinkoje [21, 24].

3. Verslo ir informacinės sistemų įvykių vaizdavimo ir modeliavimo analizė

Informacinių sistemų modeliavimo procese gali būti kuriama visa aibė įvairiausių modelių. Šiame darbe pagrindinis dėmesys yra skiriamas modeliams kuriuose dalyvauja įvykiai. Taip galima bus pamatyti įvykių vaidmenį sistemų modeliavime.

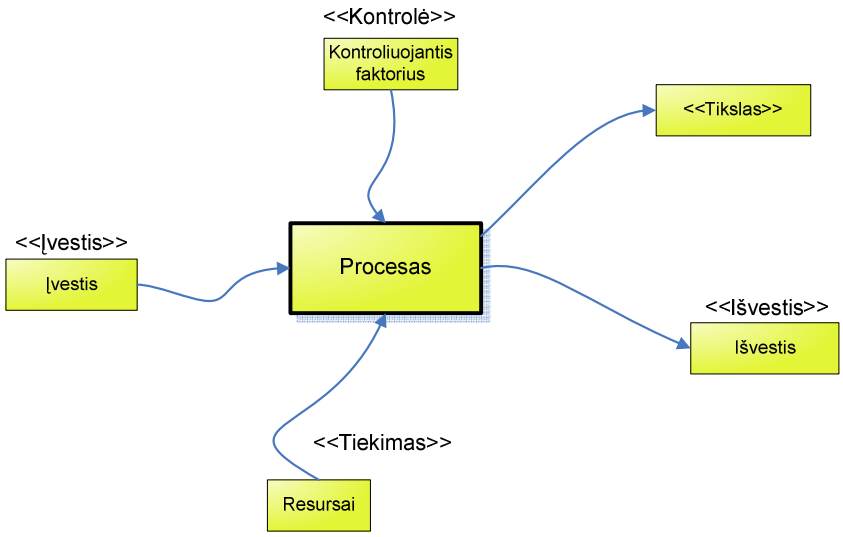
3.1. Įvykių vaidmuo procesų modelyje

Procesas yra tam tikra veiksmų seka, skirta tam tikram tikslui pasiekti. Verslo procesas – tai procesas, vykstantis verslo sistemoje skirtas sukurti naują objektą, pakeisti jau egzistuojantį objektą ar patenkinti verslo dalyvių poreikius. Verslo procesas keičia verslo sistemos būseną.

Procesas negali savaime prasidėti ir pradėti keisti sistemos būseną. Procesą inicijuoja įvykiai. Iš įvykio apibrėžimo suprantama, kad paprastas įvykis yra labai trumpas ir neturi tęsimosi laike, galima sakyti – momentinis objektas. Įvykiu baigiasi procesas ir yra fiksuojamas būsenos pasikeitimas. Neretai pagrindiniai sistemose vykstantis procesai susideda iš kitų smulkesnių procesų, vykstančių skirtinguose lygmenyse (verslo, informaciniame, programų ir t. t.).

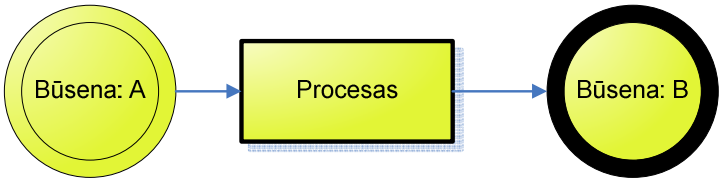
3.1.1. Įvykiai verslo sistemos procesų modelyje

Verslo procesų modelis parodo procesus, vykstančius verslo sistemoje, taip pat procesų įvestis, išvestis ir kitus juos veikiančius faktorius. Diagramoje nurodomi proceso įvestis (objektai, kuris bus keičiami proceso vykdymo metu), kontrolė (procesą kontroliuojantys objektai), tiekimas (resursai, reikalingi procesui), tikslas ir išvestis (3 pav.).



3 pav. ***Proceso modelis***

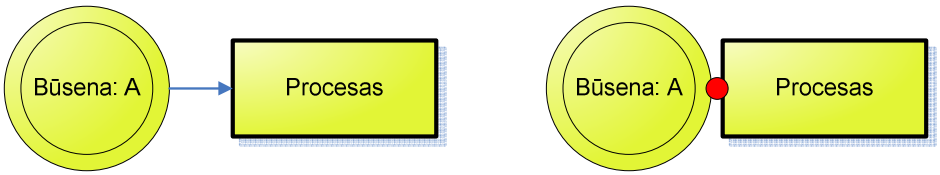
Verslo procesas keičia verslo sistemos būseną. 4 pav. procesas keičia sistemos būseną iš pradinės būsenos A į būseną B.



4 pav. ***Sistemos būsenos keitimas procesu***

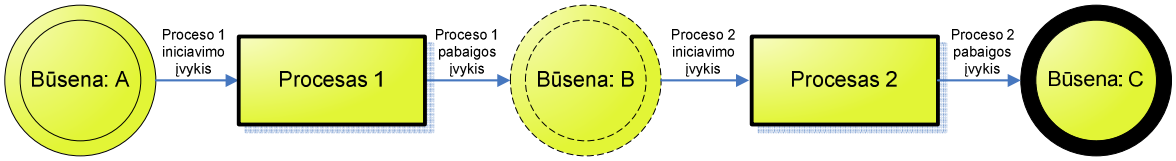
Procesas negali savaime prasidėti ir pradėti keisti sistemos būseną. Procesą inicijuoja įvykiai.

Iš įvykio apibrėžimo suprantama, kad įvykis yra labai trumpas ir neturi tęstinumo laike, galima sakyti, momentinis objektas. Įvykiu pasibaigia procesas ir yra fiksuojamas būsenos pasikeitimas. Tiksliau vaizduoti įvykius būtų objektais, neturinčiais ilgio, tai yra tašku ar apskritimu, bet dažniau yra naudojamos rodyklės, kad nebūtų perkrautos diagramos (5 pav.) .



5 pav. ***Įvykių vaizdavimas modeliuose***

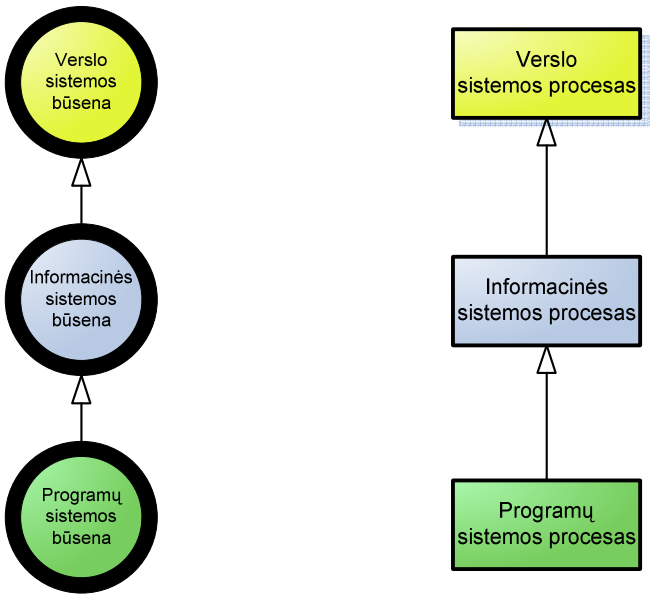
Neretai pagrindiniai sistemose vykstantis procesai susideda iš kitų smulkesnių procesų, taip 6 pav. sistemą iš pradinės į galutinę būseną perveda du smulkesni procesai. Kiekvienas iš smulkesnių procesų perveda sistemą į tam tikrą tarpinę stabilią arba nestabilią būseną.



6 pav. ***Proceso detalizavimas***

3.1.2. Informacinės ir programų sistemų procesų modelis

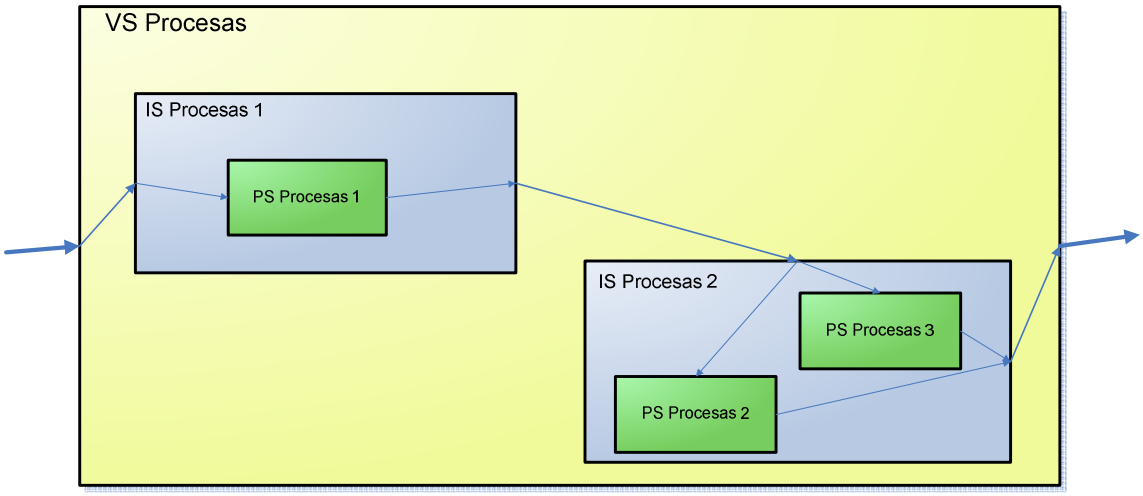
Kiekvieną verslo sistemos būseną atitinka informacinės sistemos būsena. Taip pat verslo procesą gali atitikti vienas ar keli informacinės ir programų sistemų procesai. Tai yra sistemos būsenų ir procesų dekompozicija. (7 pav.).



7 pav. ***Sistemų būsenų ir procesų dekompozicija***

Procesai informacinėje ir programų sistemose neturėtų vykti savaime, taip pat nevyksta atskirai vieni nuo kitų. Jeigu vyksta koks nors programų sistemų lygmens procesas, tai jo veikimo rezultato reikalauja informacinės sistemos procesas.

Procesai gali būti inicijuojami ne tik to paties lygmens, bet ir aukštesnių arba žemesnių lygmenų įvykiais. Yra situacijų, kai procesas yra inicijuojamas tiesiogiai žemesnio lygmenų įvykiais, tačiau dažniausiai pirma yra inicijuojamas, nors ir vienareikšmiškai atitinkantis, įvykio lygmens procesas, o jau paskui – žemesnio lygmens procesas, kaip aukštesnio proceso sudedamoji dalis. Tokia pat situacija ir su aukštesnių lygmenų procesų inicijavimų – pirma įvykis sužadina aukštesnio lygmens įvykį, o jau tik po to aukštesnio lygmens įvykis inicijuoja procesą (8 pav.).



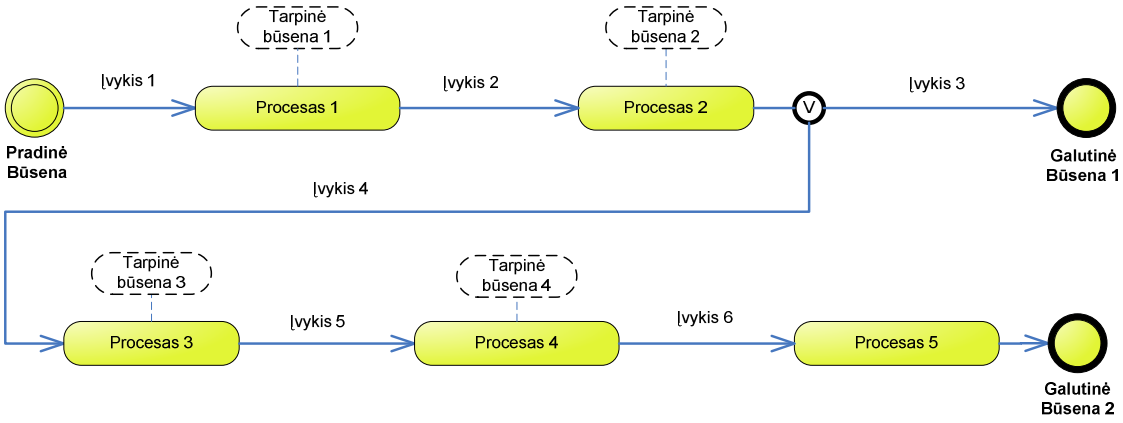
8 pav. ***Verslo proceso dekompozicija***

3.2. Įvykiai išteklių modelyje

Kad vyktų verslo sistemos procesai, reikia išteklių. Be išteklių verslo sistema yra praktiškai nefunkcionali. Modeliavimo proceso metu naudojami du išteklių modeliai: statinis ir dinaminis. Statiniame modelyje vaizduojamas statinis išteklių vaizdas ir įvykiai jame nedalyvauja.

Verslo sistemos ištekliai, kaip ir pati sistema gali, būti skirtingų būsenų. Analogiškai kaip ir su sistemomis, išteklių būsenos yra keičiamos procesais, inicijuojamais įvykiais. Šitas aspektas ir yra parodomas dinaminiame išteklių modelyje.

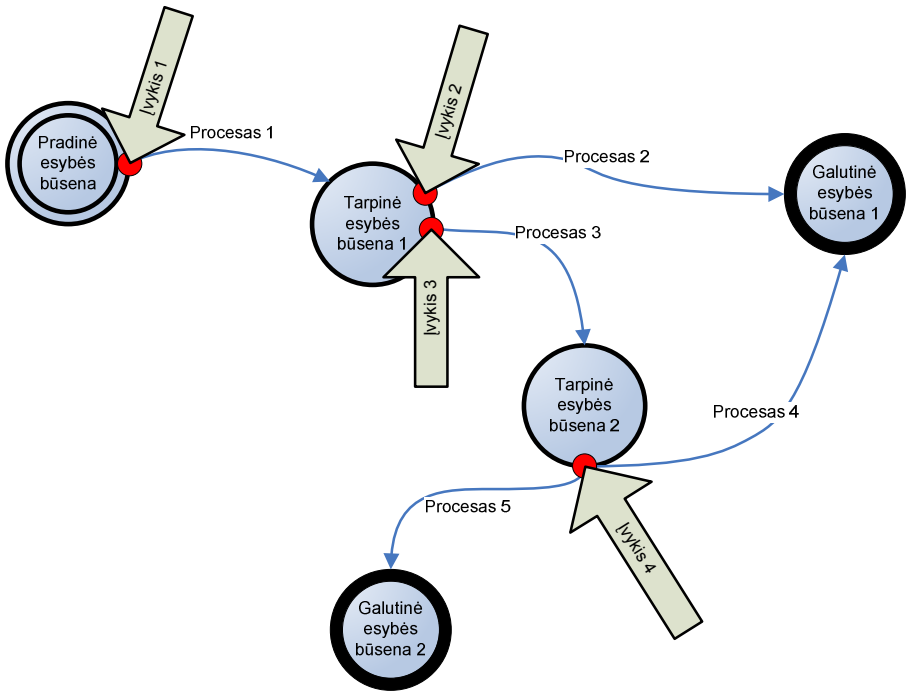
Verslo sistemos ištekliai informacinėje sistemoje yra transformuojami į esybes. Esybės savo gyvavimo metu taip pat gali turėti tam tikras būsenas, tai yra parodyta esybių gyvavimo ciklo modelyje. Esybių būsenas keičia procesai, kuriuos inicijuoja įvykiai. Esybių gyvavimo ciklo analizės tikslas yra nustatyti galimas būsenas, kuriose teisėtai gali būti esybė (9 pav.).



9 pav. ***Išteklio dinaminis modelis***

3.3. Įvykiai esybių gyvavimo ciklų modeliuose

Verslo sistemos ištekliai informacinėje sistemoje yra transformuojami į esybes. Esybės savo gyvavimo metu gali turėti tam tikras būsenas, kas yra parodyta esybių gyvavimo ciklo modelyje. Esybių būsenos keičia procesai, kuriuos inicijuoja įvykiai (10 pav.). Esybių gyvavimo ciklo analizės tikslas yra nustatyti galimas būsenas, kuriose teisėtai būti gali esybė.

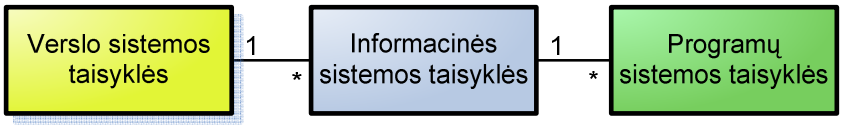


10 pav. ***Esybės gyvavimo ciklas***

Kadangi diagramoje 10 procesai yra parodomi rodyklėmis, įvykių vaizdavimui yra naudojami maži apskritimai.

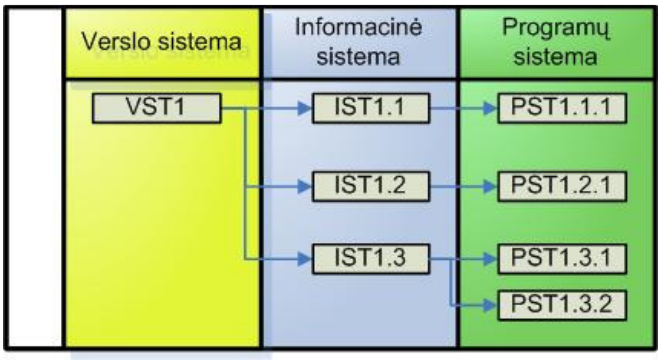
3.4. Taisyklių ir įvykių modelis

Informacinės sistemos, paremtos verslo taisyklėmis, gali tiksliai atitikti verslo poreikius. Taisyklės gali būti dokumentuotos tiesiog nesistemiškai surašant jas vieną po kitos. Toks būdas galėtų tikti mažoms sistemoms, kur taisyklių yra nedaug, tačiau, sistemoms didėjant, didėja ir verslo taisyklių, reikalaujančių apdorojimo, kiekis. Vienai taisyklei verslo sistemoje gali atitikti kelios informacinės sistemos taisyklės, taip pat ir vienai informacinės sistemos taisyklei gali atitikti kelios programų sistemos taisyklių (11 pav.).



11 pav. ***Verslo, informacinės ir programų sistemų taisyklių atitikinimas***

Tad yra tikslinga sistemingai užrašyti taisykles, jas suskirstant pagal jų šaltinius ir hierarchiją. Sukūrus tokį verslo sistemos taisyklių sąrašą, galimą sudaryti jų transformacijos į informacinės ir programų sistemų taisykles modelį. Taisyklių transformacijos modelis leidžia sekti taisyklių perėjimą iš verslo sistemos lygmens į programų sistemos lygmenį. Pasikeitus verslo taisyklėms galima skirti dėmesį tiktai toms informacinio ir programų lygmens taisyklėms, kurios yra susietos su pasikeitusiomis taisyklėmis (12 pav.).

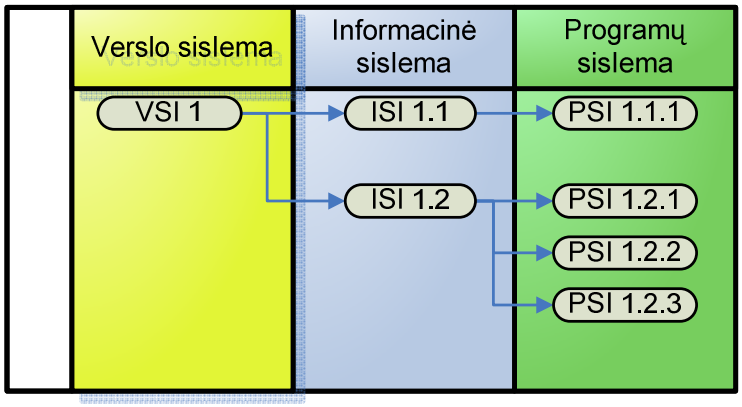


12 pav. ***Taisyklių transformacijos modelis***

Taisyklės yra inicijuojamos įvykiais o vykdomos – per procesus. Įvykių atitikimas lygmenimis yra toks pat kaip ir taisyklių (13 pav.):

13 pav. ***Verslo, informacinės ir programų sistemų įvykių atitikinimas***

Sudarius taisyklių transformacijos modelį galima sudaryti įvykių modelį, kuriame būtų parodyti įvykiai ir jų perėjimą iš verslo sistemos lygmens į programų sistemų lygmenį (14 pav.). Įvykių sistema yra nepriklausoma nuo taisyklių sistemos, tai reiškia, kad įvykių modelį galima sudaryti ir be taisyklių modelio, tačiau pastarasis leidžia išskirti tik tuos verslo sistemos įvykius, kurie turi būti apdorojami informacinėje sistemoje.



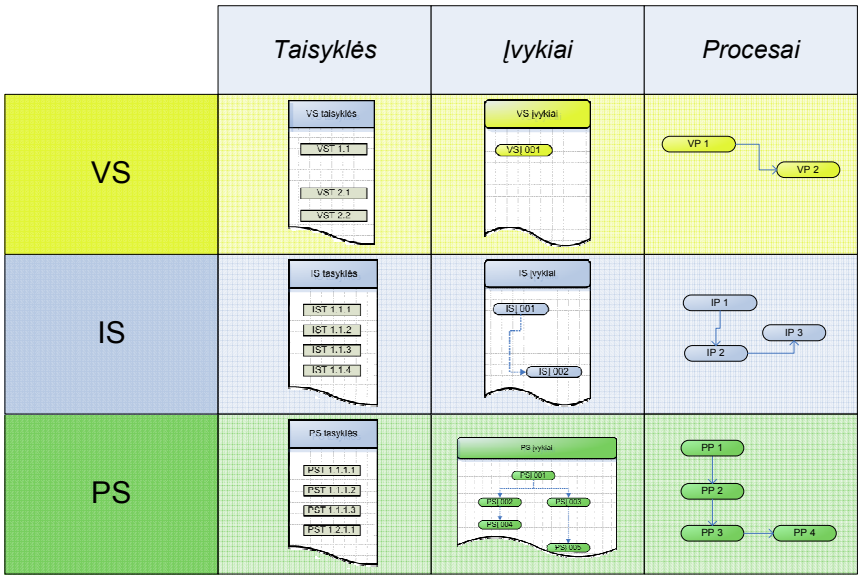
14 pav. ***Įvykių modelis***

Įvykių modelio nauda yra ta, kad susiejus įvykius su procesais, vykstančiais sistemose, visada yra matomas proceso ir įvykio šaltinis ir, įvykus pasikeitimams bet kuriame iš lygmenų, aiškiai matosi, kuriuos procesus ar įvykius reikia pakeisti nekeičiant kitų sistemos dalių.

4. Verslo ir informacinės sistemų įvykių modelio naudojimas informacinės sistemos kūrime

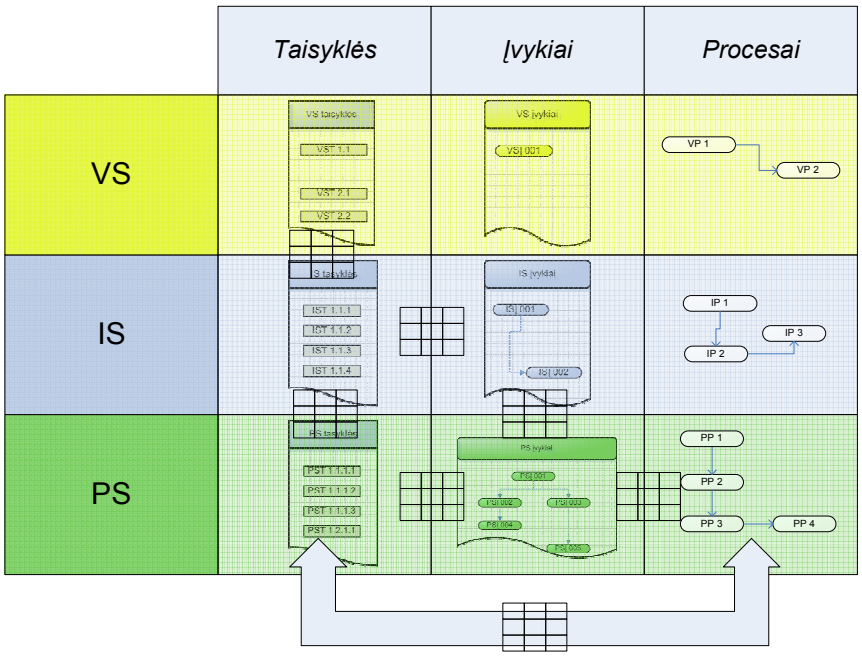
Verslo ir informacinės sistemos įvykių vaizdavimui ir modeliavimui tirti ir analizuoti buvo kuriamas realios verslo sistemos modelis. Darbe pagrindinis dėmesys skiriamas tiems verslo ir informacinės sistemų objektams, kurių gyvavimo cikle dalyvauja įvykiai. Tokiu būdu galima pamatyti bei parodyti įvykių vaidmenį sistemų modeliavime ir informacinių sistemų kūrime.

Taisyklės yra inicijuojamos įvykių ir vykdomos per procesus. Remiantis tuo galima apjungti taisyklių, įvykių ir procesų modelius į vieną bendrą karkasą (15 pav.), kur eilutėse atvaizduoti verslo, informacinės ir programų sistemų lygmenys, o stulpeliuose – taisyklės, įvykiai ir procesai.



15 pav. ***Taisyklių, procesų ir įvykių karkasas***

Verslo ir informacinės sistemų taisyklių modeliai sudaromi ir susiejami informacinės sistemos realizavimo metu., Praktiškai neįmanoma nusakyti kaip bus transformuojamos verslo ir informacinės sistemų taisyklės į programų sistemos taisykles, kadangi programų sistemos lygmuo yra priklausomas nuo technologinio aspekto. Programų sistemos taisyklės priklauso nuo to, kokiuose PS procesuose jos bus inicijuojamos, o inicijuojamos jos yra įvykių. Šioje situacijoje įvykių modelis padeda susieti procesus su taisyklėmis. Bet to, specifikuojant įvykius neretai pastebimos anksčiau (per taisyklių specifikavimą) praleistos ir nespecifikuotos taisyklės.



16 pav. ***Sistemų specifikavimas naudojant taisyklių, procesų ir įvykių karkasą***

16 pav. galime matyti, kad yra kuriami tarpusavyje susieti VS ir IS taisyklių modeliai. Informacinės sistemos taisyklių modelis yra siejamas su informacinės sistemos įvykių modeliu. Programų sistemos kūrėjas, matydamas informacinės sistemos taisyklių ir įvykių specifikacijas ir išmanydamas programų sistemos ypatumus, gali nesunkiai aprašyti PS įvykius ir jų inicijuojamus procesus. Turėdamas PS įvykių sistemos modelį, PS kūrėjas, matydamas, kokius procesus inicijuoja įvykiai, gali tiksliai aprašyti į kokias PS taisykles yra transformuojamos IS lygmens taisyklės.

4.1. Tiriamos verslo sistemos dalykinės srities aprašymas

Modeliavimo objektu buvo parinkta kompiuterinės, elektroninės bei programinės įrangos pardavimu ir aptarnavimu užsiimanti įmonė. Tokio tipo įmonėse realizuotos klasikinės užsakymo, pardavimo ir aptarnavimo schemos.

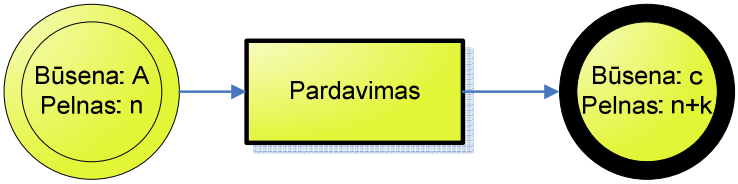
Dalykinė sritis: įmonėje vykstantis darbas su klientais, bei įrangos užsakymo, pardavimo ir serviso valdymas.

Probleminė sritis: įmonėje yra netinkamai organizuotas ir formalizuotas darbas su klientais. Didelės laiko sąnaudos įrangos pardavimo ir serviso teikimo grandinėje. Nuostoliai dėl neoptimalaus garantinio aptarnavimo teikimo. Prekių pirkimas, pardavimas praktiškai nekompiuterizuotas, dėl ko nuolat iškyla terminų nesilaikymo problemos

4.2. Įvykiai verslo ir informacinės sistemų modeliuose

Nagrinėjamoje sistemoje vyksta vienas pagrindinis procesas: prekių pardavimas.

Pardavimo proceso tikslas yra parduoti prekę klientui. Paprastai tariant, verslo procesas *pardavimas* keičia verslo sistemos būseną iš būsenos A = {Įmonė su pelnu n} į būsena B = {Įmonė su pelnu n+p}.

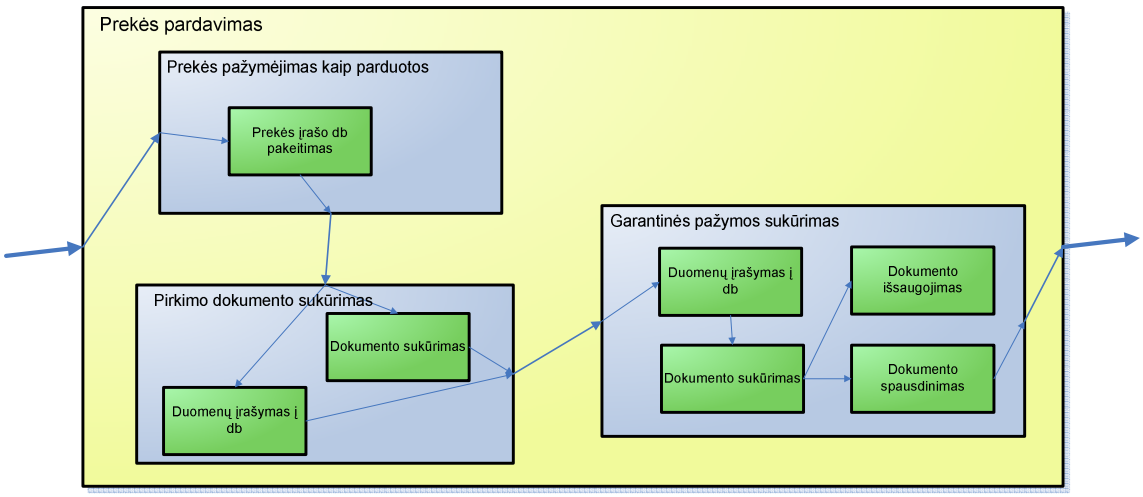


17 pav. ***Sistemos būsenos keitimas pardavimo procesu***

Pardavimo procesas susideda iš procesų *prekės užsakymas* ir *prekės pardavimas*. Kiekvienas iš smulkesnių procesų perveda sistemą į tam tikrą tarpinę būseną (18 pav.).

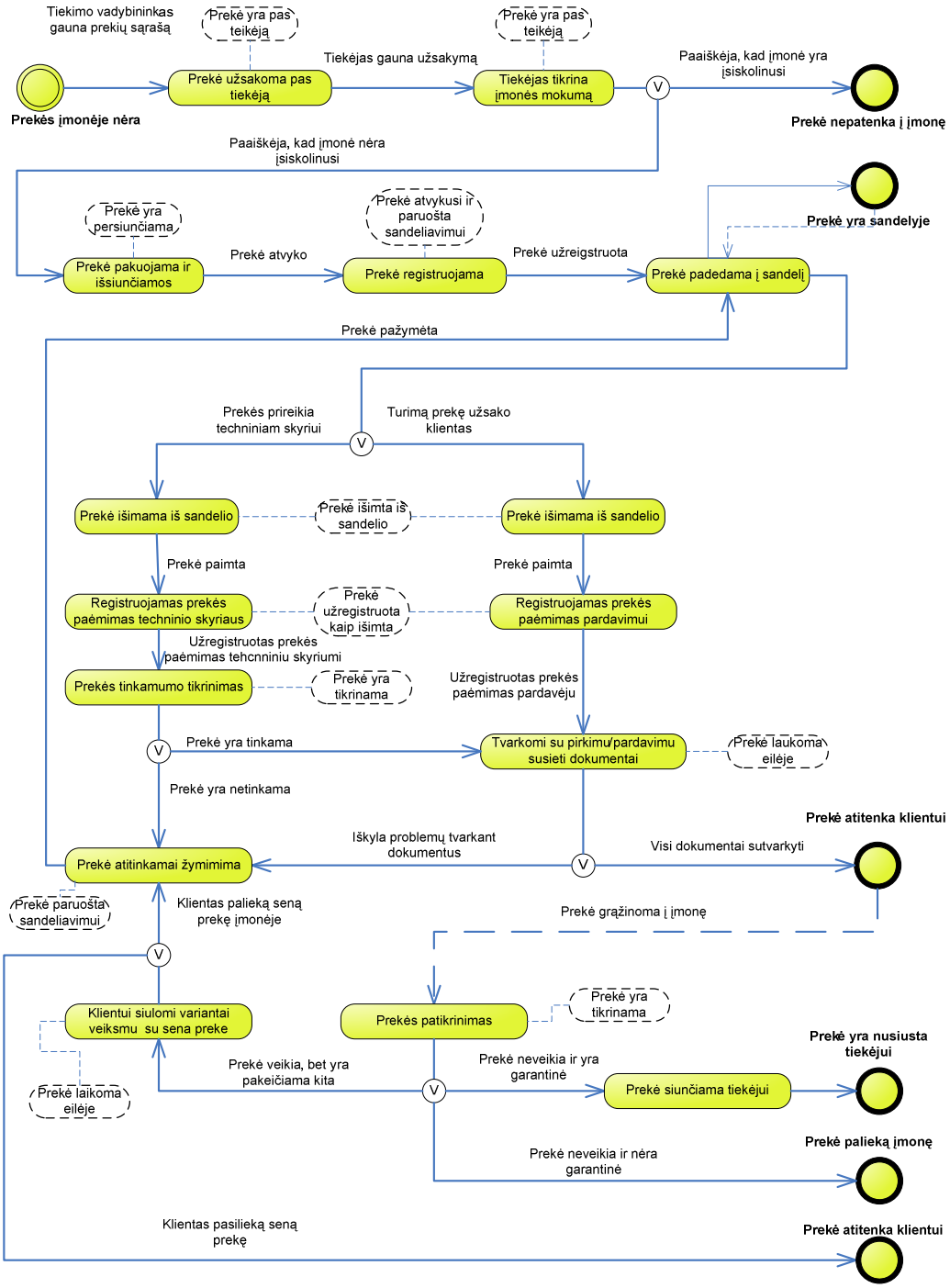
18 pav. ***Pardavimo proceso detalizavimas***

Tą patį pardavimo procesą galima dekomponuoti iki programų sistemų lygmens (19 pav.). 19 pav. matome, kad *prekės pardavimo* verslo procesas susideda iš trijų smulkesnių informacinės sistemos procesų, tuo tarpu, kiekvienas jų yra sudarytas iš tam tikrų programų sistemos procesų.



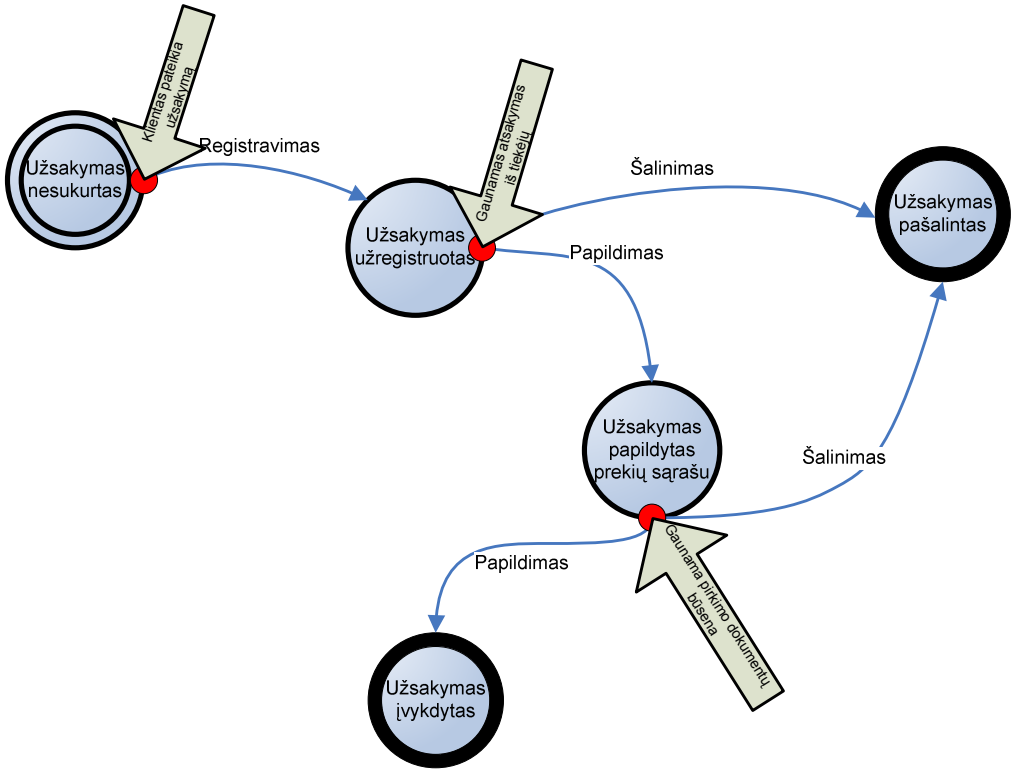
19 pav. ***Prekės pardavimo proceso dekompozicija***

Vienas pagrindinių pardavimo procese dalyvaujančių išteklių yra prekė. Prekė, kaip išteklius, savo gyvavimo metu gali pasiekti tam tikras būsenas. Išteklių modelis (20 pav.) parodo kaip gali keistis prekės būsenos, kokie procesai jas keičia ir kokiais įvykiai procesai yra inicijuojami.



20 pav. ***Išteklio „prekė“ dinaminis modelis***

Verslo sistemos objektai yra transformuojami į informacinės sistemos esybes. Kaip pavyzdys 21 pav. yra pavaizduotas esybės „užsakymas“ gyvavimo ciklo modelis. Modelyje matyti, kad užsakymą perveda į tam tikras būsenas įvykiais inicijuojami procesai.



21 pav. ***Esybės „užsakymas“ gyvavimo ciklas***

4.3. Taisyklių modelis

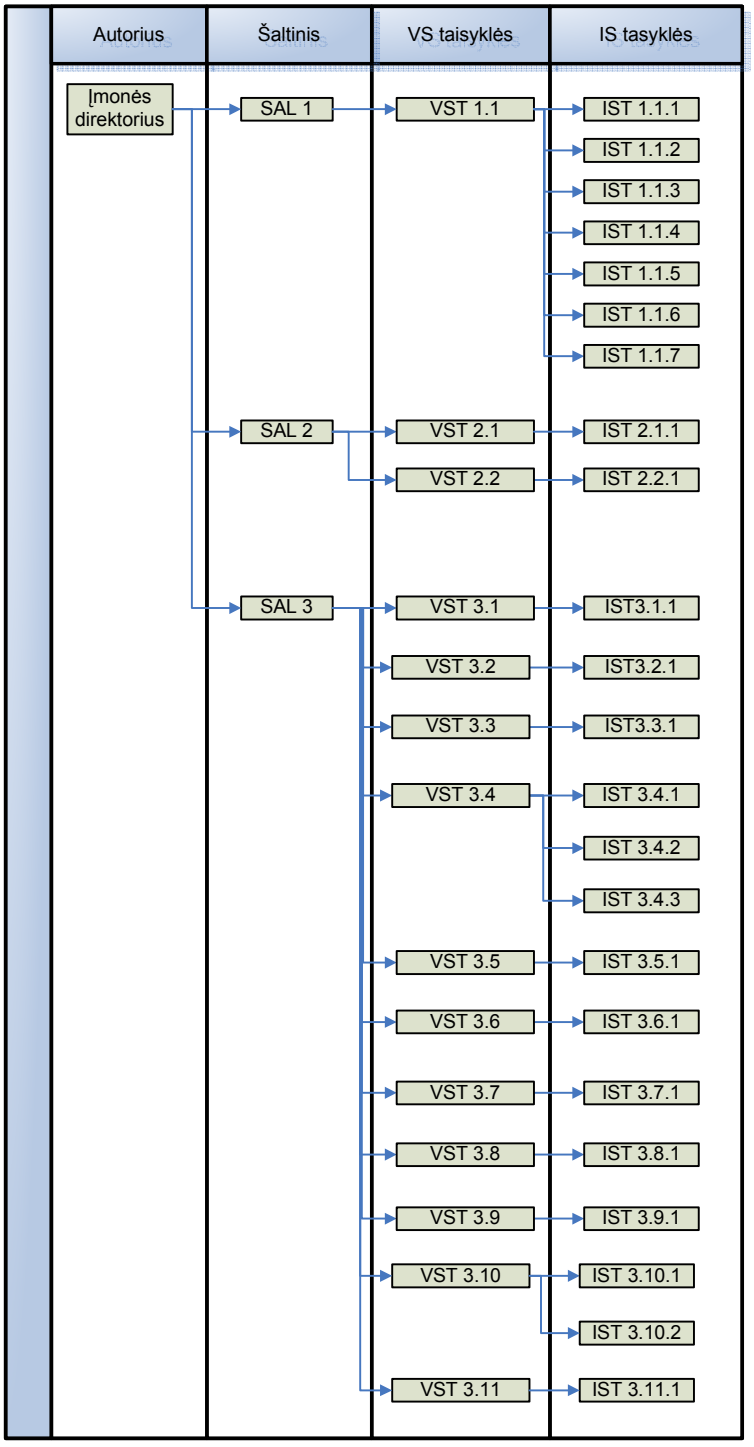
Kaip jau buvo minėta aukščiau, verslo taisykles ir joms atitinkančios informacinės ir programų sistemų taisykles patogu pavaizduoti lentele arba schema. Šiame darbe yra naudojami abu būdai, nes lentelės pagalba mes galime pamatyti tikslų kiekvienos taisyklės aprašymą, o schemoje – jų (taisyklių) tarpusavio priklausomybę. Verslo ir informacinės sistemų lygmenys yra nepriklausomi nuo technologijų, todėl nesunku (nepradėjus realizavimo) išskirti VS ir IS taisykles bei jų sąryšius. Programų sistemų taisyklės tiesiogiai priklauso nuo konkrečios realizacijos todėl pradinio modeliavimo fazėje jų tiksliai ir tinkamai nusakyti praktiškai neįmanoma . 1 lent. yra aprašytos VS ir IS taisyklės, jų šaltinis (paprastai egzistuojantis įmonėje dokumentas arba tiesiog neformalizuotų nuostatų rinkinys) ir autorius. Taisyklių autoriumi gali būti tiek bet koks susijęs su įmone asmuo (kita įmonė, direktorius, vadybininkas ir t. t.), tiek ir įvairiausios nuostatos ir valstybės ar kitų institucijų kuriami įstatymai

1 lentelė. ***Verslo ir informacinės sistemų taisyklių sistemos modelis***

Autorius	Šaltinis	VS taisyklės	IS taisyklės
Bendrovės direktorius	1. Bendrovės įstatai (klientų teisės ir pareigos)	1.1 Jeigu klientas pageidauja jis gali užsiregistruoti	1.1.1 Klientas negali būti įkeltas į sistemą daugiau negu vieną kartą
			1.1.2 Jeigu klientas yra fizinis asmuo, į sistemą turi būti įkelti kliento vardas, pavardė ir papildoma informacija.
			1.1.3 Jeigu klientas yra juridinis asmuo į sistemą turi būti įkleti kliento įmonės pavadinimas, įmonės kodas, įmonės PVM mokėtojo kodas.
			1.1.4 Juridinio kliento vardas, įmonės kodas ir PVM kodas turi būti unikalus.
			1.1.5 Juridinio kliento registracijos numeris turi būti 9 simbolių ilgio
			1.1.6 Juridinio kliento PVM kodas turi būti 11 simbolių ilgio.
			1.1.7 Juridinio kliento duomenys turi būti patikrinti įmonių registre.
	2. Prekių registravimo taisyklės	2.1 Gavus prekę iš tiekėjo, reikia užrašyti jos duomenis ir padėti ją į sandėlį.	2.1.1 Į sistemą turi būti įkelti tiekėjo duomenys, prekės pavadinimas, gamintojas, modelis, serijinis numeris ir kaina.
		2.2 Jeigu prekė yra užsakyta tai ją reikia atidėti.	2.2.1 Jeigu prekė yra užsakyta ji turi būti pažymėta kaip rezervuota.
	3. Prekių pardavimo / užsakymo taisyklės	3.1 Pirkėjui parduodama prekė turi būti sandėlyje	3.1.1 Parduodama prekė turi būti sandėlyje.
		3.2 Pirkėjui parduodama prekė negali būti rezervuota.	3.2.1 Parduodama prekė neturi būti rezervuota.
		3.3 Užsakoma prekė turi buti pas tiekėją.	3.3.1 Turi būti patikrintas prekės buvimas pas tiekėją.
		3.4 Kliento pageidavimu reikia užsakyti prekę.	3.4.1 Reikia sukurti prekės užsakymą.
			3.4.2 Reikia įtraukti prekę į užsakomų prekių sąrašą.
			3.4.3 Turi būti įrašyta informacija apie planuojamą atvykimą
		3.5 Padarius užsakymą reikia patikrinti kliento patikimumą	3.5.1 Sukūrus užsakymą reikia patikrinti ar klientas yra patikimas
		3.6 Jeigu klientas nėra „patikimas“ reikia paimti iš jo avansą	3.6.1. Jeigu klientas nėra „patikimas“ turi būti paimtas avansas
		3.7 Turi būti galimybė pamatyti užsakymų sąrašą	3.7.1. Turi būti galimybė pamatyti užsakymų sąrašą
		3.8 Jeigu prekė neatvyko reikia panaikinti jos užsakymą.	3.8.1. Negavus prekės, reikia pašalinti ją iš užsakomų sąrašo.
		3.9 Turi būti įrašyti čekio ir/ar sąskaitos duomenys	3.9.1 Tūri būti įrašyti čekio ir/ar sąskaitos duomenys.
		3.10 Perdavus prekę klientui reikia išrašyti garantiją.	3.10.1. Atspausdinti garantinę pažymą
			3.10.2. Užrašyti garantinės pažymos informaciją.

		3.11 Jeigu klientas neatsiima užsakytos prekės jis tampa nepatikimu	3.11.1 Jeigu klientas neatsiima užsakytos prekės jis tampa nepatikimu

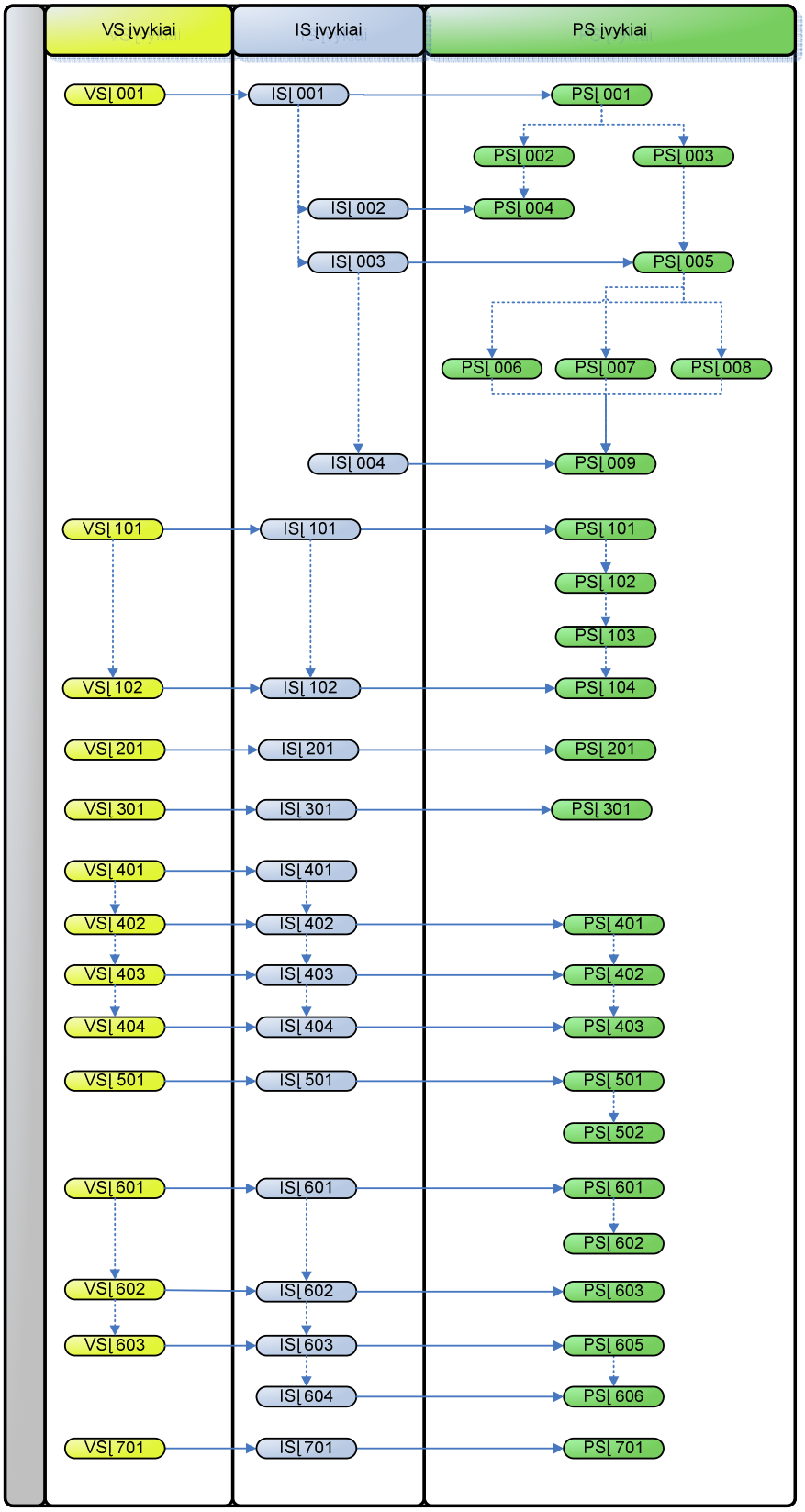
Tam kad būtų patogiau stebėti taisyklių transformavimą ir tarpusavio ryšį jas patogiau pavaizduoti diagrama (22 pav.).



22 pav. ***Taisyklių transformavimo diagrama***

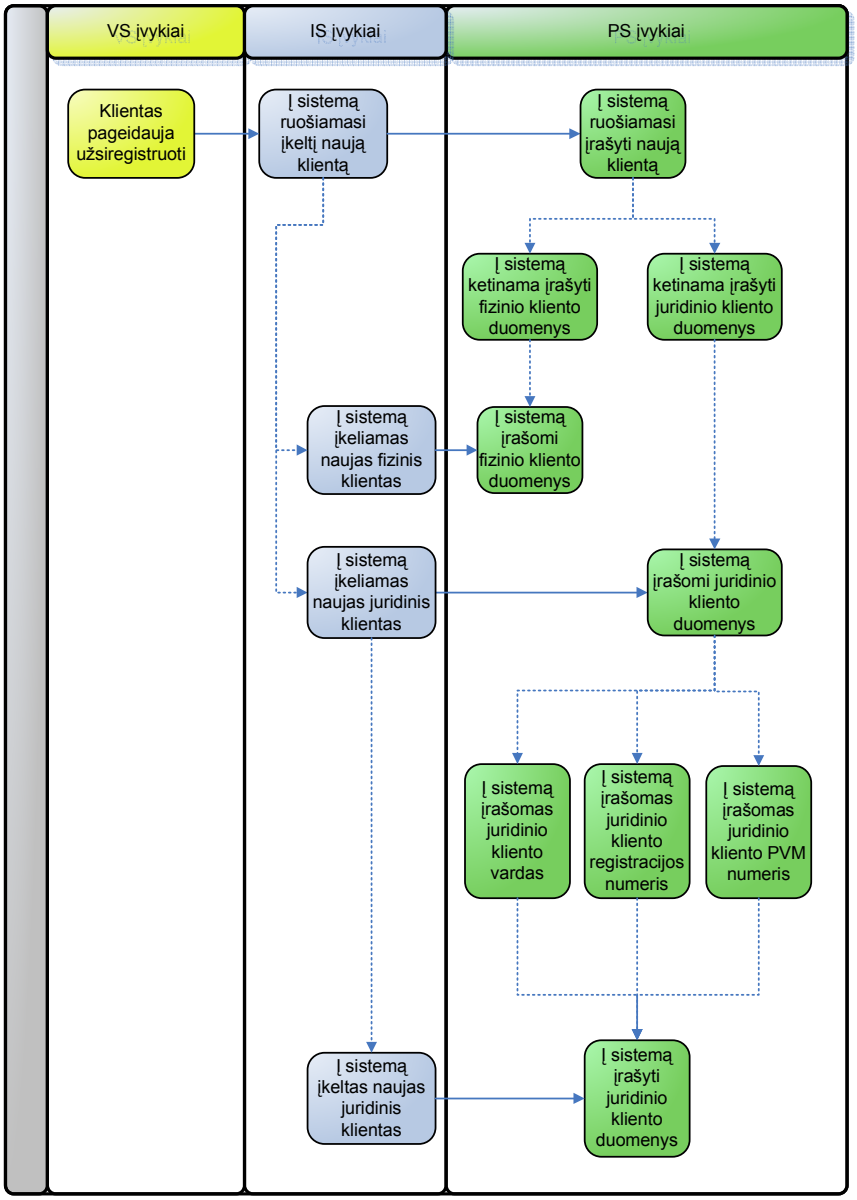
4.4. Įvykių modelis

Įvykius, kaip ir taisykles, galima atvaizduoti lentele arba diagrama. Nors šiuo atveju taip pat vaizdingiau naudoti diagramą, tačiau, dėl didelio reikalingos papildomos informacijos kiekio (įvykio aprašymas, inicijuojamas procesas ir pan.) tenka naudotis ir lentelėmis. 23 pav.. parodytas bendras nagrinėjamos srities įvykių sistemos modelis. Modelyje (23 pav.) yra parodyta įvykių dekompozicija nuo verslo sistemos iki programų sistemos lygmenų. Įvykiai yra pavaizduoti stačiakampiais su užapvalintais kampais. Mėlynomis ištisomis rodyklėmis vaizduojamas įvykio perėjimas iš vieno lygmens į kitą, o punktyrinėmis rodyklėmis (patogesniam suvokimui) pavaizduotos įvykių sekos.



23 pav. ***Įvykių susiejimo diagrama***

Įvykių diagramą galima papildyti įvykių aprašymais (24 pav.), tačiau, tai padidina diagramą ir apsunkina jos suvokimą.



24 pav. ***Detali įvykių susiejimo diagrama***

Autoriaus nuomone patogiau butų naudotis lentelėmis detaliam įvykių aprašymui, o diagrama naudotis kaip vedliu. 2 lent. pateiktas detalesnis įvykių, pavaizduotų 23 pav., aprašymas.

2 lentele. ***Verslo, informacinės ir programų sistemų įvykių modelio detalus aprašymas***

VS įvykiai	IS įvykiai	PS įvykiai
VSĮ 001: Klientas pageidauja užsiregistruoti	ĮSĮ 001: Į sistemą ruošiamasi įkeltį naują klientą	PSĮ 001: Į sistemą ruošiamasi įrašyti naują klientą
		PSĮ 002: Į sistemą ketinama įrašyti fizinio kliento duomenis
		PSĮ 003: Į sistemą ketinama įrašyti juridinio kliento duomenis
	ĮSĮ 002: Į sistemą įkeliamas naujas fizinis klientas	PSĮ 004: Į sistemą įrašomi fizinio kliento duomenys
	ĮSĮ 003: Į sistemą įkeliamas naujas juridinis klientas	PSĮ 005: Į sistemą įrašomi juridinio kliento duomenys
		PSĮ 006: Į sistemą įrašomas juridinio kliento vardas
		PSĮ 007: Į sistemą įrašomas juridinio kliento registracijos numeris
		PSĮ 008: Į sistemą įrašomas juridinio kliento PVM numeris
	ĮSĮ 004: Į sistemą įkeltas naujas juridinis klientas	PSĮ 009: Į sistemą įrašyti juridinio kliento duomenys
VSĮ 101: Prekė gauta iš tiekėjo	ĮSĮ 101: Į sistemą įkeliama nauja prekė	PSĮ 101: Į sistemą ketinama įrašyti naują prekę
		PSĮ 102: Į sistemą įrašoma nauja prekė
		PSĮ 103: Prekė yra įrašyta į sistemą
VSĮ 102: Gauta prekė yra užsakyta	ĮSĮ 102: Įkelta prekė yra užsakyta	PSĮ 104: Įrašyta prekė yra užsakyta
VSĮ 201: Klientas neatsiima prekės	ĮSĮ 201: Gauta užsakyta prekė nėra atsiimama	PSĮ 201: Įrašyta užsakyta prekė nėra atsiimama
VSĮ 301: Klientas pageidauja pirkti prekę	ĮSĮ 301: Ketinama parduoti prekę	PSĮ 301: Ruošiamasi parduoti prekę
VSĮ 401: Klientas pageidauja užsakyti prekę	ĮSĮ 401: Ketinama užsakyti prekę	-
VSĮ 402: Prekė yra pas tiekėją	ĮSĮ 402: Prekė yra pas tiekėją	PSĮ 401: Prekė yra pas tiekėją
VSĮ 403: Sukuriamas naujas užsakymas	ĮSĮ 403: Sukuriamas naujas užsakymas	PSĮ 402: Sukuriamas naujas užsakymas
VSĮ 404: Klientas yra nepatikimas	ĮSĮ 404: Klientas nėra patikimas	PSĮ 403: Klientas nepažymėtas kaip patikimas
VSĮ 501: Užsakyta prekė neatvyksta	ĮSĮ 501: Gautas pranešimas kad prekė neatvyko	PSĮ 501: Gautas pranešimas kad prekė neatvyko
		PSĮ 502: Pašalintas prekės užsakymas
VSĮ 601: Prekė yra parduoda klientui	ĮSĮ 601: Prekė perkeliama į parduodamų aibę	PSĮ 601: Yra pažymima parduodama prekė
		PSĮ 602: Patvirtinamas prekės pardavimas
VSĮ 602: Sukurti čekis ir sąskaita	ĮSĮ 602: Sukurti čekis ir sąskaita	PSĮ 603: Gauti čekio ir sąskaitos duomenys
VSĮ 603: Prekė yra perduota klientui	ĮSĮ 603: Prekė yra įkelta į parduotų aibę	PSĮ 604: Prekė yra įrašyta į parduotų sąrašą
	ĮSĮ 604: Sukurta garantinė pažyma	PSĮ 605: Sukurta garantinė pažyma
VSĮ 701: Ketinama peržiūrėti užsakymus	ISĮ 701: Ketinama peržiūrėti užsakymų sąrašą	PSĮ 701: Ketinama gauti užsakymų sąrašą

Programų sistemos įvykiai inicijuoja programų sistemos procesus per kuriuos yra realizuojamos taisyklės. Aprašant programų sistemą naudinga pažymėti ne tik koks procesas kokio įvykio yra inicijuojamas, bet ir to įvykio šaltinį ir aprašymą (3 lent.).

3 lentele. ***Programų sistemos įvykių sistemos modelio detalus aprašymas***

Įvykis	Įvykio aprašymas	Įvykio šaltinis	Inicijuojamo PS proceso įvestis
PSĮ 001	Į sistemą ruošiamasi įrašyti naują klientą	Naudotojo sąsaja (cMainMenuInsertClient)	InsertClientClick()
PSĮ 002	Į sistemą ketinama įrašyti fizinio kliento duomenis	Naudotojo sąsaja (cInsertNaturalClientRadio)	InsertClientSelect()
PSĮ 003	Į sistemą ketinama įrašyti fizinio kliento duomenis	Naudotojo sąsaja (cInsertLegalClientRadio)	InsertClientSelect()
PSĮ 004	Į sistemą įrašomi fizinio kliento duomenys	Naudotojo sąsaja (cSaveNewClientButton)	SaveNewClient()
PSĮ 005	Į sistemą įrašomi juridinio kliento duomenys	Naudotojo sąsaja (cSaveNewClientButton)	SaveNewClient()
PSĮ 006	Į sistemą įrašomas juridinio kliento vardas	DB procedūros kodas (sp_InsertLegalClient)	sp_InsertLegalClient
PSĮ 007	Į sistemą įrašomas juridinio kliento registracijos numeris	DB procedūros kodas (sp_InsertLegalClient)	sp_InsertLegalClient
PSĮ 008	Į sistemą įrašomas juridinio kliento PVM numeris	DB procedūros kodas (sp_InsertLegalClient)	sp_InsertLegalClient
PSĮ 009	Į sistemą įrašyti juridinio kliento duomenys	Dalykinės programos kodas (SaveNewLegalClientCompleted())	SaveNewLegalClient Completed()
PSĮ 101	Į sistemą ketinama įrašyti naują prekę	Naudotojo sąsaja (cMainMenuInsertItem)	InsertItemClick()
PSĮ 102	Į sistemą įrašoma nauja prekė	Naudotojo sąsaja (cInsertNewItemButton)	InsertNewItem()
PSĮ 103	Prekė yra įrašyta į sistemą	DB operacija (Įrašo įkėlimas į lentelę *Items*)	DB trigeris tr_Items_Insert
PSĮ 201	Įrašyta užsakyta prekė nėra atsiimama	Naudotojo sąsaja („Del“ mygtukas)	DeleteOrder()
PSĮ 301	Ruošiamasi parduoti prekę	Naudotojo sąsaja (cMainMenuNewOrder)	NewOrderClick()
PSĮ 401	Prekė yra pas tiekėją	Naudotojo sąsaja (cSaveOrderButton)	SaveOrder()
PSĮ 402	Sukuriamas naujas užsakymas	DB operacija (Įrašo įkėlimas į lentelę *Orders*)	DB trigeris tr_Orders_Insert
PSĮ 403	Klientas nepažymėtas kaip patikimas	DB trigeris (tr_Orders_Insert)	SaveNewOrder Completed()
PSĮ 501	Gautas pranešimas kad prekė neatvyko	Naudotojo sąsaja („Cancel“ mygtukas)	CancelOrder()
PSĮ 502	Pašalintas prekės užsakymas	DB procedūra (sp_CancelOrder)	CancelOrder Completed()
PSĮ 601	Yra išrenkama parduodama prekė	Naudotojo sąsaja (cSearchItemButton)	SearchItemButton Clicked ()
PSĮ 602	Patvirtinamas prekės pardavimas	Naudotojo sąsaja (cSellItems)	SellItem()
PSĮ 603	Gauti čekio ir sąskaitos duomenys	Naudotojo sąsaja (cConfirmSellButton)	ConfirmSellClicked()
PSĮ 604	Prekė yra įrašyta į parduotų sąrašą	DB procedūra (sp_SellItem)	SaveItemSell Completed()
PSĮ 605	Sukurta garantinė pažyma	Dalykinės programos kodas (CreateWarranty())	SaveWarranty()
PSĮ 701	Ketinama gauti užsakymų sąrašą	Naudotojo sąsaja (cMainMenuOrdersList)	OrdersListClick()

Lentelėse 2 ir 3 specifikuoti sistemos modelio įvykiai. Specifikavus įvykius galima susieti juos su jais inicijuojamomis taisyklėmis. Pradžioje reikia susieti informacinės sistemos įvykius ir taisykles. Tam galima sukurti bendrą matricą. Nagrinėjamu atvejų susiejimo taškai yra išdėstyti netoli įstrižainės, todėl didelę matricą galima suskaidyti į, šiuo atveju, tris mažesnes matricas (4 lent., 5 lent. ir 6 lent.)

4 lentele. ***Informacinės sistemos taisyklių ir įvykių susiejimo matrica 1***

Taisyklės / Įvykiai	IS 1.1.1	IS 1.1.2	IS 1.1.3	IS 1.1.4	IS 1.1.5	IS 1.1.6	IS 1.1.7
ISĮ 001		X	X				
ISĮ 002	X	X					
ISĮ 003	X		X	X	X	X	
ISĮ 004							X

5 lentele. ***Informacinės sistemos taisyklių ir įvykių susiejimo matrica 2***

Taisyklės / Įvykiai	IS 2.1.1	IS 2.2.1	IS 3.1.1	IS 3.11.1	IS 3.3.1	IS 3.4.1	IS 3.4.2	IS 3.4.3	IS 3.5.1	IS 3.6.1
ISĮ 101	X									
ISĮ 102		X								
ISĮ 201			X							
ISĮ 301				X						
ISĮ 401					X					
ISĮ 402						X	X	X		
ISĮ 403									X	
ISĮ 404										X

6 lentele. ***Informacinės sistemos taisyklių ir įvykių susiejimo matrica 3***

Taisyklės / Įvykiai	IS 3.8.1	IS 3.2.1	IS 3.9.1	IS 3.10.1	IS 3.10.2	IS 3.7.1
ISĮ 501	X					
ISĮ 601		X				
ISĮ 602			X			
ISĮ 603				X		
ISĮ 604					X	
ISĮ 701						X

Žinodami kokiems informacinės sistemos įvykiams atitinka kokie programų sistemos įvykiai (2 lent.), pasitelkus lenteles 4,5 ir 6 galime susieti informacinės sistemos taisykles su programų sistemų įvykiais. Jų susiejimo taškuose bus programų sistemų procesai, kuriuose yra realizuotos IS taisyklės, inicijuojami PS įvykiais. Į atitinkamus lentelių langelius galimą įrašinėti programų sistemų procesus, bet galima iškarto įrašyti tose procesuose realizuotas taisykles (7. lent., 8 lent. ir 9 lent.).

7 lentele. ***Informacinės sistemos taisyklių ir programų sistemų įvykių susiejimo matrica 1***

Taisyklės / Įvykiai	IS 1.1.1	IS 1.1.2	IS 1.1.3	IS 1.1.4	IS 1.1.5	IS 1.1.6	IS 1.1.7
PSĮ 001		PS 1.1.2.1	PS 1.1.3.1				
PSĮ 002		PS 1.1.2.2					
PSĮ 003			PS 1.1.3.2				
PSĮ 004	PS 1.1.1.1	PS 1.1.3.3					
PSĮ 005	PS 1.1.1.2		PS 1.1.2.3				
PSĮ 006				PS 1.1.4.1			
PSĮ 007				PS 1.1.4.2	PS 1.1.5.1		
PSĮ 008				PS 1.1.4.3		PS 1.1.6.1	
PSĮ 009							PS 1.1.7.1

8 lentele. ***Informacinės sistemos taisyklių ir programų sistemų įvykių susiejimo matrica 2***

Taisyklės / Įvykiai	IS 2.1.1	IS 2.2.1	IS 3.1.1	IS 3.11.1	IS 3.3.1	IS 3.4.1	IS 3.4.2	IS 3.4.3	IS 3.5.1	IS 3.6.1
PSĮ 101	PS 2.1.1.1									
PSĮ 102	PS 2.1.1.2									
PSĮ 103	PS 2.1.1.3									
PSĮ 104		PS 2.2.1.1								
PSĮ 201			PS 3.1.1.1							
PSĮ 301				PS 3.11.1.1 3.11.1.2						

PSĮ 401						**PS 3.4.1.1**	**PS 3.4.2.1**	**PS 3.4.3.1**		
PSĮ 402									**PS 3.5.1.1**	
PSĮ 403										**PS 3.6.1.1**

9 lentele. ***Informacinės sistemos taisyklių ir programų sistemų įvykių susiejimo matrica 3***

Taisyklės / Įvykiai	**IS 3.8.1**	**IS 3.2.1**	**IS 3.9.1**	**IS 3.10.1**	**IS 3.10.2**	**IS 3.7.1**
PSĮ 501	**PS 3.8.1.1**					
PSĮ 502	**PS 3.8.1.2**					
PSĮ 601		**PS 3.2.1.1**				
PSĮ 602			**PS 3.9.1.1**			
PSĮ 603				**PS 3.10.1.1**		
PSĮ 604					**PS 3.10.1.1**	
PSĮ 605					**PS 3.10.1.2**	
PSĮ 701						**PS 3.7.1. 1**

Sudarius įvykių modelį ir susiejus jį su programų sistemos procesais galima tiksliai nusakyti kaip programų sistemoje realizuojamos verslo ir informacinės sistemos taisyklės (10 lent.). Geresniam suvokimui programų sistemų taisykles galima klasifikuoti priklausomai nuo programų sistemos ypatumų. Mūsų atveju PS taisyklės yra klasifikuojamos į: struktūros taisykles – taisyklės realizuojamos per duomenų bazės lentelių ribojimus; duomenų bazės kodo taisykles – taisyklės aprašytos duomenų bazės procedūrų, trigerių ir funkcijų kode; duomenų bazės trigerius – taisyklės realizuojamus per duomenų bazės trigerius inicijuojamus duomenų pasikeitimu; programos kodo taisykles – taisyklės realizuojamos pačios programos kode (ar tai būtų kreipiniai į vidines programos funkcijos, ar į kokius išorinius servisus).

10 lentele. ***Verslo, informacinės ir programų sistemų taisyklių sistemos modelis***

VS taisyklės	IS taisyklės	PS taisyklės			
		Struktūra	**DB kodas**	**Trigeris**	**Programos kodas**
1.1 Jeigu klientas pageidauja, jis gali užsiregistruoti	1.1.1 Klientas negali būti įkeltas į sistemą daugiau negu vieną kartą				1.1.1.1 1.1.1.2
	1.1.2 Jeigu klientas yra fizinis asmuo, į sistemą turi būti įkelti kliento vardas, pavardė ir papildoma informacija.				1.1.2.1 1.1.2.2 1.1.2.3
	1.1.3 Jeigu klientas yra juridinis asmuo, į sistemą turi būti įkelti kliento įmonės pavadinimas, įmonės kodas, įmonės PVM mokėtojo kodas.				1.1.3.1 1.1.3.2 1.1.3.3
	1.1.4 Juridinio kliento vardas, įmonės kodas ir PVM kodas turi būti unikalus.		1.1.4.1 1.1.4.2 1.1.4.3		
	1.1.5 Juridinio kliento registracijos numeris turi būti 9 simbolių ilgio	1.1.5.1	1.1.5.2		
	1.1.6 Juridinio kliento PVM kodas turi būti 11 simbolių ilgio.	1.1.6.1	1.1.6.2		
	1.1.7 Juridinio kliento duomenys turi būti patikrinti įmonių registre.				1.1.7.1
2.1 Gavus prekę iš tiekėjo, reikia užrašyti jos duomenis ir padėti ją į sandėlį.	2.1.1 Į sistemą turi būti įkelti tiekėjo duomenys, prekės pavadinimas, gamintojas, modelis, serijinis numeris ir kaina.		2.1.1.2	2.1.1.3	2.1.1.1
2.2 Jeigu prekė yra užsakyta, tai ją reikia atidėti.	2.2.1 Jeigu prekė yra užsakyta, ji turi būti pažymėta kaip rezervuota.		2.2.1.1		
3.1 Pirkėjui parduodama prekė turi būti sandėlyje	3.1.1 Parduodama prekė turi būti sandėlyje.				3.1.1.1
3.2 Pirkėjui parduodama prekė negali būti rezervuota.	3.2.1 Parduodama prekė neturi būti rezervuota.				3.2.1.1
3.3 Užsakoma prekė turi buti pas tiekėją.	3.3.1 Turi būti patikrintas prekės buvimas pas tiekėją.	-	-	-	-
3.4 Kliento pageidavimu reikia užsakyti prekę.	3.4.1 Reikia sukurti prekės užsakymą.				3.4.1.1
	3.4.2 Reikia įtraukti prekę į užsakomų prekių sąrašą.				3.4.2.1
	3.4.3 Turi būti įrašyta informaciją apie planuojamą atvykimą				3.4.3.1

3.5 Padarius užsakymą, reikia patikrinti kliento patikimumą	3.5.1 Sukūrus užsakymą, reikia patikrinti ar klientas yra patikimas			3.5.1.1	
3.6 Jeigu klientas nėra „patikimas“, reikia paimti iš jo avansą	3.6.1. Jeigu klientas nėra „patikimas,“ turi būti paimtas avansas		3.6.1.1		
3.7 Turi būti galimybė pamatyti užsakymų sąrašą	3.7.1. Turi būti galimybė pamatyti užsakymų sąrašą				3.7.1.1
3.8 Jeigu prekė neatvyko, reikia panaikinti jos užsakymą.	3.8.1. Negavus prekės, reikia pašalinti ją iš užsakomų sąrašo.			3.8.1.2	3.8.1.1
3.9 Čekio ir/ar sąskaitos duomenys turi būti įrašyti	3.9.1 Čekio ir/ar sąskaitos duomenys tūri būti įrašyti.				3.9.1.1
3.10 Perdavus prekę klientui, reikia išrašyti garantiją.	3.10.1. Atspausdinti garantinę pažymą				3.10.1.1
	3.10.2. Užrašyti garantinės pažymos informaciją.				3.10.1.2
3.11 Jeigu klientas neatsiima užsakytos prekės, jis tampa nepatikimu	3.11.1 Jeigu klientas neatsiima užsakytos prekės jis tampa nepatikimu			3.11.1.2	3.11.1.1

5. Eksperimentinis metodo patikrinimas

Šio darbo 4 skyriuje aprašytu metodu buvo sukurtas informacinės sistemos prototipas. 4 skyriuje aprašytas karkasas buvo naudojamas kaip specifikacija prototipo kūrimui. Tai leido sudaryti veikiančią pagal VS taisykles sistemą ir praktiškai išvengti perteklinio funkcionalumo.

Prototipo kūrimui buvo parinkta tinklinės programos (angl. *web application*) koncepcija. Tokį pasirinkimą lėmė darbe paminėtas vis didėjantis šiuolaikinio verslo judrumas (angl. *agility*). Dažnai atsiranda toks poreikis, kada ketinama dirbti su programa keliuose skirtinguose darbo vietose, kas yra nepatogu kai sistema yra įdiegta lokaliai kurioje nors vienoje darbo vietoje. Įdiegus programą keliuose vietose gali atsirasti sunkumai su vienodų programos versijų užtikrinimu. Tuo tarpu tinklinės programos gali būti naudojamos praktiškai iš bet kurios darbo vietos pasitelkus lengvai prieinamas priemones (pvz. naršyklę). Tinklinių programų atveju darbas yra vykdomas tik su vienintele programos versija, kas leidžia laisvai daryti įvairius pakeitimus programų sistemoje žinant, kad tai pasieks visus naudotojus.

Kuriant IS prototipą buvo sukurta reliacinė duomenų bazė naudojant *Microsoft SQL Server 2005* duomenų bazių valdymo sistemą. IS prototipas buvo kuriamas *Microsoft Visual Web Developer 2005 Express Edition* kūrėjo terpėje (angl. *develpoer environment*). Duomenų bazės struktūra, duomenų bazės kodas ir programos kodas yra pateikti darbo prieduose. Taip pat. diagramų pagalba prieduose yra pateikta sukurto prototipo dalykinės srities ir duomenų struktūros analizė.

Kaip prototipo veikimo pavyzdys darbe pateikiamas sukurtas prekės užsakymo šalinimas kada klientas neatsiima užsakytos prekės. Darbas programoje prasideda pagrindiniu puslapiu (25 pav.).

25 pav. ***Pagrindinis prototipo puslapis***

Užsakymo šalinimo procesas susideda iš dviejų smulkesnių procesų: užsakymų sąrašo peržiūra ir užsakymo šalinimas. Tarkime verslo sistemoje inicijuojamas įvykis *VSĮ 701: Ketinama peržiūrėti užsakymus,* šiam VS įvykiui atitinka programų sistemos įvykis *PSĮ 701: Ketinama gauti užsakymų sąrašą,* kuris yra inicijuojamas naudotojo aplinkos elemento *cMainMenuOrdersList* aktyvavimu, paprastai tariant pagrindinio meniu punkto „Užsakymai“ paspaudimu (26 pav.).

Verslo sistemos įvykiui *VSĮ 201: Klientas neatsiima prekės* atitinka naudotojo aplinkos įvykis PSĮ 201: „Del“ mygtuko paspaudimas, kuris inicijuoja užsakymo šalinimą iš duomenų bazės (programų sistemos taisyklė 3.11.1.1) ir kliento pervedimo į nepatikimų klientų aibę (programų sistemos taisyklė 3.11.1.2) procesus.

27 pav. ***Užsakymo pašalinimo išrinkimas***

Rezultatą, kaip pašalintą užsakymą, galima matyti 28 pav.

28 pav. ***Užsakymų sąrašas po vieno iš užsakymų pašalinimo operacijos***

Šis demonstravimas parodė, kad sukūrus sistema remiantis pasiūlytų karkasu galima nesunkiai sužinoti kokios taisyklės yra atkyvojamos tam tikrais įvykiais ir kaip tos taisyklės yra realizuotos sukurtoje sistemoje.

IŠVADOS

Informacinėms sistemoms, kurios geriausiai galėtų patenkinti verslo poreikius, kurti yra naudojamos verslo taisyklės. Bet siekiant efektyvesnio darbo, reikia nepamiršti įvykių sistemos ir jos realizavimo.

1. Literatūroje įvykiai aprašomi ir klasifikuojami beveik vienodai. Aišku, yra išimčių, tačiau jos dažniausiai papildo standartinį apibrėžimą ar klasifikavimą. Dažniausiai straipsniuose ir publikacijose, kai kalbama apie įvykių klasifikavimą, turima omenyje tik programų sistemos įvykiai. Mokslinių darbų analizė parodė, kad yra vykdoma nemažai tyrimų pasirinktos temos srityje ir yra pakankamai literatūros esminėms bendroms darbo sąvokoms aprašyti. Tačiau darbo tema aktyviau moksliniai darbai pradėjo vykti tik palyginamai neseniai. Nors dauguma terminų nėra griežtai formalizuoti, jų apibrėžimai yra nusistovėję ir daugumoje šaltinių yra vienodi.

2. Šiuo metu įvykiais valdomos architektūros aspektus stengiamasi pritaikyti vis didesniame verslo sistemų kiekyje. Tai leis pagreitinti ir „pajudrinti“ (angl. *agile*) sistemoje vykstančius procesus. Procesų judrumas iš esmės keičia procesus, kad šie atitiktų pasaulio pokyčius, – pavyzdžiui, automobilių gaminimo linijos pertvarkymas gaminti naujus, atitinkančius rinkos poreikius, kėbulus vietoj prieš tai gamintų. Maksimalus judrumo lygis būtų reaguoti į įvykius bet kuriuo laiku, netgi proceso vykdymo metu. Įvykiais valdomų informacinių sistemų koncepcija yra vis dažniau naudojama kuriant informacines sistemas, nes sistemų užsakovai vis labiau stengiasi sumažinti informacinių sistemų kainą ir pagerinti sistemų reagavimo į vartotojų užklausas savybę.

3. Įvykiai dažniausiai yra vaizduojami dinaminiuose sistemų modeliuose, ten kur reikia parodyti būsenų keitimąsi ar perėjimą, procesų sekų inicijavimą arba taisyklių aktyvavimą. Kadangi taisyklės yra inicijuojamos įvykiais ir vykdomos per procesus galima apjungti taisyklių, įvykių ir procesų modelius į vieną bendrą karkasą, kur eilutėse būtų atvaizduoti verslo, informacinės ir programų sistemų lygmenys, o stulpeliuose – taisyklės, įvykiai ir procesai. Tokio karkaso panaudojimo dėka galima sukuti programų sistemą, kuri pilnai atitinka verslo taisykles. O sistemos kūrimo metu yra sukuriamas taisyklių, įvykių ir procesų modelis, kuris leis, pasikeitus taisyklėms ar jas inicijuojantiems įvykiams, nesunkiai gerinti ar kitaip keisti jau sukurtą sistemą.

4. Informacinės sistemos prototipo kūrimo metu buvo aiškiai pastebėtas ryšys tarp verslo, informacinės ir programų sistemų taisyklių ir jas inicijuojančių įvykių. Įvykiai, kaip ir taisyklės, yra transformuojami nuo verslo sistemų lygmens iki programų sistemų. Šitas aspektas leidžia atsekti informacinės sistemos realizavimo ypatybių šaltinius verslo sistemoje ir leidžia esant

kokiems nors pasikeitimams koncentruoti dėmesį tik reikiamai sistemos daliai, o ne visai sistemai.

5. Pasiūlyto karkaso panaudojamas leido sukurti IS prototipą tiksliai pagal suformuluotas kūrimo pradžioje verslo ir informacinio lygmenų taisykles, o kūrimo pabaigoje tiksliai išskirti joms (verslo ir informacinės sistemų taisyklėms) atitinkančias programų sistemų taisykles ir jas inicijuojančius įvykius. Tokio vieningo karkaso dėka galima, pasikeitus verslo taisyklei, nesunkiai atsekti kokios PS taisyklės reikalauja pakeitimo.

Šiais laikais vis plačiau taikomos įvykiais valdomos sistemos. Tai vyksta ne tik dėl padidėjusio poreikio, bet ir dėl vis dažniau pasitaikančių galimybių jas pritaikyti verslo sistemose. Yra pakankamas kiekis literatūros, skirtos atskiroms įvykiais valdomų sistemų dalims, nes atskirai jos buvo naudojamos ir realizuojamos jau pakankamą laiko tarpą. Tačiau įvykiais valdomoms sistemoms kaip atskirai sąvokai skirti straipsniai ir kita literatūra kuriami palyginti neseniai.

Yra didelė erdvė vystyti įvykių sistemos bei įvykiais valdomų sistemų kaip vientisos sistemos taip ir daugelio posistemių srityje mokslinį ir inžinerinį darbą. Tolimesnio darbo pavyzdžių gali būti darbe pasiūlyto karkaso kompiuterizavimas ir automatizavimas bei įvykių valdymo sistemos išsamus tyrimas.

TERMINŲ IR SANTRUMPŲ ŽODYNAS

ADBVS	–	aktyvi duomenų bazių valdymo sistema .
Aktyvi duomenų bazių valdymo sistema	–	tai duomenų bazių valdymo sistema, kuri automatiškai gali reaguoti į veiksmus atliekamus su duomenų bazėse saugomais duomenimis.
Business Rules Community	–	verslo taisyklių problematika užsiimanti organizacija.
DB	–	duomenų bazė.
DBS	–	duomenų bazių sistema.
DBVS	–	duomenų bazių valdymo sistema .
DBVS transakcija	–	tai loginis darbo su duomenimis vienetas.
Duomenų bazė	–	tai tarpusavy susijusių duomenų rinkinys. Duomenys gali būti įvairūs: tekstai, paveikslai, garsai. Juos tvarko duomenų bazės valdymo sistema (DBVS).
Duomenų bazių valdymo sistema	–	tai programų sistema, skirta patikimai ir veiksmingai kurti ir valdyti dideles, integruotas ir daugelio vartotojų duomenų bazes.
ECA (Event-Condition-Action)	–	taisyklė įvykis-sąlyga-veiksmas angl.
Gyvavimo laikas (angl. *time-to-live*)	–	laikas per kuri egzistuoja arba yra aktualus tam tikras objektas.
Informacinė organizacijos sistema	–	tai organizacinio viene-to informacijos saugojimo, paieškos, perdavimo ir apdorojimo sistema.
IS	–	informacinė sistema.
Įvykis	–	atsitikimas tam tikru laiko momentu dominančioje vietoje.

Įvykių sistema	–	tarpusavyje susietų įvykių visuma.
Kompiuterizuota informacinė sistema	–	jeigu bent vienas informacinės sistemos posistemis yra kompiuterizuotas, tai sistema yra vadinama kompiuterizuota.
OMG (Object Management Group)	–	organizacija tvarkanti ir kurianti objektinio kūrimo standartus.
Procesas	–	veiksmų seka skirta tikslui pasiekti.
Programų sistema	–	tai integruota programų, rinkmenų, duomenų bazių ir žinių bazių visuma, skirta tam tikros klasės uždaviniams spręsti arba tam tikriems įrenginiams ar procesams valdyti.
PS	–	programų sistema.
Servisas	–	tarnybinė programa, kuri gali automatiškai pasileidinėti ir veikti kaip sistemos dalis.
SOAP (Simple Object Access Protocol)	–	informacijos apsikeitimo protokolas
The Business Rules Group	–	verslo taisyklių problematika užsiimanti organizacija.
Transakcija	–	veiksmų su duomenų bazių seka, kurioje kiekvienas veiksmas turi pasibaigti sėkmingai, kad transakcija būtų laikoma sėkminga
Trigeris	–	tai tam tikras saugomos procedūros tipas, kuri pradedama vykdyti, kai INSERT, UPDATE, DELETE sakiniais keičiamas su trigeriu susietos lentelės turinys arba keičiasi DB būsena.
UML (angl. *Unified Modeling Language*)	–	unifikuota modeliavimo kalba.
Verslo taisyklė (angl. *business rule*)	–	formali procedūra arba metodika, konkrečioje situacijoje atliekanti tipišką valdymo sprendimą.
Verslo taisyklės	–	tai visuma visų taisyklių, kurios reglamentuoja veiklos vykdymą

(angl. *business rules*)		(įstatymai, politika, vidaus tvarka ir t.t.). Šis taisyklių rinkinys apima statinius (struktūrinius) ir dinaminius (funkcinius, elgsenos) organizacijos veiklos aspektus.
VS	–	verslo sistema.
VT	–	verslo taisyklė.
WSDL (Web Services Description Language)	–	servisų aprašymo kalba.
Zachmano karkasas	–	formalus ir griežtai struktuirzuotas sistemų aprašymo budas atvaizduojamas specialioje lentelėje (karkase).

LITERATŪRA

[1] Bednarcyk B. (IBM), Dasan V. (Sun Microsystems), Grose A. (Microsoft), Jagasia J. (Microsoft) *Information Technology Standards.* Architecture Advisory Board, Governor's Office of Innovation and Technology, State of Colorado, 2004. www.colorado.gov/oit/documents/standards/State_IT_Standards_072004_v1.3.pdf [žiūrėta 2006.08.22].

[2] Berndtson M., Calestam B. *A Uniform Approach for Supporting Active Database Features in UML and OMT.* Department of Computer Sciences, University of Skövde, Skövde, Sweden. 1999. http://citeseer.ist.psu.edu/cache/papers/cs/22559/http:zSzzSzwww.ida.his.sezSzidazSzadczSz..zSzresearchzSztech_reportszSzreportszSztr01zSzHS-IDA-TR-01-003.pdf/a-uniform-approach-for.pdf [žiūrėta 2006.08.22].

[3] Booch G. *OMG Unified Modeling Language Specification*. Rational Software Corporation. Language Specification, March 2000. http://www.omg.org/uml [žiūrėta 2006.08.22].

[4] Business Rules Community. 2006: www.brcommunity.com.

[5] Caplinskas A., Lupeikiene A., Vasilecas O.. A Framework to Analyse and Evaluate Information Systems Specification Languages. *In Y. Manolopoulos, P. Navrat (eds.). Advances in Databases and Information Systems, ADBIS 2002, LNCS 2435, Springer.* September 2002, pp. 248-262.

[6] Caroli P., José P., de Lucena C., Fontoura M. (1999) *An Architecture for the Evolution of Web Applications.* (OOPSLA'2000), Minneapolis, USA, 1999, pp. 83-84. http://www.almaden.ibm.com/cs/people/fontoura/papers/poster2000.pdf [žiūrėta 2006.08.22].

[7] Chakravarthy S., Krishnaprasad V., Anwar E.; Kim S.-K. (1994) *Composite Events for Active Databases: Semantics, Contexts and Detection.* 12 p., Database Systems Research and Development Center, Computer and Information Sciences Department, University of Florida, Gainesville. The Twentieth International Conference on Very Large Data Bases, Santiago, Chile, September 1994, pp. 606-617. www.vldb.org/conf/1994/P606.pdf [žiūrėta 2006.08.22].

[8] Chakravarthy S., Varkala S. Dynamic programming environment for active rules. *In O. Vasilecas et al. (eds.), Proceedings of the 2006 Seventh International Baltic Conference on Databases and Information Systems*, *Vilnius, Technika,* 2006, pp. 3-16.

[9] Cilia M. An Active Functionality Service for Open Distributed Heterogeneous Environments. *Ph.D. Dissertation, Department of Computer Science, Darmstadt University of Technology.* 2002. www.dvs1.informatik.tu-darmstadt.de/publications/pdf/active-functionality.pdf [žiūrėta 2006.08.22].

[10] Cilia M., Bornhövd C., Buchmann A. P.. Event Handling for the Universal Enterprise. *Information Technology and Management, Kluwer Academic Publishers.* January 2005, Vol. 6, 1, pp. 123-148. http://www.springerlink.com/content/gu6816mp287l0214/ [žiūrėta 2006.08.22].

[11] Cousins J., Stewart T. *What is Business Process Design and Why Should I Care?* RivCom Ltd. 2002. www.rivcom.com/resources/RivCom WhatIsBPD WhyShouldICare.pdf [žiūrėta 2006.08.22].

[12] Date C. J. Twelve Rules for Business Rules. 2000. www.alphora.com/12Rules-Date.pdf [žiūrėta 2006.08.22].

[13] Fiege L., Mühl G., Gärtner F. C. Modular Event-Based Systems. *The Knowledge Engineering Review, Cambridge University Press.* 2002, Vol. 17, 4, pp. 359-388. http://citeseer.ist.psu.edu/728444.html [žiūrėta 2006.08.22].

[14] Gregoriades A., Karakostas V. *A Simulation Methodology Unifying System Dynamics and Business Objects*. Dept. of Computation, UMIST University, Manchester. XIX International Conference, 2001. www.systemdynamics.org/conf2000/PDFs/gregoria.pdf [žiūrėta 2006.08.22].

[15] Hay D., Healy K. A., The Business Rules Group. Defining Business Rules ~. What Are They Really? *Technical Report 1.3, The Business Rules Group*. July 2000. http://www.businessrulesgroup.org/first_paper/BRG-whatisBR_3ed.pdf [žiūrėta 2006.08.22].

[16] Heimrich T., Specht G. Enhancing ECA Rules for Distributed Active Database Systems. *Web Information Systems – WISE 2004, Springer Berlin / Heidelberg*. 2003, pp. 199-205. http://www3.tu-ilmenau.de/fakia/fileadmin/template/FakIA/Strukt-Fakultaet_IA/ipim/dbis/heimrich/2002-LNCS-Enhancing-ECA.pdf [žiūrėta 2006.08.22].

[17] IBM Corporation. *Business Rule Beans (BRBeans),* 2001. http://publib.boulder.ibm.com/infocenter/wasinfo/v4r0/topic/com.ibm.websphere.v4.doc/wasee_content/pdf/eebrb.pdf [žiūrėta 2006.08.22].

[18] Le Dinh T. Information system upon information systems: A conceptual framework. *Ph.D. Thesis, Geneva University, Geneva.* 2004. http://www.unige.ch/cyberdocuments/theses2004/LeDinhT/these.pdf [žiūrėta 2006.08.22].

[19] Le Roger. *Support for Composite Events and Rules in Distributed Heterogeneous Environments*. The University Of Florida. 1998. http://itlab.uta.edu/sharma/People/ThesisWeb/leroger_thesis.pdf [žiūrėta 2006.08.22]

[20] Limeric University. *Information Modeling*. Department of Computer Science and Information Systems, University of Limeric. http://sky.fit.qut.edu.au/~edmond/im/ [žiūrėta 2006.08.22].

[21] Mehran N. *Service-Oriented Architecture, Where do we stand?* Dunstan Thomas Consulting. www.dthomas.co.uk/src/requestDoc.asp?DocumentID=57&Category=Consulting%20Articles [žiūrėta 2006.08.22].

[22] Meier R., Cahill V. STEAM: Event-Based Middleware for Wireless Ad Hoc Networks. *Proceedings of the International Workshop on Distributed Event-Based Systems (ICDCS/DEBS'02).* 2002, pp. 639-644. http://doi.ieeecomputersociety.org/10.1109/ICDCSW.2002.1030841 [žiūrėta 2006.08.22].

[23] Michiels C., Snoeck M., Lemahieu W., Goethals F., Dedene G. A Layered Architecture Sustaining Model Driven and Event Driven Software Development. *Conceptual Modeling-ER2000, 19th International Conference on Conceptual Modeling, Springer.* October 2000, pp. 454-469. www.econ.kuleuven.ac.be/public/NDBAA30/PSI-Michiels.pdf [žiūrėta 2006.08.22].

[24] Nickull D. *Service Oriented Architecture Whitepaper.* Adobe Systems Incorporated. 2005. www.adobe.com/enterprise/pdfs/Services_Oriented_Architecture_from_Adobe.pdf [žiūrėta 2006.08.22].

[25] Noran O. S. *Business Modeling: UML vs. IDEF*. Griffith University. School of Computing and Information Technology. 2000. www.cit.gu.edu.au/~noran/Docs/UMLvsIDEF.pdf [žiūrėta 2006.08.22].

[26] Object Management Group. *Business Process Definition Metamodel*. 2003. http://www.bpmn.org/Documents/BPDM/OMG-BPD-2004-01-12-Revision.pdf [žiūrėta 2006.08.22].

[27] Object Management Group. 2006: www.omg.org

[28] Object Management Group. Semantics of Business Vocabulary and Business Rules Specification. 2006.

[29] Schark Research Group. *Descrete Event Driven Simulator Manual.* 2000. http://citeseer.ist.psu.edu/cache/papers/cs/16155/http:zSzzSzwww.ics.uci.eduzSz~scharkzSzsimulatorzSzManual.pdf/discrete-event-driven-simulator.pdf [žiūrėta 2006.08.22].

[30] Schulte R. Using Events for Business Benefit. *Business integration journal.* May 2004, 43-45 p.

[31] SPARX System. *The Business Process Model*. Sparx Systems UML Tutorials. 2004. www.sparxsystems.com/downloads/whitepapers/The_Business_Process_Model.pdf

[32] Speidel O. A Rule-Based Approach to Monitoring Moving Objects. *Universität Bonn, Institut für Informatik III.* http://citeseer.ist.psu.edu/650274.html [žiūrėta 2007.03.18].

[33] The Business Rules Group. 2006: www.businessrulesgroup.org .

[34] Vaduva A. Rule Development for Active Database Systems. Dissertation, der Wirtschaftswissenschaftlichen Fakultat der Universitat, Zürich. 1999. http://www.info.cu.edu.tr/Content/Data/RuleDevelopmentforActiveDatabaseSystems.pdf [žiūrėta 2007.03.18].

[35] Valatkaite I., Vasilecas O. On Business Rules Automation: the BR-centric IS Development Framework. *In J. Eder et al. (Eds.): ADBIS 2005, Springer.* 2005, LNCS 3631, pp. 350 – 365.

[36] Vasilecas O., Bugaite D. On Approach of Events Modelling in the Process of Business Rules-Based Information Systems Development. *In O. Vasilecas, A. Caplinskas, J. Eder (eds.), Databases and Information Systems. Communications. Materials of Doctoral Consortium (Baltic DB&IS 2006), Vilnius, Technika*. 2006, pp. 15-25.

[37] Vasilecas O., Smaizys A. The Framework: an Approach to Support Business Rule Based Data Analysis. *In O. Vasilecas et al. (eds.), Proceedings of the 2006 Seventh International Baltic Conference on Databases and Information Systems*, *Vilnius, Technika,* 2006, pp. 141-147.

[38] Weigand H., de Moor A. Linking event-driven and communication-oriented business modeling. *The 10th International Working Conference on the Language Action Perspective on Communication Modeling (LAP 2005), STARLab, Vrije Universiteit Bruseel, Belgium*. June 19-20 2005, pp.107-118. www.vits.org/konferenser/lap2005/Paper%208-LAP.pdf [žiūrėta 2007.03.18].

[39] WordNeta. Lexical database for the English language. *Princeton University*. http://wordnet.princeton.edu/ [žiūrėta 2005.12.20].

[40] Zoumboulakis M., Roussos G., Poulovassilis A. Active Rules for Sensor Databases. *DMSN '04: Proceeedings of the 1st international workshop on Data management for sensor networks, ACM Press.* 2004, Vol. 72, pp. 98 – 103. http://portal.acm.org/citation.cfm?id=1052215&dl=ACM&coll=&CFID=15151515&CFTOKEN=6184618 [žiūrėta 2007.03.18].