



BENDRASIS PROGRAMAVIMO DOKUMENTAS



EUROPOS SĄJUNGA
Europos socialinis fondas



ŠVIETIMO IR MOKSLO MINISTERIJA



KURKIME ATEITĮ DRAUGE!

Albertas Čaplinskas

REIKALAVIMŲ INŽINIERIJA

Mokymo medžiaga

**Vilnius
2010**

Turinys

1	Ižanga.....	5
1.1	Apie ką ši knyga?.....	5
1.2	Kam skirta ši knyga?.....	6
1.3	Kuo ypatinga ši knyga?.....	8
2	Reikalavimų rūšys	12
2.1	Reikalavimų lygmenys	12
2.2	Produkto reikalavimai ir projekto reikalavimai	18
2.3	Funkciniai ir nefunkciniai produkto reikalavimai	19
2.4	Verslo reikalavimai	22
	2.4.1 Verslo reikalavimų formulavimo ypatumai	22
	2.4.2 Kodėl?	22
	2.4.3 Kaip?	42
	2.4.4 Ką?	48
	2.4.5 Kas?.....	49
	2.4.6 Kur?.....	50
	2.4.7 Kada?	51
	2.4.8 Baigiamosios pastabos	53
2.5	Vartotojo reikalavimai	55
	2.5.1 Vartotojo reikalavimų formulavimo ypatumai	55
	2.5.2 Kodėl?	57
	2.5.3 Kaip?	59
	2.5.4 Ką?	61
	2.5.5 Kas?.....	64
	2.5.6 Kur?.....	67
	2.5.7 Kada?	67
	2.5.8 Baigiamosios pastabos	68
2.6	Informacinių sistemų reikalavimai.....	69
	2.6.1 IS reikalavimų formulavimo ypatumai	69
	2.6.2 Kodėl?	73
	2.6.3 Kaip?	79
	2.6.4 Ką?	83
	2.6.5 Kas?.....	84
	2.6.6 Kur?.....	86
	2.6.7 Kada?	87
	2.6.8 Baigiamosios pastabos	87
2.7	Programinės įrangos reikalavimai	87
	2.7.1 Programinės įrangos reikalavimų formulavimo ypatumai.....	87
	2.7.2 Kodėl?	88
	2.7.3 Kaip?	88
	2.7.4 Ką?	95
	2.7.5 Kas?.....	96
	2.7.6 Kur?.....	101
	2.7.7 Kada?	102
	2.7.8 Projektavimo reikalavimai	104
	2.7.9 Realizavimo reikalavimai	104
	2.7.10 Baigiamosios pastabos	104
2.8	Reikalavimų svarba projekto sėkmėi.....	105
3	Reikalavimų inžinerijos procesas.....	107

3.1	Reikalavimų inžinerijos proceso samprata	107
3.2	Reikalavimų specifikacijos traktuotės	108
3.3	Reikalavimų inžinerijos proceso sąsajos su kitais programų sistemų inžinerijos procesais.....	110
3.4	Reikalavimų inžinerijos proceso modeliai.....	112
3.4.1	Reikalavimų inžinerijos proceso modelio samprata	112
3.4.2	I.K. Bray reikalavimų inžinerijos proceso modelis	113
3.4.3	Reikalavimų inžinerijos proceso modelis „Volere“	117
3.5	Reikalavimų inžinerijos proceso dalyviai.....	125
3.6	Reikalavimų inžinerijos proceso palaikymas ir vadyba	126
3.7	Reikalavimų inžinerijos proceso kokybė.....	130
3.8	Reikalavimų inžinerijos proceso branda.....	135
3.9	Reikalavimų inžinerijos proceso diegimas ir tobulinimas	139
4	Reikalavimų išsiaiškinimas ir formulavimas.....	143
4.1	Reikalavimų aiškinimosi metodai.....	143
4.2	Reikalavimų šaltiniai	147
4.2.1	Verslo tikslai	147
4.2.2	Dalykinės žinios	148
4.2.3	Šalys, suinteresuotos kuriamą programų sistemą	148
4.2.4	Operacinė kuriamos programų sistemos aplinka	149
4.2.5	Organizacinė verslo sistemos aplinka	149
4.3	Reikalavimų formulavimo būdai	149
4.4	Nefunkcinių reikalavimų formulavimo problemos.....	150
5	Reikalavimų analizė	160
5.1	Reikalavimų analizės esmė	160
5.2	Reikalavimų analizės problemos	162
5.3	Konteksto modeliavimas.....	163
5.4	Reikalavimų konkretizavimas.....	164
5.5	Reikalavimų maketavimas.....	165
5.6	Prieštaravimų radimas ir šalinimas	168
5.7	Skirtingų lygmenų reikalavimų darnos analizė	171
5.8	Reikalavimų įgyvendinamumo analizė.....	172
5.9	Koncepcinis reikalavimų modeliavimas	174
5.10	Duomenų žodyno sudarymas.....	175
5.11	Architektūros parinkimas.....	176
5.12	Reikalavimų prioritetizavimas	177
5.13	Kokybės funkcijų sklaida.....	179
5.14	Reikalavimų lokalizavimas, nuleidimas žemyn ir operacionalizavimas.....	196
5.14.1	Reikalavimų lokalizavimo ir nuleidimas žemyn metodų svarba	196
5.14.2	Reikalavimų lokalizavimo ir nuleidimo žemyn samprata.....	196
5.14.3	Sistemos reikalavimų lokalizavimas ir nuleidimas žemyn	197
6	Reikalavimų dokumentavimas	202
6.1	Reikalavimų specifikavimo problemos	202
6.2	Reikalavimų dokumentavimo standartai	205
6.3	Verslo poreikių specifikacija	207
6.4	Operacinių poreikių specifikacija	209

6.5	Sistemos reikalavimų specifikacijos	212
6.6	Programinės įrangos reikalavimų specifikacija	214
	6.6.1 IEEE 830-1998 standarto nustatyta reikalavimų specifikacijos struktūra	214
	6.6.2 Reikalavimų inžinerijos procese VOLERE nustatyta reikalavimų specifikacijos struktūra	218
7	Reikalavimų tikrinimas ir vertinimas	223
7.1	Reikalavimų tikrinimo ir vertinimo problemos	223
7.2	Reikalavimų tikrinimo ir vertinimo metodai	224
	7.2.1 Reikalavimų darnos analizė	225
	7.2.2 Juodraštinės vartotojui skirtos dokumentacijos rengimas.....	226
	7.2.3 Reikalavimų reformalizavimas ir modelių vertinimas.....	226
	7.2.4 Reikalavimams tikrinti skirtų testų specifikavimas	227
8	Reikalavimų tvarkymas	228
8.1	Reikalavimų tvarkymo esmė	228
8.2	Iteracinis reikalavimų formulavimo proceso pobūdis	229
8.3	Reikalavimų klasifikavimas.....	230
8.4	Reikalavimų matavimo problemos	232
8.5	Reikalavimų atributai.....	234
8.6	Pokyčių valdymas	240
8.7	Reikalavimų trasavimas	242
8.8	Instrumentinės sistemos	247
	Literatūra	250
	Reikalavimų inžinerijos standartai.....	257
	Terminų žodynėlis	260

1 Ižanga

1.1 Apie ką ši knyga?

Prieš pradedant kokį nors darbą, dažnai yra sakoma: "O kaip tiltą statysim? Skersai ar išilgai upės?". Šiame pasakyme slypi gili išmintis. Imantis bet kokio darbo, būtina žinoti, ką gi iš tiesų reikia padaryti. Siuvėjas paprastai klausia kliento ne tik ką, kelnes, apsiaustą ar eilutę, jis nori siūdintis, bet ir įvairių detalių: dvibortę, ar vienabortę eilutę siūti, kiek ir kokių vidinių švarko kišenių daryti, kokį pamušalą dėti, kokias sagas įsiūti ir pan. Kitaip tariant, siuvėjas reikalauja, kad klientas jam pateiktų siuvinio *reikalavimų specifikaciją*. Be abejo, klientas gali nežinoti visų svarbių niuansų, gali ką nors pamiršti ar pražiopsoti. Tačiau, patyręs siuvėjas pats žino, ko reikia papildomai paklausti ir vargu, ar jis ilgai išsaugotų savo klientus, jei, išgirdęs pretenzijas dėl storais drobiniais siūlais susiūto prabangaus smokingo, pareikštų: "O jūs man nesakėte, kokiaisiai siūlais siūti. Kokius turėjau, tokiaisiai ir susiuvaus. Tačiau tai nieko tokio. Išardysiu ir susiūsiu iš naujo. Tiesa teks dar mėnesį palükėti ir, aišku, dar kartą sumokėti man už darbą. Be to ir senų siūlių žymės liks.". Panašiai reikalai klostosi ir kirpykloje. Tik čia dažnai apskritai yra neįmanoma sugadintą šukuoseną kaip nors perdaryti. Tiesa, neišrankus klientas vietoj išsamių reikalavimų gali paprasčiausiai pasakyti: "O aš nežinau. Jūs geriau žinote, kaip mane nukirpti. Noriu tik vieno. Kad būtų gražu.". Tačiau patyręs kirpėjas net gi ir tada pabandys iš kliento iškvosti kokius nors tikslėnius reikalavimus. Pavyzdžiui, pateiks jam madų žurnalą su šukuosenų pavyzdžiais (inžinierius sakyta, pateiks šukuosenos *maketus*) ir paprašys pasirinkti vieną iš jų. Panašiai viskas vyksta ir visose kitose veiklos srityse. Vykdymas (siuvėjas, kirpėjas, architektas ar kas nors kitas), veikdamas kaip *sisteminių analitikas* (jis analizuoją sprendžiamą problemą), kartu su užsakovu kuria reikalavimų specifikaciją arba pateikia jam būsimojo gaminio pavyzdžius ar maketus ir prašo užsakovą juos aprobuoti. Ir juo sudėtingesnis ir brangesnis gaminys yra gaminamas, tuo daugiau dėmesio yra skiriama reikalavimams, nes tuo didesni nuostoliai bus pagaminus ne tai, ko iš tiesų norėjo užsakovas. Paprastais atvejais, pavyzdžiui, kirpykloje, reikalavimų specifikacija gali būti žodinė, bet jau siuvėjas tikriausiai užsirašys užsakovo suformuluotus reikalavimus, o, tarkime, statant namą, tie reikalavimai bus ne tik išdėstyti raštu, bet ir juridiškai įteisinti, pateikiant juos kaip vieną iš sudaromo sandorio dalį. Programų sistemos, aišku, nėra jokia išimtis. Ir tiktais visiškai nieko apie savo specialybę nenutuokiantis programuotojas pradės rašyti programos kodą, neturėdamas išsamios reikalavimų specifikacijos. Žinoma, išskyrus tuos atvejus, kai kuriaama sistema yra tokia paprasta ir pigi, kad jai, panašiai kaip kirpykloje, pakanka žodinės specifikacijos. Beje, programų sistema – ne švarkas ir ką nors pražiopsoti čia yra gerokai lengviau. Be to, programuotojai nebūtinai visą laiką kuria tos pačios paskirties programų sistemas, todėl papildyti užsakovo pateiktą neišsamią specifikaciją jiems gali būti daug sunkiau negu, tarkime, kirpėjui ar siuvėjui. Net ir tada, kada tą specifikaciją jie rengia kartu su užsakovu ar netgi kartu su būsimaisiais sistemos vartotojais. Ką gi daryti? Tam tikra išeitis, be abejo, yra. Ir netgi ne viena.

Visų pirma egzistuoja kolektyvinė patirtis, kurią apibendrina atitinkami *standartai* bei *metodiniai nurodymai*. Ten parašyta, kokios reikalavimų grupės yra būdingos programų sistemoms ir kokius sistemos aspektus reikia aptarti reikalavimų specifikacijoje. Be abejo, ir šitaip galima daug ką pražiopsoti. Galima suformuluoti visas reikalavimų grupes, bet tos grupės gali būti neišsamios. Čia gali padėti būsimos programų sistemos maketai. Vykdymas gali netgi sukaupti, panašiai, kaip šukuosenų

pavyzdžius madų žurnaluose, savo kuriamų sistemų maketu rinkinį ir pademonstruoti tuos maketus užsakovui. Pamatęs būsimos sistemos maketą, o juolab, jeigu tokia galimybė yra, su juo padirbėjės, užsakovas gali daug lengviau pasakyti, kas jam patinka, o kas ne. Panašiai elgiasi ir siuvėjas, kada kviečia klientą pasimatuoti neišbaigtą (tik „sufastriguotą“) eilutę. Be abejo, tokia eilutė yra daug sudėtingesnis maketas, negu nuotrauka žurnale, nes, nors jos ir negalima dėvėti, vis tik ją galima apsivilkti, t. y. atlikti tam tikrą eksperimentą. Taigi, maketai esti įvairūs. Programų sistemoms taip pat galima kurti įvairaus sudėtingumo maketus. Tačiau net ir sudėtingas maketas privalo būti pakankamai pigus (t. y. nesunkiai sukuriamas) ir negali pademonstruoti daugelio svarbių būsimos programų sistemos savybių, pavyzdžiu, jos patikimumo ar darbo greičio. Čia gali padėti užsakovo *poreikių analizė*. Iš tiesų užsakovas nori įsigyti ne kokią nors eilutę, namą, o juolab ne kokią nors programų sistemą. Jis nori patenkinti tam tikrus savo *poreikius*. Pavyzdžiu, jam reikia rūbo darbui sode arba rūbo vestuvėms. Savo poreikius jis gali suformuluoti lengviau negu reikalavimus. Patyręs specialistas gali ir pats iš poreikių suformuluoti reikalavimus ir netgi sukurti atitinkamą maketą. Tačiau sudėtingesniais atvejais užsakovas gali nesuvokti savo poreikių arba suvokti juos klaudingai. Todėl tenka pakilti į dar aukštesnį, *tikslų* lygmenį. Užsakovas siekia tam tikrų tikslų, pavyzdžiu, nori išspręsti tam tikrą verslo problemą. Iš čia išplaukia jo poreikiai, jais remiantis jau galima bandyti formuluoja reikalavimus. Sudėtingais atvejais tikslai gali būti irgi ne iki galo suvokti. Pavyzdžiu, verslas sekasi ne taip, kaip norėtusi, bet niekas tiksliai negali pasakyti kodėl. Yra tam tikros nuomonės ir galima bandyti jas tapatinti su tikslais. Bet čia vėl slypi didžiulė rizika, nes galima pradėti spręsti visai ne tą problemą, kuri iš tiesų lemia verslo nesėkmes ir sukurtoji programų sistema padėties nepagerins. Pinigai bus paprasčiausiai paleisti vėjais. Todėl daug geriau yra pabandyti atlikti objektyvią *verslo analizę*, atskleisti tikrąsias nesėkmų priežastis, suformuluoti realius tikslus ir tik po to pereiti prie poreikių analizės ir sistemos reikalavimų formulavimo. Visas tas veiklas ir dar daug ką kitą apima *reikalavimų inžinerija*. Taigi, kaip matome, tai sudėtinga, labai svarbi ir labai įdomi disciplina. Ji ir yra nagrinėjama šioje knygoje.

Pirmieji darbai, kuriuose akcentuojama reikalavimų formulavimo svarba projekto sėkmei, pasirodė apie 1956 metus. Tačiau kaip savarankiška programų sistemų inžinerijos šaka reikalavimų inžinerija buvo pripažinta tik 1976 metais. Tai buvo padaryta tais metais vykusioje tarptautinėje konferencijoje programų sistemų inžinerijos klausimais. Iki 1976 metų buvo manoma, kad programų sistemų inžineriją apima tik programų sistemų projektavimą, kodavimą ir testavimą. Tačiau ir po 1976 metų reikalavimų inžinerija buvo sutelkta tik reikalavimų analizės ir reikalavimų tvarkymo klausimus. Taigi, iš tiesų tai dar nebuvo savarankiška inžinerinė disciplina. Padėtis pasikeitė 1993 metais. Tais metais įvyko tarptautinis reikalavimų inžinerijos simpoziumas RE'93 [92]. Tai buvo pirmasis renginys ištisai skirtas vien tik reikalavimų inžinerijos klausimams. Nuo tada įsigalėjo vadinamas holistinis požiūris, sutinkamai su kuriuo reikalavimai pradėti naudoti visose projekto stadijose ir atitinkamos veikos buvo inkorporuotos į programų sistemų gyvavimo ciklo modelius.

1.2 Kam skirta ši knyga?

Ši knyga – tai vadovėlis. Jis skirtas informatikos specialybės magistrantūros studijų studentams. Daroma prielaida, kad jie jau yra išklausę šios specialybės bakalauro studijų studentams skaitomą įvadinį programų sistemų inžinerijos kursą ir žino, kas tai yra reikalavimų inžinerija bei kokius uždavinius ji sprendžia. Jie taip pat turėtų būti jau įvaldę šios disciplinos sąvokų sistemą ir suvokti, kokią vietą ji užima

visos programų sistemų inžinerijos kontekste. Visos svarbiausios reikalavimų inžinerijos sąvokos vadovelyje yra apibrėžtos ir netgi pridėtas aiškinamasis terminų žodynas. Tačiau išsamiai šios sąvokos nėra nagrinėjamos, nes tai turėjo būti padaryta bakalauro studijų studentams skirtame kurse. Manoma, jog skaitytojui pakanka tik priminti primirštą tą sąvoką prasmę. Todėl skaitytojams, kurie nėra susipažinę su reikalavimų inžinerijos pradmenimis, prieš skaitant šį vadovėlį patartina perskaityti kokį nors įvadui į reikalavimų inžineriją skirtą vadovėlį, pavyzdžiui, Ian K. Bray vadovėli *Introduktion to Requirements Engineering* [8], arba bent jau kokį nors įvadui į programų sistemų inžineriją skirtą vadovėlį, pavyzdžiui, Timothy Letbridge ir Robert Laganiere vadovėli *Object-Oriented Software Engineering: Practical Software Development using UML and Java* [72], Stephen Schach vadovėli *Object-Oriented and Classical Software Engineering* [103] ar labai populiarų Ian Sommerville vadovėli *Software Engineering* [106]. Deja, lietuvių kalba kol kas tokį vadovėlių nėra. Vienintelė šiai temai skirta knyga lietuvių kalba yra dvitomis *Programų sistemų inžinerijos pagrindai* [20]. Tačiau ši knyga yra monografinio pobūdžio ir skaitoma gana nelengvai.

Vadovėlis skirtas 48 valandos paskaitų kursui. Daroma prielaida, kad greta paskaitų studentai vykdys mokomajį projektą, kuriame praktiskai panaudos paskaitų metu įgytomis teorinėmis žiniomis ir jas įtvirtins.

Vadovėlis neapima visų reikalavimų inžinerijos nagrinėjamų klausimų. Kaip jau kalbėjome, tai labai plati ir daugiaplanė disciplina. Per vieną semestrą ją visą išdėstyti yra neįmanoma net ir tuomet, kuomet kursas yra skaitomas antrosios studijų pakopos studentams ir yra daroma prielaida, jog studentai jau yra susipažinę su šios disciplinos pradmenimis. Be to, reikalavimų inžinerija yra nuolat plėtojama ir tobulinama. Specialistai vis dar diskutuoja, kokius klausimus ir kokioje apimtyje reikėtų išdėstyti universitete. Dvi įtakingiausios pasaulyje specialistų organizacijos, IEEE ir ACM, parengė ir aprobavo oficialias rekomendacijas, jos vadinasi *SWEBOK®* [111]¹, numatančias, kokios programų sistemų inžinerijos žinios turėtų būti išdėstytos ir įsisavintos studijų universitete metu. Rekomendacijos apima ir nagrinėtinus reikalavimų inžinerijos klausimus. Šis vadovėlis yra parašytas griežtai vadovaujantis IEEE ir ACM parengtomis rekomendacijomis. Tai reiškia, kad vadovėlis siekia išmokyti:

- skirti vienus nuo kitų veiklos reikalavimus, vartotojo reikalavimus (operacinus poreikius), vartotojo reikalavimus tenkinančios kompiuterizuotos sistemos reikalavimus ir tos sistemos programinės įrangos reikalavimus;
- formuluoti funkcinius ir nefunkcinius programinės įrangos reikalavimus;
- paaiškinti, kuo skiriiasi skirtini reikalavimų inžinerijos proceso modeliai, ir pasirinkti konkrečiam projektui geriausiai tinkamą modelį;
- dirbtis vadovaujantis konkretaus reikalavimų inžinerijos proceso reikalavimais, matuoti ir vertinti to proceso sėkmingumą, suprasti, kaip tas procesas turėtų būti tobulinamas;
- nustatyti konkrečios sistemos reikalavimų šaltinius ir vertinti tų šaltinių patikimumą;

¹ Yra ir daugiau panašaus pobūdžio rekomendacijų. Paminėtiniai tarptautinės sistemų inžinerijos tarybos INCOSE parengtos rekomendacijos [42], rekomendacijos, ko reikėtų mokyti informacinių sistemų inžinierius [50], ko mokyti apie programų sistemų apsaugą ir apsaugos reikalavimus [94], ko mokyti apie programinės įrangos matavimus [11] ir ANSI/PMI standartas apie projektų vadybą [ST17]. Rengiant šį vadovėlį, buvo atsižvelgta į visuose tuose dokumentuose pateiktus reikalavimus.

- pasinaudoti dalykinės srities analizės metodais, naudojamais renkant reikalavimams suformuluoti reikalingą informaciją;
- formuluoti, modeliuoti, analizuoti ir vertinti reikalavimus;
- parinkti geriausiu būdu reikalavimus išgvendinančią programų sistemos architektūrą, susieti reikalavimus su konkrečiais tos architektūros komponentais, operacionalizuoti nefunkcinius reikalavimus;
- trasuoti reikalavimus, valdyti jų pokyčius ir atliki jų konfigūracijos valdymą.

Daugeliui realias programų sistemas kuriančių inžinierių to visiškai pakanka. Skaitytojams, norintiems giliau susipažinti su reikalavimų inžinerijos problematika, rekomenduojama pastudijuoti vadovėlio gale pateiktame literatūros sąraše išvardintas knygas [3], [5], [13], [13], [23], [25], [31], [32], [37], [39], [40], [44], [47], [48], [49], [52], [63], [64], [66], [68], [70], [74], [75], [76], [95], [107], [113], [117], [118], [119], [120], [121]. Ypač rekomenduotinos yra knygos [13], [23], [39], [52], [66], [68], [95], [117] ir [119].

1.3 Kuo ypatinga ši knyga?

Anglų kalba yra parašyta daug neblogų reikalavimų inžinerijos vadovelių. Natūraliai kyla klausimas, kam gi yra reikalingas dar vienas, originalus vadovėlis. Galbūt tikslingiau būtų išversti kurį nors iš esamų vadovelių? Juolab, kad vadovelyje pateikta medžiaga nėra, o ir negalėtų būti, kokių nors originalių tyrinėjimų rezultatas, nes vadoveliuose yra dėstomos tik tokios idėjos, kurios jau yra aprobuotos mokslinės visuomenės ir pasiteisino praktikoje. Be abejo, jos arba bent jau didžioji jų dauguma yra vienaip ar kitaip jau aptart esamuose vadoveliuose. Juolab, kad šiuo metu dauguma vadovelių yra rašoma prisilaikant minėtų IEEE ir ACM rekomendacijų. Pirmasis šių rekomendacijų variantas buvo parengtas dar 1993-2000 metais ir jomis jau spėjo pasinaudoti daugelio vadovelių autoriai. Kadangi programų sistemų inžinerija vystosi labai sparčiai, tai jos technologijos taip pat pasensta labai greitai ir yra pakeičiamos naujomis, modernesnėmis technologijomis. Todėl IEEE ir ACM padarė labai griežtą atranką ir į savo rekomendacijas stengėsi įtraukti tik fundamentines žinias, kuriomis grindžiamos visos technologijos ir kurios jau neturėtų pasenti. Manoma, kad su konkretiomis technologijomis studentai turėtų susipažinti po universiteto baigimo savo darbovietėse. Universitetas gi turi jiem suteikti tik fundamentines žinias, neturėdami kurių jie būtų nepajęgūs perprasti tokią technologiją. Taigi, net ir prieš keletą metų parašyti vadoveliai neturėjo pasenti. Tačiau, iš tiesų padėtis yra truputį kitokia. Praktiškai nei viename vadovelyje nepavyksta visiškai išvengti konkretių technologinių klausimų nagrinėjimo. Skirtingi autoriai renkasi skirtinges technologijas. Pasenus šioms technologijoms, pasensta ir jas nagrinėjantys vadoveliai. Be to, netgi ir fundamentines žinias kiekvienas autorius pateikia savaičių, jas nagrinėja vienu ar kitu detalumo laipsniu, akcentuoja tuos ar kitus aspektus ir dažniausiai IEEE ir ACM rekomendacijomis numatytais klausimais papildo klausimais, kurie, autoriaus nuomone, dėl vienų ar kitų priežasčių yra labai svarbūs. Taigi, kiekvienas vadovėlis yra savitas ir gerokai skiriasi nuo kitų reikalavimų inžinerijai skirtų vadovelių. Tuo nesunku įsitikinti palyginus kokius nors du populiarus vadovėlius, tarkime, Alan Davis vadovėlių *Software Requirements: Objects, Functions and States* [23] ir Suzanne ir James Robertson vadovėlių *Mastering the Requirements Process* [95]. Jie yra visiškai nepanašūs, nors abu yra skirti programų sistemų inžinerijos klausimams ir abu prisilaiko IEEE ir ACM rekomendacijų. Mūsų vadovėlis taip pat skiriasi nuo visų kitų reikalavimų inžinerijos vadovelių. Kokie gi yra jo ypatumai, kokius reikalavimų inžinerijos aspektus jis akcentuoja?

Visų pirma, vadovėlyje akcentuojamas požiūris, kad programų sistemų inžinerija yra viena iš inžinerinių disciplinų ir kaip tokia turėtų būti nagrinėjama bendrosios sistemų inžinerijos kontekste. Tas pats pasakytina ir apie programų sistemų reikalavimų inžineriją. Toks požiūris nėra naujas. Žymus amerikiečių reikalavimų inžinerijos specialistas Merlin Dorfman plačiai žinomame savo straipsnyje [25] šį požiūrį gana išsamiai išdėstė jau 1990 metais. Šiuo metu šis požiūris pamažu visuotinai įsigali. IEEE yra netgi paruošusi programą [12], pagal kurią turėtų laikyti egzaminus asmenys, norintys gauti programų sistemų inžineriaus sertifikatą. Tačiau, mūsų nuomone, esamuose reikalavimų inžinerijos vadoveliuose tokis požiūris kol kas vis dar yra akcentuojamas nepakankamai.

Antras mūsų vadovėlio ypatumas yra tas, kad Jame akcentuojama, jog programų sistemos visuomet yra kuriamos vienai ar kitai veiklai, pavyzdžiui, kokiam nors verslui, palaikyti ir todėl programų sistemos reikalavimai turi būti tiesiogiai siejami su tos veiklos tobulinimo strategija. Šiuo metu programų sistemos reikalavimų sąsajas su tos sistemos palaikomos veiklos reikalavimais akcentuoja daugelio vadovėlių autoriai. Pavyzdžiui, Hugh Beyer ir Karen Holtzblatt [5], Robin Goldsmith [37], David Hay [47], Daryl Kulak ir Eamonn Guiney [66], Dean Leffingwell ir Don Widrig [70] bei Karl Wiegers [117]. Tačiau apie programų sistemos reikalavimų sąsajas su veiklos tobulinimo strategija nei viename iš paminėtų vadovelių praktiskai nieko nėra kalbama. Kitaip tariant, esamuose vadoveliuose nėra parodyta, iš kur yra gaunami veiklos reikalavimai ir kaip jais remiantis nustatyti kuriamos programų sistemos vartotojų operacinius poreikius.

Siejant programų sistemos reikalavimus su veiklos tobulinimo strategija, natūraliai išskyla skirtingu reikalavimų lygmenų ir perėjimo nuo aukštesnio lygmens reikalavimų prie žemesnio lygmens reikalavimų, kitaip tariant, *reikalavimų nuleidimo žemyn* metodikos klausimai. Tiesa, daugiau kaip prieš 15 metų Merlin Dorfman minėtame savo straipsnyje [25] jau aptarė reikalavimų nuleidimo žemyn metodiką ir tą padarė bendrosios sistemų inžinerijos kontekste. Jo idėjos dėstomas daugelyje vadovelių. Tačiau iki pat paskutiniųjų metų buvo kalbama tik apie pačios programų sistemos architektūrinius lygmenis, neliečiant reikalavimų nuleidimo iš verslo lygmens į informacinės sistemos lygmenį ir iš informacinės sistemos lygmens į programų sistemos lygmenį klausimų. Tiesa, informacinių sistemų inžinerijoje apie perėjimą nuo verslo lygmens reikalavimų prie informacinės sistemos reikalavimų, o, po to, ir prie programų sistemos reikalavimų kalbama jau gana seniai. Bene pirmasis apie tai dar 1987 metais savo straipsnyje [122] prabilo John Zachman. Šiame straipsnyje pasiūlyta metodika buvo pavadinta autoriaus vardu, t. y. Zachmano metodika. Ji taip pat yra vadinama Zachmano architektūra, nes joje yra daromos tam tikros prielaidos apie tai, kokia turėtų būti informacinės sistemos konceptinio lygmens architektūra. Zachmano metodika numato šešis reikalavimų formulavimo lygmenis. Šie lygmenys aprašo skirtingu projekto dalyvių požiūrius į kuriamąjį programų sistemą. Tai verslo inžinieriaus arba verslo konsultanto (Zachmano terminais – verslo planuotojo) požiūris, dalykinės srities specialisto arba programų sistemos vartotojo (Zachmano terminais – verslo savininko) požiūris, informacinių sistemų inžinieriaus (Zachmano terminais – architekto) požiūris, programų sistemų inžinieriaus (Zachmano terminais – projektuotojo) požiūris, programuotojo (Zachmano terminais – konstruktoriaus) požiūris ir veikiančias sistemas aptarnaujančių bei prižiūrinčių tarnybų (Zachmano terminais – veikiančios sistemos) požiūris. Nuo verslo inžinieriaus lygmens yra pradedama todėl, kad Zachmanas nagrinėjo tik vieno tipo, būtent, verslo tipo veiklas. Iš tiesų, Zachmano metodikos

lygmenys gali būti apibendrinti taip, kad tiktų ir kitokio tipo veiklas, tarkime, gamybą ar technologinių procesų valdymą, palaikančioms programų sistemoms kurti. Tačiau šiame vadovėlyje mes, kaip ir Zachmanas, nagrinėsime tik verslui kompiuterizuoti skirtų programų sistemų reikalavimus. Taip darome konkretumo dėlei. Ribojimasis tokiomis programų sistemomis dėstymo bendrumo nesumažina, nes kitų sistemų reikalavimų lygmenys yra labai panašūs ir beveik visi vadovėlyje išdėstyti reikalavimų inžinerijos principai bei metodai joms yra tiesiogiai pritaikomi. Vienintelis skirtumas yra informacijos apie veiklos taisykles rinkimo būdai. Kiekvienam veiklų tipui informacija yra renkama iš kitokių šaltinių. Kuriant verslą palaikančias programų sistemas, didžioji tokios informacijos dalis yra surenkama iš dalykinės srities specialistų, o, tarkime, technologinių procesų valdymo sistemoms ji yra surenkama iš standartų, instrukcijų bei kitos techninės dokumentacijos. Aišku, jog iš skirtinį šaltinių informacija yra renkama skirtiniais būdais. Vadovėlyje nagrinėjami tik su verslą palaikančiomis programų sistemomis siejami informacijos rinkimo būdai. Kiti informacijos rinkimo būdai tame neaptarti. Taip daroma todėl, kad verslui kompiuterizuoti yra skirta didžioji dauguma Lietuvoje kuriamų programų sistemų.

Grįžtant prie Zachmano metodikos, reikia pabrėžti, kad šioje metodikoje vadovaujamas *turinių atskyrimo principu*. Verslo inžinieriaus požiūris atskirtas nuo dalykinės srities specialistų požiūrio, pastarasis – nuo informacinių sistemų požiūrio, šis – nuo PS inžinieriaus ir programuotojų požiūrių ir pagaliau išskirtas dar ir sistemą aptarnaujančių bei prižiūrinčių tarnybų požiūris. Tačiau ne visuomet turinius atskirti būtent šitaip yra geriausiai. Juos galima atskirti ir kitaip. Pavyzdžiu, ISO/IEC ir ITU-T (CCITT) standartais [ST18, ST29, ST30, ST31] numatytas vadinamasis atvirųjų išskirstytų skaičiavimų pavyzdinis modelis [13] numato tokius lygmenis: organizacijos, informacinių ir skaičiavimo paslaugų, inžinerinį ir technologinį. Organizacijos lygmuo apima verslo inžinieriaus ir dalykinės srities specialisto požiūrius, informacinių ir skaičiavimo paslaugų lygmuo atitinka informacinių sistemų inžinieriaus požiūrį, inžinerinis lygmuo atitinka PS inžinieriaus požiūrį ir technologinis lygmuo atitinka programuotojo požiūrį. Kitose metodikose turiniai yra atskirti dar kitaip. Kaip atskirti turinius priklauso nuo konkrečios metodikos paskirties. Šiame vadovėlyje išskiriami tie patys reikalavimų lygmenys kaip ir Zachmano metodikoje, tačiau, atsižvelgiant į vėlesniuosius reikalavimų inžinerijos pasiekimus, reikalavimų turinys yra traktuojamas šiek tiek kitaip, negu tai darė Zachmanas. Be to, nėra gilinamasi į Zachmano metodikos panaudojimo sistemoms projektuoti bei realizuoti klausimus.

Pažymėtina, kad praktikoje, kuriant verslui kompiuterizuoti skirtas programų sistemos, verslo reikalavimai ir informacinės sistemos reikalavimai išreikštiniu būdu dažnai nėra formuluojami. Taip esti todėl, kad dalykinės srities specialistai dažnai gali iš karto pasakyti, kokios programų sistemos jiems reikia ir projekta galima pradėti vartotojo ar netgi programų sistemos reikalavimų formulavimu. Tokiais atvejais reikalavimus formuluoja vadinamasis sisteminis analitikas. Nei verslo inžinieriai ar verslo konsultantai, nei informacinių sistemų inžinieriai projekte apskritai nedalyvauja. Teoriškai toks projekto vykdymo būdas nėra geras, nes negalima garantuoti, kad dalykinės srities specialistų nuomonė yra pagrįsta. Todėl iškyla realus atotrūkio tarp programų sistemos teikiamų paslaugų ir tikrujų verslo poreikių pavoju. Kaip teigia David C. Hay [47], neturint gerai apgalvotos organizacijos verslo plėtros ir tobulinimo strategijos, suformuluoti pagrįstus programų sistemos reikalavimus yra labai sunku. Tikroji programų sistemos vertė priklauso nuo to, kiek ši sistema

prisideda prie organizacijos misijos ir vizijos įgyvendinimo. Tačiau vargu ar galima turėti gerai apgalvotas bent kiek didesnės organizacijos misiją ir viziją, jei jos nėra suformuluotos išreikštiniu būdu. Be to, pradedant projektą programų sistemos reikalavimų formulavimu, informacinė sistema ir verslas yra traktuojami kaip viena visuma. Informacinė sistema apskritai nėra apmastoma ir projektuojama. Ji klostosi stichiškai, savaiminiu būdu, dėl ko anksčiau ar vėliau dažniausiai iškyla įvairių problemų. Tačiau, nedidelėms organizacijoms toks sistemų kūrimo būdas dažnai pasiteisina. Jis yra daug pigesnis, o mažos organizacijos verslo problemas ir jų sprendimo būdai dalykinės srities specialistams gali būti akivaizdūs. Būtent todėl daugelis reikalavimų inžinerijos vadovelių ignoruoja verslo ir informacinių sistemų reikalavimų lygmenis. Kiek yra žinoma šio vadovėlio autorui, visi reikalavimų lygmenys yra nagrinėjami tik viename programų sistemų reikalavimų inžinerijos vadovėlyje, būtent, David Hay vadovėlyje *Requirements Analysis. From Business Views to Architekture* [47]. Tačiau šiame vadovėlyje irgi yra vadovaujamasi gana pasenusiomis metodinėmis nuostatomis. Jis jau paseno. Taigi, trečiasis mūsų vadovėlio ypatumas yra tas, kad Jame nuosekliai, pradedant verslo reikalavimų lygmeniu, yra taikoma moderni reikalavimų nuleidimo žemyn metodika. Daroma prielaida, kad reikalavimai pradedami formuluoti nuo verslo lygmens ir lygmuo po lygmens yra nuleidžiamė žemyn iki realizacinių reikalavimo lygmens. Ši reikalavimų nuleidimo žemyn metodika apima ir nefunkcinių reikalavimų operacionalizavimo klausimus. Tam pasinaudota L. Chung, B.A. Nixon, E. Yu ir J. Mylopoulos rekomendacijomis [13].

Paskutinis mūsų vadovėlio ypatumas yra nuoseklus su UML kalba [113] siejamų reikalavimų inžinerijos nuostatų (verslo užduočių modeliavimas, RUP [65], reikalavimų modeliavimas UML kalba) taikymas. Nors šiomis nuostatomis vadovavosi daugelio šiuolaikinių reikalavimų inžinerijos vadovelių autoriai, pavyzdžiui, Ellen Gottesdiener [39], Ian Graham [40], Elizabeth Hull ir jos kolegos [49], Daryl Kulak ir Eamonn Guiney [66], Dean Leffingwell ir Don Widrig [70], Howard Podeswa [76] ir Ralph Young [119], jie ignoruoja vienus ar kitus mūsų aukščiau aptartus programų sistemų reikalavimų inžinerijos aspektus ir todėl negali būti rekomenduoti mūsų siūlomam kursui.

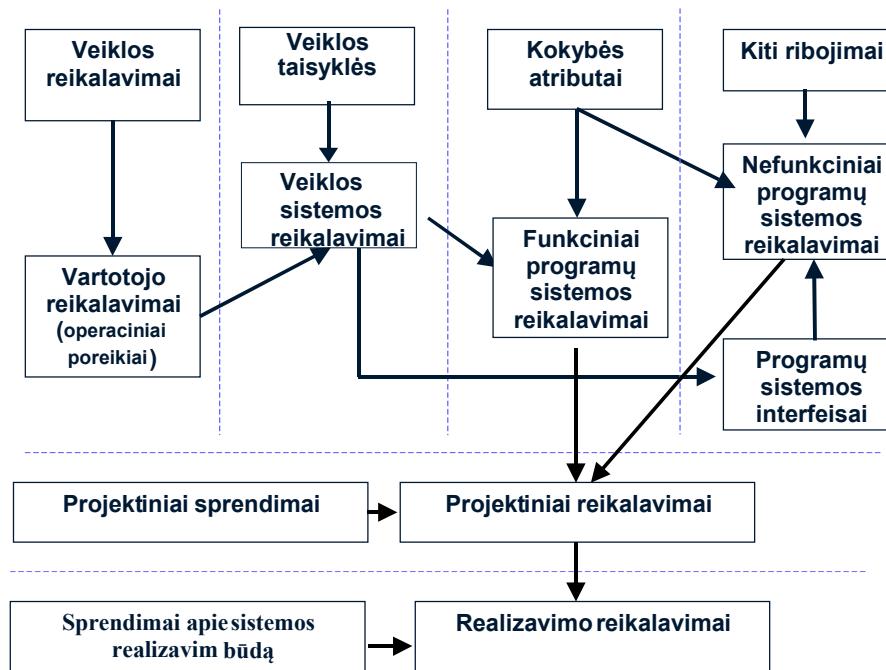
Taigi, šis vadovėlis buvo rašomas todėl, kad nei viename iš žinomų vadovelių nėra nuosekliai vadovaujamasi visais keturiais aukščiau aptartais požiūriais, darniai integrnuojant juos į vieną visumą, kas, mūsų nuomone, yra būtina moderniam programų sistemų reikalavimų inžinerijos vadoveliui.

2 Reikalavimų rūšys

2.1 Reikalavimų lygmenys

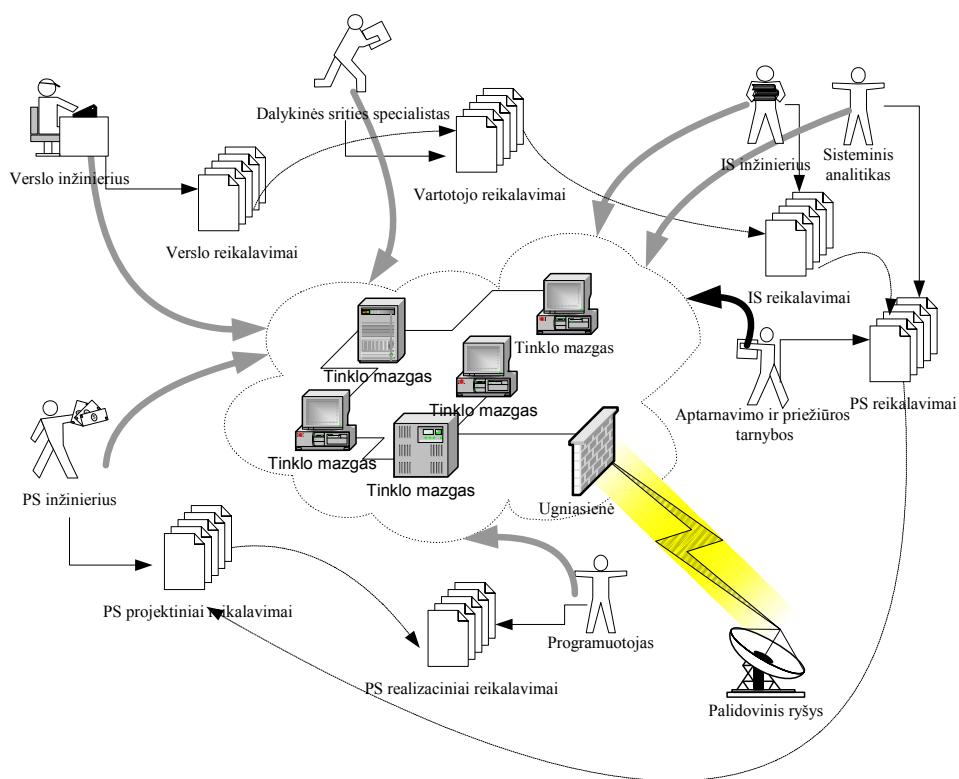
Bakalaurams skirtuose įvadiniuose programų sistemų inžinerijos kursuose programų sistemos reikalavimas paprastai yra apibrėžiamas kaip sandoriu su užsakovu, tos sistemos specifikacija, kokiui nors privalomu standartu arba kokiui nors kitu juridinę galią turinčiu dokumentu numatyta kuriamos programų sistemos savybė. Kitaip tariant, tai tokia kuriamos programų sistemos savybė, kurią būtinai reikia įgyvendinti, nes kitaip sistemą kurianti organizacija neįvykdys savo formalų įsipareigojimų. Ši apibrėžtis yra plačiai paplitusi. Pirmą kartą ją pateikė žymūs reikalavimų inžinerijos specialistai Richard Thayer ir Merlin Dorfman [112]. Tokia programų sistemos reikalavimo samprata, be abejo, yra teisinga, tačiau ji gana siaura. Čia išryškinta tik vieną iš termino "reikalavimas" prasmį. Ši prasmė yra bene pati svarbiausia, bet ne vienintelė. Su reikalavimais dirbama visą programų sistemos gyvavimo ciklo laikotarpį ir kiekvienoje šio ciklo stadijoje terminas "reikalavimas" yra suprantamas šiek tiek kitaip. Jeigu pirmosiose ciklo stadijose reikalavimai esti labai bendro pobūdžio ir aukšto abstrakcijos lygmens, tai paskutinėse stadijose jie esti labai griežti ir tikslūs, kartais netgi išreiškiami matematinėmis išraiškomis.

Programų sistemos nėra savitikslis. Jos kuriamos tam, kad padėtų atlikti tam tikrą darbą arba, kalbant plačiau, padėtų vykdyti kokią nors konkrečią veiklą ir pasiekti tam tikrų užsibrėžtų tikslų. Pavyzdžiu, programų sistema gali būti skirta kokiam nors, nebūtinai komercinio pobūdžio, verslui ar kokiam nors gamybos procesui kompiuterizuoti, kokiam nors objektui ar kokiam nors procesui stebeti ar valdyti. Todėl, kaip parodyta 1 paveikslėlyje, sistemos reikalavimų formulavimo procesas turėtų būti pradedamas veiklos reikalavimų formulavimu. Kadangi iš tiesų yra siekiama patobulinti ar išplėtoti kokią nors veiklą, tai visų pirma reikia suformuluoti būtent kokius tos veiklos aspektus norima tobulinti ar plėtoti. Tai ir yra veiklos reikalavimai.



1 pav. Programų sistemos reikalavimų lygmenys

Suformulavus veiklos reikalavimus reikia išsiaiškinti, kokių informacinių, skaičiavimo ar komunikacinių paslaugų prieiks tiems reikalavimams įgyvendinti. Kitaip tariant, reikia išsiaiškinti, kokie yra tą veiklą vykdančių subjektų (darbuotojų, padalinių, įrenginių ar pan.) *operaciniai poreikiai*, t. y. kokių informacinių, skaičiuojamujų ar komunikavimo paslaugų jiems reikia atliekant tą darbą, kurį jie privalo atlikti. Operaciniai poreikiai dar yra vadinami *vartotojo reikalavimais*. Vartotojas čia suprantamas plačiaja prasme, t. y. jis gali būti ne tik konkretus asmuo ar organizacinis padalinys, bet ir koks nors įrenginys ar procesas. Išsiaiškinus operaciniaus poreikius reikia suformuluoti tuos poreikius tenkinančios veiklai palaikyti skirtos sistemos reikalavimus. Veiklai palaikyti skirtą sistemą, pavyzdžiui, tai gali būti organizacijos informacinė sistema, sudaro ne tik planuojama kurti programų sistema, bet ir visas programų sistemos panaudojimo kontekstas, išskaitant žmones, įrenginius, sisteminę programinę įrangą, informaciją, informacines bei komunikavimo technologijas, darbo procedūras (taip pat ir rankines) ir kt. Kitaip tariant, formuluojant veiklą palaikančios sistemos reikalavimus yra nusprendžiama, kaip bus dirbama po to, kada planuojama kurti programų sistema bus sukurta ir įdiegta. Kadangi bet kuri veikla yra reglamentuojama tam tikromis taisyklėmis - verslo atveju tokios taisyklės yra vadinamos verslo taisyklėmis - tai sistemos reikalavimai turi būti suformuluoti taip, kad tos taisyklės nebūtų pažeistos. Vadovaujantis veiklą palaikančios sistemos reikalavimais, yra sprendžiama, kokias funkcijas turi vykdyti ir kokius interfeisus privalo įgyvendinti kuriamoji programų sistema. Kitaip tariant, pradedami formuluoti tiesioginiai programų sistemos reikalavimai, kurie ir yra aptarinėjami įvadiniuose programų sistemų inžinerijos kursuose. Šie reikalavimai baigiami formuluoti pridedant nefunkcinius reikalavimus, gaunamus atsižvelgiant į kokybės tikslų prioritetus bei kitus, papildomus reikalavimus. Naudojant standartines reikalavimų nuleidimo žemyn technikas, programų sistemos reikalavimai yra transformuojami į projektinius, o po to ir į realizacinius reikalavimus.



2 pav. Zachmano metodikos ir PS reikalavimų lygmenų sąsajos

1 lentelė. Zachmano metodika, pritaikyta konkrečiam užsakovui kuriamos sistemos reikalavimams formuluoti

	Kodėl? <i>(motyvacija)</i>	Kaip? <i>(veiklos)</i>	Ką? <i>(apdorojami objektai)</i>	Kas? <i>(funkciniai vienetai)</i>	Kur? <i>(vieta)</i>	Kada? <i>(laikas)</i>
Verslo reikalavimai (verslo inžinieriaus požiūris)	Misija, verslo sėkmės matavimų rezultatai, problemos, grėsmės, neišnaudotos galimybės, rinkos strategija, verslo vykdymo strategija, organizacijos veiklos strategijos, verslo nuostatos ir verslo taisyklos, vizija	Viziją įgyvendinantis verslo tikslų medis ir jo ribojimai.	Verslo tikslams pasiekti reikalingų verslo objektų reikalavimai	Verslo tikslų susiejimo su vykdytojais reikalavimai, vykdytojų įgaliojimų reikalavimai	Verslo tikslų susiejimo su darbo vietomis reikalavimai	Verslo našumo reikalavimai
Vartotojo reikalavimai (dalykinės srities specialisto požiūris)	Patobulintų verslo procesų reikalavimai	Informaciniai, skaičiavimo, komunikavimo ir kiti operaciniai poreikiai bei jų ribojimai.	Koncepciniai verslo objektus modeliuojančių informacinių objektų reikalavimai, išskaitant verslo duomenų apsaugos reikalavimus,	Kvalifikaciniai vykdytojų reikalavimai, preliminariniai panaudojamumo reikalavimai	Verslo sistemos darbo vietų reikalavimai	Verslo užduočių vykdymo našumo reikalavimai
IS reikalavimai (IS inžinieriaus požiūris)	IS vizija, IS galimybų medis, Organizacijos darbo su informacija taisyklos	IS architektūros reikalavimai, IS komponentų funkciniai, saugos ir patikimumo reikalavimai	Informacijos saugyklių ir jose saugomų informacinių objektų bei jų apsaugos reikalavimai	IS interfeisų ir panaudojamumo reikalavimai	IS darbo vietų reikalavimai (įskaitant Tl ir Pl)	Informacijos apdorojimo užduočių vykdymo našumo reikalavimai
PS reikalavimai (sisteminio analitiko požiūris)	Ekonominiai, politiniai ir teisiniai ribojimai, PS kokybės vertinimo kriterijai	PS funkciniai, saugos, patikimumo, diegimo, aptarnavimo ir priežiūros reikalavimai	PS apdorojamų duomenų, išskaitant jų apsaugą reikalavimai	Vartotojo interfeisų ir PS panaudojamumo reikalavimai	PS veikti reikalingos Tl ir sisteminės Pl reikalavimai, Tl išdėstymo reikalavimai	PS našumo reikalavimai
Projektiniai PS reikalavimai (PS inžinieriaus požiūris)	Eskizinio lygmens PS architektūros reikalavimai	PS komponentų funkciniai, saugos, robastiškumo, patikimumo, diegimo, aptarnavimo ir priežiūros reikalavimai	Duomenų bazių ir kitų duomenų saugyklių loginio lygmens reikalavimai	PS komponentų interfeisų reikalavimai	PS komponentų išdėstymo kompiuterių tinkle reikalavimai	PS komponentų našumo reikalavimai
Realizaciniai PS reikalavimai (programuotojo požiūris)	Detaliiosios PS architektūros reikalavimai	PS komponentų realizavimo ir testavimo reikalavimai	Duomenų bazių ir kitų duomenų saugyklių fizinio lygmens reikalavimai	Komponentų vidinių dalių (procedūrų, klasių ir kt.) interfeisų reikalavimai	Skirtinguose tinklo mazguose veikiančių komponentų sąveikos reikalavimai	Algoritmų efektyvumo reikalavimai

Pabandykime susieti reikalavimų lygmenis (2 pav.) ir Zachmano metodikos lygmenis (1 lentelė). Kadangi mes nagrinėjame tik tai verslui kompiuterizuoti skirtas

programų sistemas, tai veiklos reikalavimai yra suprantami kaip verslo reikalavimai, o veiklos taisykles – kaip verslo taisykles. Verslo reikalavimai aprašo verslo inžinieriaus (verslo konsultanto) požiūrį į kuriamą programų sistemą. Šie reikalavimai nustato strateginius verslo tikslus, nusakančius kompiuterizuojamo verslo plėtros ir tobulinimo kryptis. Šitaip nusakomi programų sistemos pačio aukščiausio lygmens reikalavimai, nes ši sistema yra kuriama, siekiant išspręsti verslo problemas ir padėti įgyvendinti strateginius verslo tikslus.

Dalykinės srities specialistų požiūrį aprašo vartotojo reikalavimai. Šie reikalavimai nusako, kokių konkrečių informacinės sistemos paslaugų (nebūtinai, kompiuterizuotų) reikia dalykinės srities specialistams. Jie gaunami detalizuojant strateginius verslo tikslus iki operacinių ir taktinių tikslų lygmens, išreiškiant juos verslo procesų bei užduočių terminais ir nustatant, kokių informacinių, skaičiavimo ar komunikacių paslaugų prireikia vykdant tas užduotis.

Informacinių sistemų inžinieriaus požiūrį aprašo informacinės sistemos reikalavimai. Jie gaunami detalizuojant ir konkretizuojant vartotojo reikalavimus. Tai daroma atsižvelgiant į verslo taisykles. Be kita ko čia yra sprendžiama, kurios informacinės sistemos teikiamos paslaugos turi būti kompiuterizuotos, t. y. sprendžiama, kokias programų sistemas reikia sukurti informacinei sistemai palaikyti. Išsamūs funkciniai ir nefunkciniai programų sistemos reikalavimai formuluojami poto. Jie aprašo sisteminio analitiko požiūrį.

Taigi, tarp trečiojo ir ketvirtojo Zachmano metodikos lygmenų tenka įvesti papildomą lygmenį, aprašantį sisteminio analitiko požiūrį. Zachmanui jo neprireikė, nes informacines sistemas jis faktiškai tapatino su jų programine įranga. Sisteminio analitiko lygmuo apima net du 1 paveikslėlyje parodytus reikalavimų lygmenis: funkcinius PS reikalavimus ir nefunkcinius PS reikalavimus (įskaitant vartotojo interfeiso reikalavimus). Kaip jau minėjome, kuriant mažoms organizacijoms skirtas paprastas programų sistemas, reikalavimų formulavimas gali būti pradėtas nuo šio lygmens.

PS inžinieriaus požiūrį aprašo projektiniai PS reikalavimai. Šiame lygmenyje priimami sprendimai, kokia infrastruktūra (kompiuterinė platforma, DBVS tipas ir kt.) bus naudojama programų sistemai kurti, parenkama tos sistemos architektūra ir, remiantis tos sistemos reikalavimais, suformuluojami jos komponentų reikalavimai. Tai yra padaroma naudojant tradicines programų sistemos reikalavimų lokalizavimo ir nuleidimo žemyn procedūras.

Programuotojo požiūrį aprašo realizacinių PS reikalavimai. Jie nusako, kokiomis programavimo kalbomis, kokiais kompiliatoriais, kokia tarpine programine įranga ir kokiomis kitomis konkrečiomis instrumentinėmis priemonėmis gali naudotis programuotojai, kurdami šią programų sistemą.

Zachmano metodika numato dar vieną lygmenį, tapatinamą su sukurtą programų sistemą prižiūrinčiu ir aptarnaujančiu specialistų požiūriu. Šis požiūris nėra aprašomas kokiais nors savarankiškais reikalavimais. I ji yra atsižvelgiama formuluojant tiek funkcinius, tiek ir nefunkcinius programų sistemos reikalavimus.

Kaip parodyta 1 lentelėje, Zachmano metodika yra aprašoma lentele, kurios eilutės atitinka šios metodikos lygmenis (t. y. skirtingus požiūrius į kuriamą programų sistemą). Siekiant palengvinti reikalavimų formulavimą, reikalaujama, kad kiekviename lygmenyje būtų atsakoma į klausimus “Kodėl?”, “Kaip?”, “Ką?”, “Kas?”, “Kur?”, “Kada?”. Pradiniame metodikos variante [122] buvo tik trys

klausimai: "Ką?", "Kaip?" ir "Kur?". Likusiais klausimais 1992 metais metodiką papildė John Sowa [109]. Taigi, šiuo metu metodika reikalauja, kad rengiant bet kurio lygmens reikalavimus būtų atsakoma į šešis klausimus. Šiuos klausimus atitinka 1 lentelės stulpeliai. Originali Zachmano metodika (ji išsamiai aprašyta David Hay vadovelyje *Requirements Analysis. From Business Views to Architekture* [47].) daugiau yra pritaikyta sistemos projektavimo, o ne reikalavimų formulavimo poreikiams. Todėl lentelės langelių turinys ten yra kitoks negu mūsų pateiktoje lentelėje. Be to, pas mus yra sukeista ir klausimų tvarka. Mes pritaikėme Zachmano metodiką reikalavimų formulavimo poreikiams. Taigi, mūsų aprašinėjama metodika turi kitą paskirtį negu originali Zachmano metodika. Todėl atsakymai į metodikos numatytyus klausimus pas mus yra visiškai kitokie. Be to, mes vadovaujamės požiūriu, kad bet kurio lygmens reikalavimų formulavimas turi būti pradedamas nuo motyvacijos nagrinėjimui. Todėl pas mus lentelės stulpeliai yra išdėstyti kitokia tvarka. Tačiau mūsų aprašinėjamoje metodikoje yra išlaikyti visi Zachmano metodiniai principai ir todėl toliau mes ją vadinsime tiesiog Zachmano metodika. Tai neturėtų suklaidinti skaitytojų, nes apie originaliąjį Zachmano metodiką toliau šiam vadovelyje mes nieko nebekalbėsime.

Aptarkime 1 lentelės stulpelius išsamiau.

Pirmajame stulpelyje atsakinėjama į klausimą "Kodel?". Viršutinėje lentelės eilutėje čia pateikiama motyvacija, kodėl norima tobulinti verslą. Tai padaroma aprašant verslo misiją, problemas bei grėsmes, trukdančias ją įgyvendinti ir viziją, kaip patobulinti verslą. Leidžiantis eilutę po eilutės šiuo stulpeliu lentele žemyn, yra nusileidžiama iki reikalavimų, kokia turi būti programų sistema, padedanti šią viziją įgyvendinti.

Antrajame stulpelyje atsakinėjama į klausimą "Kaip pasiekti pirmame stulpelyje suformuluotus tikslus?". Pirmojoje eilutėje čia verslo tikslų medžio pavidalu yra pateikiama verslo tobulinimo strategija, detalizuojanti patobulinto verslo viziją. Po to, leidžiantis šiuo stulpeliu eilutę po eilutės lentele žemyn, yra nusprendžiama:

- kokių informacinių, skaičiavimo ir komunikacinių paslaugų reikia, siekiant tikslų medyje numatytyų strateginių ir operaciinių tikslų, nusileidžiama iki reikalavimų,
- kokiomis IS procedūromis galima pasinaudoti toms paslaugoms gauti ir kiek saugios bei patikimos turi būti tos procedūros,
- kokie turi būti tas IS procedūras palaikančios PS funkcionalumas, patikimumas, robastiškumas ir sauga,
- kokie turi būti kiekvieno iš tų PS sudarančių komponentų funkcionalumas, patikimumas, robastiškumas ir sauga,
- kokiomis priemonėmis tie komponentai turi būti kuriami ir kokiu mastu turi būti testuojančios kiekvienas iš jų.

Trečiajame stulpelyje atsakinėjama į klausimą, "Ką turi apdoroti antrajame stulpelyje numatytos procedūros arba, tiksliau, kokius reikalavimus turi tenkinti tomis procedūromis apdorojami objektai?". Viršutiniame lygmenyje kalbama apie verslo objektus, po to apie juos modeliuojančius informacinius objektus, dar vėliau – apie tuos informacinius objektus kompiuteryje vaizduojančius skaitmeninius objektus, t. y. atitinkamus duomenis.

Ketvirtajame stulpelyje atsakinėjama į klausimą "Kas naudosis programų sistemos teikiamomis paslaugomis ir kokius įgaliojimus jie turi?" Kitaip tariant, čia kalbama apie dalykinės srities specialistus, įgyvendinančius verslo tikslus ir jų įgaliojimus. Leidžiantis lentele žemyn čia yra nusileidžiama iki programų sistemos interfeisų ir jos apsaugos reikalavimų.

Penktajame stulpelyje atsakinėjama į klausimą "Kur (t. y. kokiose darbo vietose) dirbs dalykinės srities specialistai?". Leidžiantis lentele žemyn čia yra nusileidžiama iki techninės įrangos ir programinės įrangos išdėstymo kompiuterių tinkle reikalavimų.

Šeštajame stulpelyje atsakinėjama į klausimą "Kada?". Kitaip tariant, šis stulpelis yra skirtas verslo užduočių, IS, PS ir jos komponentų našumo reikalavimams formuluoti.

Pažymėtina, kad originalijoje Zachmano metodikoje pirmajame stulpelyje atsakinėjama į klausimą "Ką?", t. y. reikalavimų analizė yra pradedama nuo verslo duomenų analizės. Tokiu pat metodiniu požiūrio savo vadovelyje *Requirements Analysis. From Business Views to Architekture* [47] vadovaujasi ir David Hay. Tačiau šiuo metu vyrauja kitoks požiūris. Manoma, kad reikia pradėti nuo verslo užduočių ir jas vykdančių dalykinės srities specialistų informacinių, skaičiavimo ir komunikavimo poreikių analizės. Kokias verslo užduotis tikslina kompiuterizuoti, savo ruožtu, išplaukia iš verslo plėtros ir tobulinimo strategijos analizės. Kaip jau buvo minėta, tai ir yra vienas iš svarbiausių mūsų vadovėlio skirtumų nuo daugumos kitų reikalavimų inžinerijos vadovelių.

2 lentelė. Zachmano metodika, pritaikyta rinkoje skirtos parduoti programinės įrangos reikalavimams formuluoti

	Kodėl? <i>(motyvacija)</i>	Kaip? <i>(veiklos)</i>	Ką? <i>(apdorojami objektai)</i>	Kas? <i>(funkciniai vienetai)</i>	Kur? <i>(vieta)</i>	Kada? <i>(laikas)</i>
Verslo reikalavimai (rinkos analitiko požiūris)	Rinkos segmentas, kuriam skiriamas produktas. Rinkos, kuriose produktas turėtų būti konkurencingas. Esamų produkų analizė. Problemos, kurių negalima išspręsti arba kurios sprendžiamos neefektyviai be šio produkto. Kiti produkto pranašumai. Produktų vizija.	Viziją įgyvendinantis produkto galimybų medis ir jo ribojimai.	Realaus pasaulio objektai, su kuriais arba su kurį informaciniais modeliais dirba (valdo, apdoroja, kuria, analizuja ir kt.) produktas	Produkto galimybų susiejimas su potencialiomis vartotoju grupėmis ir tų grupių teisių bei įgaliojimų nustatymas	Produkto galimybų susiejimas su darbo vietomis, kuriose tomis galimybėmis galima pasinaudoti	Produkto galimybų našumo reikalavimai
Vartotojo reikalavimai (dalykinės srities specialisto požiūris)	Vykdomos užduotys	Informacinių, skaičiavimo, komunikavimo ir kitos paslaugos, reikalingos užduotims realizuoti.	Koncepciniai informaciinių objektų, su kuriais dirba produktas, reikalavimai, išskaitant jų apsaugą	Kvalifikacioniai vartotoju reikalavimai, preliminarūs produkto panaudojamumo reikalavimai	Bendrieji darbo vietu reikalavimai	Užduočių vykdymo našumo reikalavimai
PS reikalavimai	Ekonominiai, politiniai ir teisiniai	Produkto funkciniai,	Produkto apdorojamų	Vartotojo interfeisų ir	Produktui veikti	Produkto našumo

	Kodėl? <i>(motyvacija)</i>	Kaip? <i>(veiklos)</i>	Ką? <i>(apdorojami objektai)</i>	Kas? <i>(funkciniai vienetai)</i>	Kur? <i>(vieta)</i>	Kada? <i>(laikas)</i>
(sisteminio analitiko požiūris)	ribojimai, produkto kokybės vertinimo kriterijai	saugos, patikimumo, diegimo, aptarnavimo ir priežiūros reikalavimai	duomenų reikalavimai, produkto apsaugos reikalavimai	produkto panaudojamumo reikalavimai	reikalingos TĮ ir sisteminės PI reikalavimai	reikalavimai
Projektiniai PS reikalavimai (PS inžinieriaus požiūris)	Eskizinio lygmens PS architektūros reikalavimai	PS komponentų funkciniai, saugos, robastiškumo, patikimumo, diegimo, aptarnavimo ir priežiūros reikalavimai	Loginio lygmens duomenų baziu ir kitų duomenų saugykļų reikalavimai	PS komponentų interfeisių reikalavimai	PS komponentų išdėstymo kompiuterių tinkle reikalavimai	PS komponentų našumo reikalavimai
Realizaciniai PS reikalavimai (programuotojo požiūris)	Detaliосios PS architektūros reikalavimai	PS komponentų realizavimo ir testavimo reikalavimai	Fizinio lygmens duomenų baziu ir kitų duomenų saugykļų reikalavimai	Komponentų vidinių dalių (procedūrų, klasių ir kt.) interfeisių reikalavimai	Skirtinguose tinklo mazguose veikiančių komponentų sąveikos reikalavimai	Algoritmų efektyvumo reikalavimai

Kitą vertus, pradėti nuo verslo plėtros ir tobulinimo strategijos ne visuomet yra įmanoma. Kuriant sistemas rinkai, o ne konkretiems užsakovams, negalima atlikti nei verslo sėkmės matavimų, nei suformuluoti kokią nors visiems potencialiems pirkėjams bendrą verslo plėtros bei tobulinimo strategiją. Tačiau ir šiuo atveju reikia pradėti nuo klausimo “Kodėl?”. Atlikus rinkos analizę, galima išsiaiškinti potencialių pirkėjų tipines problemas ir pasiūlyti priemonę toms problemoms spręsti. Tačiau, kuriant sistemas rinkai, 1 lentelėje aprašytoji darbo su reikalavimais metodika, ypač tai, kas parašyta pirmosiose lentelės eilutėse, gerokai skiriasi (2 lentelė). Rinkai kuriamoms sistemoms, išskyrus vadinamuosius ERP paketus, apskritai dingsta trečioji lentelės eilutė, aprašanti informacinių sistemų inžinieriaus požiūrį. Taip yra todėl, kad nėra žinoma, kokiam konkrečiam kontekste bus naudojamas kuriamasis produktas. Kadangi ERP paketą neįmanoma atskirti nuo konkrečios informacinės sistemos konteksto, tai jie “atsineša” tą kontekstą su savimi arba, kitaip tariant, yra reikalaujama, kad produktą įsigijusi organizacija taip organizuotų savo informacinę sistemą, kaip tą užmanė produkto kūrėjai. Kaip tai yra daroma, šioje knygoje nenagrinėsime, todėl 2 lentelėje trečią eilutę mes paprasčiausiai praleidome.

Kadangi didžioji dauguma programinę įrangą kuriančių Lietuvos bendrovii ją kuria ne rinkai, bet konkretiems užsakovams, tai toliau šiame vadovelyje pagrindinių dėmesį skirsiame 1 lentele aprašytai metodikai, tiktais trumpai aptardami klausimus, susijusius su 2 lentele aprašyta metodika.

2.2 *Produkto reikalavimai ir projekto reikalavimai*

Praeitame skyrylyje aptarti reikalavimai, išskyrus PS komponentų testavimo reikalavimus, yra kuriamo produkto reikalavimai. Terminą “produktas” mes vartojaame plačiąja prasme, t. y. produkту vadiname ne tik kuriamają programų sistemą, bet ir jos palaikomą informacinę sistemą ir netgi patobulintą verslo procesą. Reikia gerai įsisavinti, kad, kuriant programų sistemą, yra siekiama sukurti visus tuos

produktaus ir kad iš tiesų užsakovams reikia ne programų sistemos, o sėkmingo verslo. Šiemis produktams sukurti turi būti organizuotas atitinkamas projektas. Norint vykdyti projektą, taip pat reikia suformuluoti atitinkamus reikalavimus. Šie reikalavimai vadinami projekto reikalavimais. 1 paveikslėlyje jie neparodyti. Tipiniai projekto reikalavimų pavyzdžiais yra projekto trukmė ir jo kaina. Kiti reikalavimai apima projekto valdymo būdą, kokybės kontrolės procedūras, sistemos projektavimo metodus, jos testavimo apimtis ir pan. Iš pirmo žvilgsnio atrodo, kad sistemos ir projekto reikalavimai tarpusavyje yra mažai, o gal ir visiškai nesusiję ir todėl gali būti formuluojami nepriklausomai vieni nuo kitų. Tačiau toks įspūdis yra klaidingas. Iš tiesų sistemos ir projekto reikalavimai susiję labai tampriai ir dažnai vieni su kita susipina labai sudėtingu būdu. Pavyzdžiu, pareikalavus didelio sistemos patikimumo laipsnio, tenka didinti jos testavimo apimtis, o tai daro įtaką daugeliui projekto reikalavimų, įskaitant projekto trukmę ir jo kainą. Kita vertus, parinkus konkretų projekto valdymo būdą, tenka atitinkamu būdu organizuoti projekto vykdytojų kolektyvą, o tai, savo ruožtu, gali turėti įtakos kuriamos programų sistemos architektūrai. Taigi, sistemos reikalavimai ir projekto reikalavimai turi būti formuluojami kartu ir jie privalo derėti vieni su kita. Kadangi šiame vadovėlyje pagrindinis dėmesys yra skirtas produkto, visų pirma programų sistemos, reikalavimams, tai projekto reikalavimų išsamiau neaptarinėsime ir pereisime prie detalesnio skirtingo lygmens produkto reikalavimų nagrinėjimo. Apie projekto reikalavimus galima pasiskaityti specialiai šiam klausimui skirtuose vadovėliuose.

2.3 Funkciniai ir nefunkciniai produkto reikalavimai

Funkciniai programų sistemos ar kokio nors kito produkto (verslo sistemos, informacinių sistemų, technologinių procesų valdymo sistemos ar pan.) reikalavimai nusako, kokias funkcijas turi turėti tas produktas, kad būtų patenkinti užsakovo (vartotojų) poreikiai ir kokios yra tų funkcijų pageidaujamos savybės.



3 pav. Funkciniais reikalavimais aprašomos kuriamo produkto funkcijų savybės

Funkciniai reikalavimai yra skirstomi į šešias grupes (3 pav.):

- produkto tinkamumo funkciniu požiūriu reikalavimus;
- produkto generuojamų rezultatų tikslumo reikalavimus;
- produkto sąveikos su kitaisiais produktais (interoperabilumo) reikalavimus;
- produkto atitikimo galiojantiems aktams ir standartams (darnos su tais aktais ir standartais) reikalavimus;
- produkto apsaugotumo reikalavimus;
- produkto trasuojamumo reikalavimus.

Produkto tinkamumo (angl. *suitability*) reikalavimai aprašo kokias funkcijas turi turėti kuriama programą ar kokia nors kita sistema. Kiekybinis produkto tinkamumo matas yra to produkto užsakovo (vartotojų) operacinių poreikių padengimo produkto funkcijomis procentas ir to padengimo išsamumas, kurį taip pat galima įvertinti procentais.

Tikslumo (angl. *accuracy*) reikalavimai yra skirtomi į objektų vaizdavimo sistemoje tikslumo ir tų objektų apdorojimo tikslumo reikalavimus. Vaizdavimo tikslumo reikalavimai aprašo, kiek detaliai informaciniai ar kiti objektais turi būti vaizduojami sistemoje, kad skirtingi realaus pasaulio objektai joje būtų atskiriami vienas nuo kito. Objektų apdorojimo tikslumo reikalavimai aprašo leistinas objektų apdorojimo (skaičiavimų, transformavimo iš vieno pavidalo į kitą ir pan.) paklaidas.

Sistemos interoperabilumo (angl. *interoperability*) reikalavimai apibūdina sistemos gebėjimą kooperuotis su kitomis sistemomis. Jie aprašo:

- pastangas, reikalingas sistemos sąveikai su kitomis, iš anksto nežinomomis sistemomis realizuoti;
- formatus duomenų, kuriais sistema turi keistis su kitomis sistemomis;
- grafinius ir valdančiuosius ženklus, kuriais sistema turi keistis su kitomis sistemomis;
- sąveikos su kitomis sistemomis interfeisus;
- sąveikos su kitomis sistemomis standartus.

Produkto atitikimo galiojantiems teisės aktams bei standartams (angl. *compliance*) reikalavimai aprašo:

- kokių teisės aktų ir standartų reikalavimus turi tenkinti kuriamas produktas;
- kokių informacinių objektų vaizdavimo formatai ir kaip turi būti standartizuoti kuriamojoje sistemoje;
- kokių media (garsų, vaizdų ir kt.) saugojimo formatai ir kaip turi būti standartizuoti kuriamojoje sistemoje;
- kokie grafiniai ir valdantieji ženklai ir kaip turi būti standartizuoti kuriamojoje sistemoje;
- kokie interfeisai ir kaip turi būti standartizuoti kuriamojoje sistemoje;
- kurios kuriamos sistemos funkcijos ir kokius standartus privalo tenkinti.

Produkto apsaugotumo (angl. *security*) reikalavimai apibūdina kokių mastų kuriamoji sistema turi būti apsaugota nuo tyčinio ar netyčinio jos funkcionalumo ar joje saugomų duomenų panaudojimo neteisėtu būdu (t. y. neturint tam reikiamų įgaliojimų). Kiekybiškai tokius reikalavimus galima suformuluoti, nurodant:

- tikimybę, kad, turint nurodytas resursų (pinigų, laiko, aparatūros) apimtis, galima apeiti sistemos apsaugos priemones;
- per kiek laiko patyrę hakeriai, turėdami fizinę prieigą prie sistemos, turi nepajėgti į ją įsilaužti;

- kokie sistemoje saugomi duomenys turi būti saugomi užšifruoti;
- prie kokio didžiausio procento sistemoje saugomų konfidentialių duomenų gali būti neteisėtai prieita per nurodytą laiko periodą;
- ne daugiau kaip kiek kartų per nurodytą laiko periodą gali būti sugadinti sistemoje saugomi duomenys.

Sistemos trasuojamumo (angl. *traceability*) reikalavimai apibūdina pastangas, kurių reikia tam, kad pasirinktuose sistemos taškuose būtų galima patikrinti jos generuojamų tarpinių rezultatų teisingumą. Kiekybiškai tie reikalavimai gali būti suformuluoti nurodant:

- kiek laiko, skaičiuojant procentais nuo viso sistemos veikimo laiko, leidžiama sistemai sugaišti trasavimui palaikyti;
- kiek laiko, skaičiuojant procentais nuo viso užduočiai atliki reikalingo laiko, turi pakakti žmogui sistemos trasavimui atliki.



4 pav. Nefunkciniai produkto reikalavimai

Nefunkciniais produkto reikalavimais (4 pav.) vadinami ribojimai, kurie, nusakydami nefunkcines produkto savybes (patikimumą, panaudojamumą ir kt.), vienaip ar kitaip ribojantys kuriamo produkto įgyvendinimo būdą. Kadangi nefunkciniai reikalavimai apibūdina visas nefunkcines produkto savybes, jie dar yra vadinami produkto kokybės reikalavimais.

Produkto patikumo (angl. *reliability*) reikalavimai aprašo ne mažiau kaip kokį laiko periodą kuriamoji sistema, jei ji bus tinkamai eksplotuojama, privalo išlikti korektiška ir produktyvi.

Produkto panaudojamumo (angl. *usability*) reikalavimai riboja maksimalias pastangas, kurių turi pakakti vartotojams pasinaudoti kuriana sistema.

Produkto prižiūrimumo (angl. *maintainability*) reikalavimai aprašo, kokių pastangų turi pakakti sistemos perdarymams atliki.

Produkto našumo (angl. *efficiency*) reikalavimai aprašo sistemos veikimo greičio ir, prie nurodytų sąlygų, jos naudojamų išteklių santykį.

Perkeliamumo (angl. *portability*) reikalavimai apibūdina kuriamos sistemos galimybes kelti iš vienos platformos į kitą. Jie yra formuluojami tik programų sistemoms.

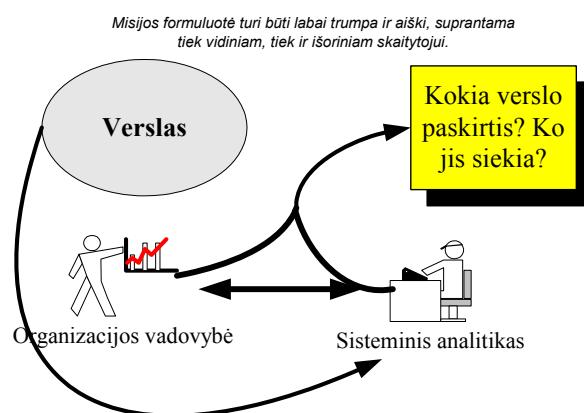
2.4 Verslo reikalavimai

2.4.1 Verslo reikalavimų formulavimo ypatumai

Vadovaujantis 1 lentelėje pateikta metodika, reikalavimų formulavimas pradedamas verslo reikalavimų formulavimu. Verslo reikalavimai aprašo pačias bendriausias operacines būsimosios programų sistemos savybes t. y. aprašo, kaip ta sistema bus naudojamas versle ir ką verslas iš to laimės. Kadangi, bent jau kuriant konkretiems užsakovams skirtas programų sistemas, verslo reikalavimus skaito ir vykdytojai, ir užsakovai, tai šie reikalavimai turi būti rašomi labai taisyklinga ir paprasta kalba, vengiant specialių informatikos terminų ir, jei tokią terminų išvengti nepavyksta, išsamiai ir užsakovui suprantamai juos paaiškinant. Kitaip tariant, reikalavimai turi būti taip aprašyti, kad juos skaitant nei dalykinės srities specialistams, nei programų sistemų inžinieriams nereikėtų bendrauti su autoriais ir prašyti jų ką nors paaiškinti. Kadangi verslo reikalavimai aprašo verslo inžinieriaus arba verslo konsultanto požiūrį į kuriamąjį verslo sistemą, tai juos ir turėtų formuliuoti verslo inžinieriai arba verslo konsultantai. Deja, praktikoje dažnai esti kitaip ir verslo reikalavimus, jeigu jie apskritai yra formuluojami, neretai tenka formuliuoti informacinių sistemų inžinieriams arba netgi programų sistemų inžinieriams. Tačiau, kad ir kas beformuluotų reikalavimus, tai darydamas jis veikia kaip sisteminis analitikas. Taip toliau šiame poskyryje mes jį ir vadinsime. Kadangi mūsų vadovėlis visų pirma nagrinėjamas darbas su žemesnių lygmenų reikalavimais, t. y. tiesioginiai programų sistemų reikalavimais, tai verslo reikalavimų formulavimo problemas mes aptarsime gana trumpai, tik tiek, kiek reikia tam, kad būtų galima suprasti iš kur turi būti gaunami tiesioginiai programų sistemos reikalavimai.

2.4.2 Kodėl?

Sistemoms, kuriamoms konkrečiam užsakovui. Verslo reikalavimų, kaip ir visų kitų lygmenų reikalavimų, formulavimas pradedamas atsakymu į klausimą „Kodėl?” arba, tiksliau kalbant, bandymu atsakyti į klausimą, kodėl reikia kurti programų sistemą. Norint atsakyti į šį klausimą, visų pirma būtina suvokti ir išreikštiniu būdu suformuluoti *verslo misiją*. (5 pav.).



5 pav. Verslo misijos formulavimas

Gali būti, kad organizacijos vadovybė jau anksčiau yra suformulavusi verslo misiją. Tokiu atveju sisteminiam analitikui reikia tik susipažinti su ja, atliskti jos analizę ir galbūt ją patikslinti. Priešingu atveju, sisteminis analitikas formuluoja verslo misiją kartu su organizacijos vadovybe. Be abejo, tam, kad sisteminis analitikas galėtų suformuluoti verslo misiją, jis turi išsamiai susipažinti su pačiu verslu. Bet kuriuo atveju misija turi būti formulojama trumpai ir aiškiai, taip, kad ji būtų lengvai suprantama ir organizacijos darbuotojams, ir jos klientams, ir programų sistemų inžinieriams. Misija turi būti oficialiai aprobuota organizacijos vadovybės.

Pavyzdys

Gerai suformuluotos misijos pavyzdys.

Sutinkamai su valstijos įstatymais, Y universiteto ligoninė ir klinikos veikia kaip valstijos medicinos darbuotojų mokymo centras ir kaip visapusiškas sveikatos priežiūros centras, besirūpinantis visų valstijos gyventojų sveikata, nepriklausomai nuo tų gyventojų mokumo. Y universiteto ligoninė ir klinikos kartu su Y universiteto medicinos mokslų koledžu konsultuoja valstijos ir kitų valstijų sveikatos apsaugos specialistus bei organizacijas ir teikia jiems kitokią profesinę pagalbą. Y universiteto ligoninė ir klinikos:

- teikia platų medicininių paslaugų spektrą jose besigydantiems pacientams ir, per ligoninės ir klinikų vykdomas programas, neįgaliesiems valstijos gyventojams;
- yra naudojamos kaip pagrindinė bazė Universitete studijuojančių studentų praktikai atliskti;
- atlieka inovacinius tyrimus, kuriais siekiama tobulinti sveikatos apsaugą.

Blogai suformuluoto misijos pavyzdys.

X ligoninės (XUL) klinikos yra viena didžiausių šalies ligoninių, teikiančių aukščiausio lygio specializuotas asmens sveikatos priežiūros paslaugas bei vykdančių nuolatinę pedagoginę bei mokslinę tiriamajį darbą.

XUL klinikose būtinają ir planinę pagalbą teikia aukščiausios kvalifikacijos gydytojai ir slaugytojos, konsultacijų poliklinikoje konsultuoja bei tolimesniams gydymui siunčia patyrę antro ir trečio lygio specialistai-konsultantai.

Ligoninėje rasite daugumos medicinos sričių gydymo, slaugos bei reabilitacijos paslaugas ir specialistus, kurie rūpinsis jumis po sunkių ligų, operacijų bei fizinės negalios atvejais. Čia gydome šalies bei užsienio šalių piliečius, kurie, dėl kvalifikuotos, šiuolaikiškos, Europinius reikalavimus atitinkančios medicininės pagalbos, vis dažniau renkasi XUL klinikas.

Kuo skiriasi šie du pavyzdžiai? Pabandykime juos sugretinti:

Eilės nr.	Y universiteto ligoninė ir klinikos	XUL klinikos
1	<i>veikia kaip valstijos medicinos darbuotojų mokymo centras</i>	?

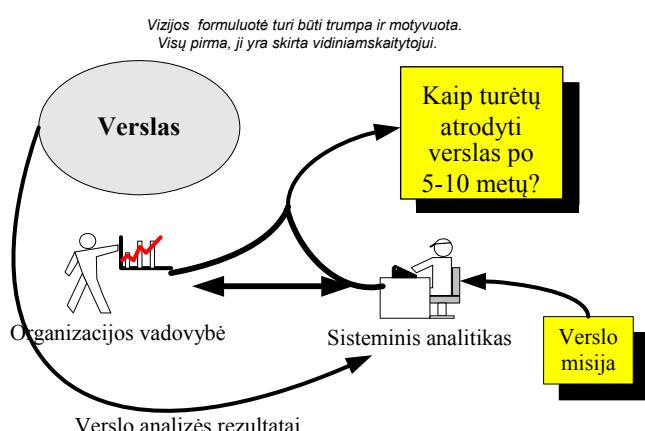
2	veikia kaip visų valstijos gyventojų (iskaitant tuos, kurie negali sumokėti) sveikatos priežiūros centras	teikia specializuotas asmens sveikatos priežiūros paslaugas (Šalies bei užsienio šalių piliečiams? Kokios paslaugos yra specializuotos?)
3	konsultuoja valstijos ir kitų valstijų sveikatos apsaugos specialistus bei organizacijas ir teikia jiems kitokią profesinę pagalbą	konsultuoja bei siunčia tolimesniam gydymui (ką? kur?)
4	teikia medicinines paslaugas savo pacientams ir valstijos neįgaliesiems	teikia būtinąjį ir planinę pagalbą (Šalies bei užsienio šalių piliečiams? ką reiškia "planinė pagalba?") bei slaugos ir reabilitacijos paslaugas (Tik savo pacientams?)
5	naudojamos studentų praktikai atlikti	vykdo pedagoginį darbą (ką moko? kaip? skaito paskaitas?)
6	atlieka inovacinius tyrimus, kuriais siekiama tobulinti sveikatos apsaugą	vykdo mokslinį tiriamąjį darbą (Koki?)

Kaip matome, abiejų organizacijų misijos yra labai panašios. Tačiau XUL klinikų misijos aprašas labai neišsamus. Be to jis "paskandintas" papildomoje informacijoje. Kita vertus, Y universiteto ligoninės ir klinikų misijoje jokios papildomos informacijos nėra.

Eilės nr.	Y universiteto ligoninė ir klinikos	XUL klinikos
1	-	yra viena didžiausių šalies ligoninių
2	-	teikia aukščiausio lygio paslaugas
3	-	pagalbą teikia aukščiausios kvalifikacijos gydytojai ir slaugytojos
4	-	konsultuoja patyrę antro ir trečio lygio specialistai-konsultantai
5	-	gydo užsienio šalių piliečius, kurie dėl kvalifikuotos, šiuolaikiškos, Europinius reikalavimus atitinkančios medicininės pagalbos, vis dažniau renkasi XUL klinikas

Ką naujo apie XUL klinikas sužinome iš papildomo teksto? Jis labai neinformatyvus. Sužinome tik, kad tai yra viena iš didžiausių šalies ligoninių, kad joje konsultacijas teikia antro ir trečio lygio specialistai (kas žino, ką tai reiškia?) ir kad ligoninėje kartais gydosi užsienio šalių piliečiai. Kodėl

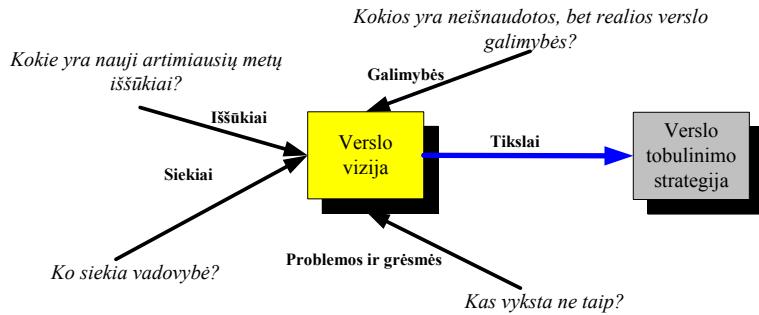
netikslinga pateikti misijoje papildomą informaciją? Visų pirma yra pažeistas *turinių atskyrimo principas* arba, kitaip tariant, informacija padėta ne į tą “lentynelę”, kur ją dera buvo padėti. Tai apsunkina informacijos skaitymą ir suvokimą. Antra, giriantis, kad ten dirba “aukščiausios kvalifikacijos gydytojai ir slaugytojos”, “konsultuoja patyrę konsultantai”, “teikiamas aukščiausio lygio paslaugos”, griaunamas solidžios organizacijos įvaizdis. Universiteto klinikose kitaip ir būti negali. Tai savaime aišku. Pateikus tokią nekonkrečią (reklaminę) informaciją, netgi jeigu ji pateikiama ne misijoje, o ten, kur ją dera pateikti, solidžios organizacijos įvaizdis griūna. O ką jau kalbėti apie tekstą “*Čia gydome šalies bei užsienio šalių piliečius, kurie, dėl kvalifiuotos, šiuolaikiškos, Europinius reikalavimus atitinkančios medicininės pagalbos, vis dažniau renkasi VUL Santariškių klinikas*” Tai paprasčiausiai provincialu. Be abejo, viso to gali nesuprasti organizacijos vadovai. Jie ne tos srities specialistai, o ir neturi laiko gilintis į tokias detales. Tačiau sisteminis analitikas, nesvarbu ar tame vaidmenyje veikia verslo konsultantas, IS inžinierius, PS inžinierius, ar dar kas nors kitas, privalo atlirkti misijos išsamumo, informatyvumo ir relevantiškumo analizę, panašią į tą, kurią atlikome analizuodami pateiktus pavyzdžius.



6 pav. Verslo vizijos formulavimas

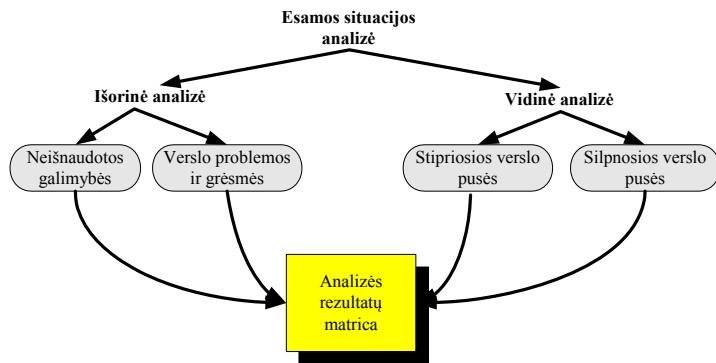
Suformulavus verslo misiją, pradedama formuluoti verslo viziją (6 pav.). Vizijos formulavimas yra vienu iš svarbiausių reikalavimų formulavimo proceso momentų. Iš verslo vizijos išplaukia kuriamos programų sistemos vizija, verslo terminais nusakanti tos sistemos paskirtį ir jos kūrimo tikslus. Todėl verslo vizijos klaidos yra perkeliamos į informacinę sistemą ir į ją palaikančią programų sistemą ir, jeigu jos pastebėtos tik pradėjus naudotis jau sukurtą programų sistemą, jas pašalinti yra labai sunku ir brangu arba netgi apskritai neįmanoma. Taigi, vizijos formulavimas yra labai atsakingas, sudėtingas ir daug darbo reikalaujantis procesas. Jis susideda iš dviejų dalių. Visų pirma, reikia išsiaiškinti, kiek sėkmingai yra įgyvendinama verslo misija ir arba pašalinti ją įgyvendinti trukdančias problemas, arba keisti pačią misiją. Antra, reikia įvertinti prognozuojamus išorinės aplinkos pokyčius per artimiausius 5-10 metų ir nuspresti, kaip verslas į juos turi reaguoti. Be to, dar yra neišnaudotos verslo galimybės ir organizacijos vadovybės ambicijos. Visa tai irgi turi būti įvertinta (7 pav.). Be abejo, bet kuriuo atveju vizija turi būti reali ir pagrįsta išsamios analizės rezultatais, o ne grindžiama vien tik tuščiomis fantazijomis ir nerealiomis ambicijomis. Tačiau sisteminis analitikas gali tik formaliai analizuoti ir vertinti viziją.

Pagrindinis vaidmuo ją formulujant turi tekti organizacijos vadovybei, kuri, be abejo, turėtų naudotis verslo konsultantų paslaugomis.



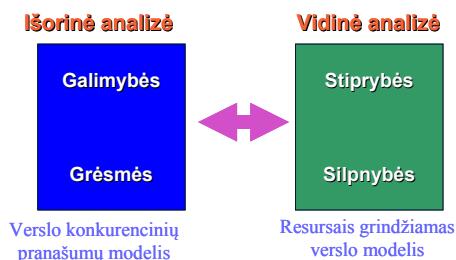
7 pav. Verslo vizijos ir verslo tobulinimo strategijos formulavimas

Vizijos formulavimas pradedamas strategine esamos situacijos analize (8 pav.). Tokia analize yra vadinama verslo stipriųjų ir silpnųjų pusiu, grėsmių ir galimybių analize (anglų kalboje vartojuamas terminas *SWOT analysis*). Ši analize apima tiek verslo išorinių, tiek ir jo vidinių veiksnių analizę. Išoriniai veiksnių yra analizuojami siekiant išryškinti verslo problemas, grėsmes ir neišnaudotas galimybes. Prie to dar reikia pridėti iššūkius, tačiau juos reikia nustatyti kitais būdais. Analizuojant tik išorinę verslo aplinką jų nustatyti neįmanoma. Vidiniai veiksnių yra analizuojami siekiant išryškinti verslo stipriasių ir silpnasių puses. Analizės rezultatų pagrindu yra formuluoja verslo vizija. Bendroji analizės schema parodyta 8 pav. Be abejo, formulujant viziją dar reikia atsižvelgti į organizacijos vadovybės strateginius siekius ir, savaimė suprantama, vizija turi būti tos vadovybės aprobuota.



8 pav. Esamos situacijos analizė

Aptarkime išsamiau, kaip vykdoma verslo išorinių ir vidinių veiksnių analizė ir kaip yra išryškinami tie veiksnių. Visų pirmiai reikia pastebeti, kad išoriniai ir vidiniai verslo veiksnių tarpusavyje yra tampriai susiję (9 pav.). Verslo problemas ir grėsmės dažniausiai didžiaja dalimi kyla iš jo silpnųjų pusiu. Kita vertus, dažniausiai yra įmanoma išnaudoti tik tas naujas verslo galimybes, kurias gali palaikyti stipriosios verslo pusės.



9 pav. Išorinės ir vidinės analizės rezultatų sugretinimas

Neišnaudotomis verslo galimybėmis vadinami išoriniai verslo aplinkos veiksniai, potencialiai padedantys greičiau ar efektyviau įgyvendinti verslo misiją. Neišnaudotomis galimybėmis gali būti kokia nors neužimta verslo niša (t. y. klientai, kurių poreikių niekas netenkina), galimybės, atsiradusios dėl įstatymų liberalizavimo ar muitų barjerų panaikinimo (pvz., Lietuvai ištojus į Europos Sąjungą), bei galimybės pasinaudoti versle kokiomis nors naujomis technologijomis, išskaitant ir informacines technologijas. Mūsų kurso kontekste svarbiausios yra būtent pastarosios galimybės. Išorinės analizės metu reikia išryškinti ir tokias galimybes, kuriomis verslas jau iš dalies ar netgi visai naudojasi. Tai gali būti, pavyzdžiui, gera verslo reputacija, pripažintas brendas, geroje vietoje esanti verslo būstinė, turimi patentai ar išskirtinės naudojimosi pardavimų tinklais sąlygos. Analizės metu reikia išsiaiškinti, kokiui mastui yra tokiomis galimybėmis pasinaudojama ir ar čia kas nors yra tobulinta.

Verslo problemomis bei grėsmėmis vadinami verslo aplinkos veiksniai realiai ar potencialiai trukdantys tą misiją įgyvendinti. Skirtumas tarp problemų ir grėsmių yra tas, kad problemos jau realizavosi, o grėsmės su tam tikra tikimybe dar tik gali realizuotis ir ta tikimybė yra pakankamai didelė. Galima išskirti problemas bei grėsmes, sąlygotas nuo verslo nepriklausančių veiksnių, ir tas, kurias sąlygoja verslo procesų organizavimo bei vykdymo ypatumai. Pirmajai grupei, pavyzdžiui, priskirtini sunkumai, su kuriais susiduria verslas staiga pasikeitus madoms ir dėl to sumažėjus verslo teikiamų paslaugų ar jo parduodamų gaminių paklausai, sugriežtėjus įstatymams, atsiradus rinkoje pigesniems teikiamų paslaugų ar gaminamų gaminių pakaitalam, staiga pabrangus žaliaivoms ar padidėjus mokesčiams. Formuluojant verslo viziją, tenkama atsižvelgti ne tik į esamas grėsmes, bet ir tas, kurios gali kilti artimiausių 5-10 metų laikotarpyje. Jos vadinamos verslo iššūkiais. Kadangi verslo grėsmių ir iššūkių prigimtis yra ta pati, toliau mes jų nebeskirime. Taipogi neaptarinėsime, kaip išryškinti tokio pobūdžio problemas bei grėsmes. Tai marketingo bei prognozavimo uždaviniai, kurių nagrinėjimas išeina už mūsų kurso ribų. Visą dėmesį sutelksime į antrają problemą bei grėsmių grupę, kuriai priklauso per didelę teikiamų paslaugų ar gaminamų gaminių savikaina, prasta jų kokybė, per ilgas gamybos ciklas ir kitos panašaus pobūdžio problemos. Šiai grupei priskirtini ir tokie dalykai, kaip prasta verslo reputacija, patentų ar brendų nebuvimas, netinkamoje vietoje esanti verslo būstinė ar neturėjimas galimybų naudotis svarbiausiuju pardavimo tinklu paslaugomis.

Stipriosioms verslo pusėms priskiriami jo unikalūs gebėjimai, kitiems nežinomas technologijos, leidžiančios ką nors padaryti pigiau, greičiau ar geriau, negu tai gali padaryti kiti, bet taip pat ir darbuotojų kompetencija, jų bendras išprusimas, sukauptos žnios bei patirtis. Silpnosioms verslo pusėms priskiriamos pasenusios technologijos, išskaitant ir nepatikimas bei ilgai trunkančias rankinio duomenų apdorojimo procedūras, netinkamas darbo organizavimas, darbuotojų kompetencijos ar bendro išprusimo stoka, reikiamos informacijos stygius ir kiti panašūs dalykai. Kartais tą patį dalyką galima traktuoti ir kaip stiprią, ir kaip silpną verslo pusę. Pavyzdžiui, jeigu programinę įrangą kurianti bendrovė turi didelį skaičių darbuotojų, gerai išmanančių kokią nors konkrečią programų sistemų kūrimo technologiją, ir kiekvienam iš jų yra įrengta visa reikiama techninė bei programinė įranga aprūpinta brangiai kainuojanti darbo vieta, tai yra neabejotinas konkurencinis verslo pranašumas, lyginant su tomis bendrovėmis, kurios viso to neturi. Tačiau, technologijai pasenus, tai tampa silpną verslo pusę, nes tokiai organizacijai pereiti prie naujos technologijos kainuoja daug daugiau, negu bendrovei turinčiai, tarkime, daug mažesnį darbuotojų skaičių. Turint didelius gamybinius pajėgumus, yra daug

sunkiau juos pertvarkyti ir todėl reaguoti į naujus rinkos iššūkius negalima taip greitai, kaip tą gali padaryti tokį pajėgumą neturintys konkurentai.

Atliktos analizės rezultatai aprašomi vadinamaja *verslo analizės rezultatų matrica* (anglų kalba ši matrica vadinama SWOT matrica).

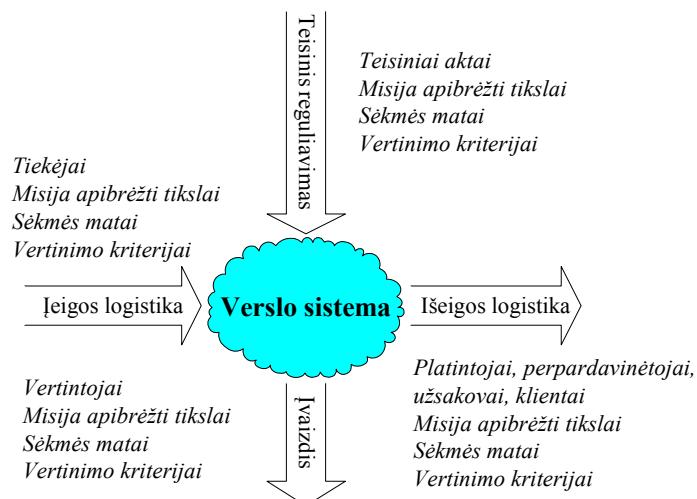
Pavyzdys

Verslo analizės matricos fragmento pavyzdys pateiktas 10 pav.

Stipriosios pusės	Silpnosios pusės
<p>Didžiausia Lietuvos ligoninė, turinti moderniausią šalyje medicininę aparatūrą, kurios pakanka visiems pacientų poreikiams tenkinti.</p> <p>Ligoninėje sutelkti geriausi Lietuvos širdies ligų gydytojai.</p>	<p>Didelės eilės registratūroje.</p> <p>Gydytojai beveik trečdalį savo darbo laiko gaišta įvairiems dokumentams pildyti.</p> <p>Trūksta kvalifikuoto aptarnaujančio personalo.</p>
Galimybės	Problemos ir grėsmės
<p>IT panaudojimas pacientų registravimui supaprastinti ir pagreitinti.</p> <p>Gydytojų raštvedybros kompiuterizavimas.</p>	<p>Maži atlyginimai, specialistų emigracija, perejimas į privačias medicinos įstaigas.</p> <p>Neprotingai vykdomos medicinos reformos primesti ribojimai.</p>

10 pav. Verslo analizės rezultatų matricos pavyzdys

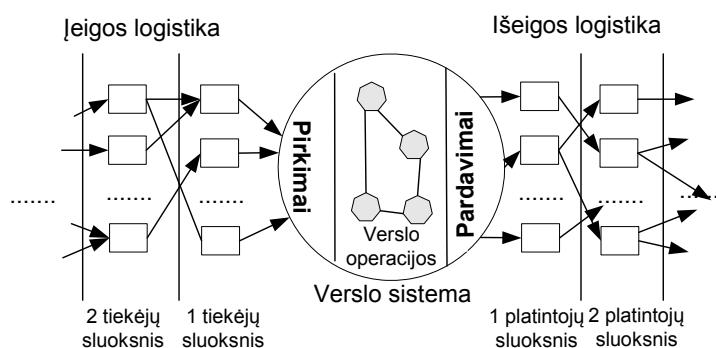
Verslo stipriųjų ir silpnųjų pusiai, grėsmių ir galimybių analizė yra palyginti paprastas ir nesunkiai panaudojamas instrumentas, padedantis atlikti esamos verslo situacijos strateginę apžvalgą. Tačiau, kita vertus, verslo vizijai suformuluoti šios analizės rezultatai dar nėra pakankamai išsamūs ir konkretūs. Tai tik išeities duomenys išsamesnei verslo išorinei ir vidinei analizei atlikti.



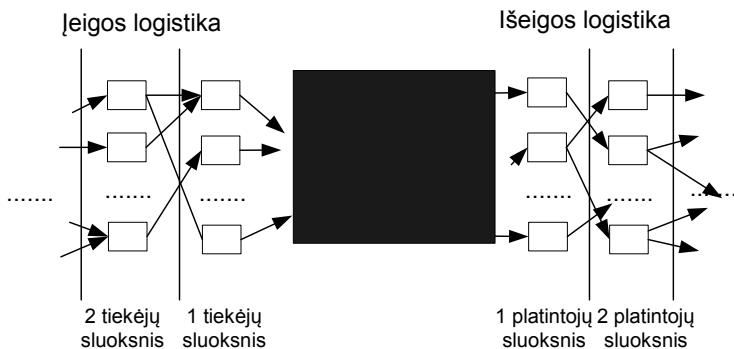
11 pav. Svarbiausieji verslo proceso sąveikos su išorine aplinka aspektai

Verslo sistemos turi daug įvairių sąveikos taškų su išore. Tačiau galima išskirti keturis svarbiausius sąveikos su išorine aplinka aspektus (11 pav.). Tai jeigos logistika, išeigos logistika, teisinis verslo reguliavimas ir verslo įvaizdis. Išorinė analizė turi apimti bent jau šiuos keturis aspektus ir išryškinti čia kylančias problemas bei grėsmes, t. y. jeigos logistikos problemas ir grėsmes, išeigos logistikos problemas ir grėsmes, teisinio reguliavimo problemas ir grėsmes bei įvaizdžio problemas ir grėsmes. Kadangi pagal apibrėžti verslo problemos ir grėsmės yra tai, kas trukdo įgyvendinti verslo misiją, tai visų pirma reikia išsiaiškinti, kokių tikslų yra siekiama pagal kiekvieną iš nagrinėjamų aspektų. Siekiami tikslai gali būti suformuluoti misijoje išreikštiniu būdu arba iš jos išvedami. Pavyzdžiui, nagrinėjant Y universiteto ligoninės ir klinikų misiją, matome, kad joje parašyta, jog ši organizacija veikia sutinkamai su valstijos įstatymais, bet ten nėra nei žodžio apie jeigos logistiką ar įvaizdį. Su šiais aspektais siejami tikslai turi būti išvedami iš misijos ir apie tai reikia galvoti formuluojant misiją. Bendruoju atveju, tikslai yra kokybiniai, bet jų įgyvendinamumo laipsnis turi būti pamatuojamas. Taigi, norint išsiaiškinti ar tikslai yra sėkmingai įgyvendinami, reikia turėti matų (jie vadinami *verslo sėkmės matais*) sistemą, panaudojant kurią būtų galima atliliki tokius matavimus. Matavimų rezultatus reikia įvertinti, t. y. reikia nuspręsti, ką mes turime: normalią situaciją, problemą ar grėsmę. Tam reikalingi matavimo rezultatų vertinimo kriterijai.

Pirmiausiai panagrinékime jeigos ir išeigos logistikos analizės klausimus (12 pav.). Logistika susijusi su verslui reikalingu išteklių, iškaitant žaliavas, įsigijimo ir verslo teikiamų paslaugų ar jo gaminamų gaminių (produktų) pardavimu. Dažniausiai, logistikos grandinė yra daugiasluoksnė, t. y. reikalingi ištekliai yra įsigyjami per keletą tarpininkų ir paslaugos bei produktai taip pat pasiekia galutinius klientus ar pirkėjus per kelis tarpininkus (prekybos tinklus, platintojus, perpardavinėtojus ir kt.). Siekiant išsiaiškinti verslo logistikos problemas ar logistines grėsmes, paprastai tenka analizuoti visą logistikos grandinę. Atliekant tokią analizę taikomas juodosios dėžės principas, t. y. analizuojama tik verslo sistemos išorė, o pati verslo sistema yra traktuojamas kaip juodoji dėžė (13 pav.). Priminsime, kad juodaja dėžė yra vadinama tokia sistema, apie kurią žinoma jos įeiga, išeiga ir išeigos priklausomybės nuo jeigos, bet nieko nėra žinoma apie pačios sistemos vidinę struktūrą ar veikimą.



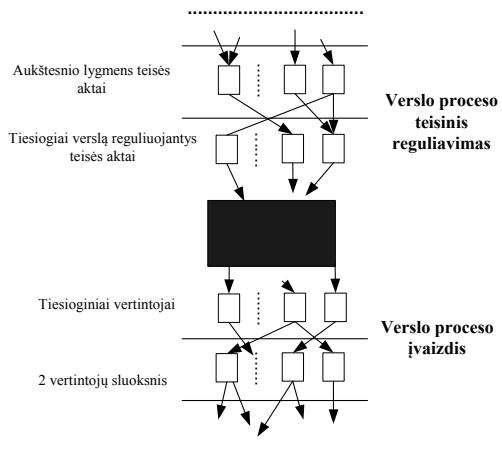
12 pav. Verslo įeigos ir išeigos logistika



13 pav. Juodosios dėžės principio taikymas verslo logistikos analizēje

Kokie galėtų būti iš verslo misijos išplaukiantys su įeigos logistika susiję verslo tikslai? Panagrinėkime keletą pavyzdžių. Natūralu, kad verslas visa, ko jam reikia, siekia įsigytį laiku, mažiausiomis kainomis ir bent jau priimtinosis kokybės. Aišku, tam tikrais atvejais kokybė gali būti svarbiau negu kaina. Galimi ir kitokie atvejai. Pavyzdžiu, verslas gali turėti galimybes sandėliuoti pakankamai dideles įsigijamų išteklių apimtis ir ritmingas tiekimas jam gali būti nelabai svarbus. Natūralu taip pat, kad paprastai verslas nenori priklausyti nuo vieno tiekėjo ir siekia turėti kelis alternatyvius tiekėjus, kurie tarpusavyje nuolat varžytasi. Kita vertus, verslui yra svarbus tiekėjų patikimumas. Jis taip pat siekia išlaikyti jau nusistovėjusias tiekimo grandines. Gali būti keliami ir subtilesni tikslai. Pavyzdžiu, verslui gali būti svarbu, kad pastatų, įrengimų, mašinų ar kito didelių kapitalinių idėjimų reikalaujančio turto įsigijimo procedūros būtų greitos ir paprastos, kad jam nereikėtų tam tikslui gaišti daug laiko, samdytis juristus, konsultantus ar turėti kitas papildomas išlaidas.

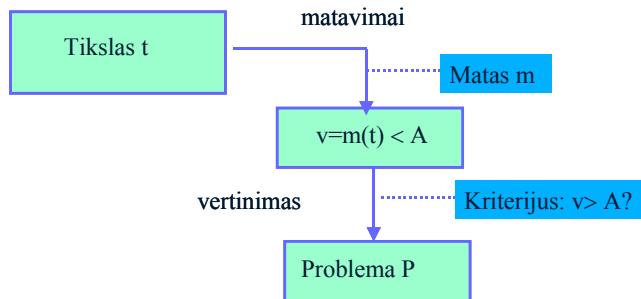
Panašiai apibrėžiami ir su išeigos logistika susiję verslo tikslai. Pavyzdžiu, gali būti norima turėti alternatyvius platinimo kanalus, siekiama garantijų, kad gaminiai ar teikiamas paslaugos klientus pasieks visuomet laiku ir kad tarpininkai parduos gaminius ar paslaugas konkurencingomis kainomis. Gali būti siekiama minimizuoti gaminių ir paslaugų gamybos bei pardavimo kaštus ar jų gamybos trukmę; optimizuoti gatavų gaminių sandėliavimo procedūras ar gaminių ir paslaugų pateikties klientams procedūras ir pan. Šie tikslai yra išvestiniai. Tiesiogiai misijoje formuluojamų tikslų pavyzdžiais gali būti siekis tenkinti visų potencialių klientų grupių poreikius arba dominuoti visuose rinkos segmentuose, kuriuose yra konkuruojama su varžovais.



14 pav. Juodosios dėžės principio taikymas analizuojant verslo teisinio reguliavimo ir verslo įvaizdžio problemas

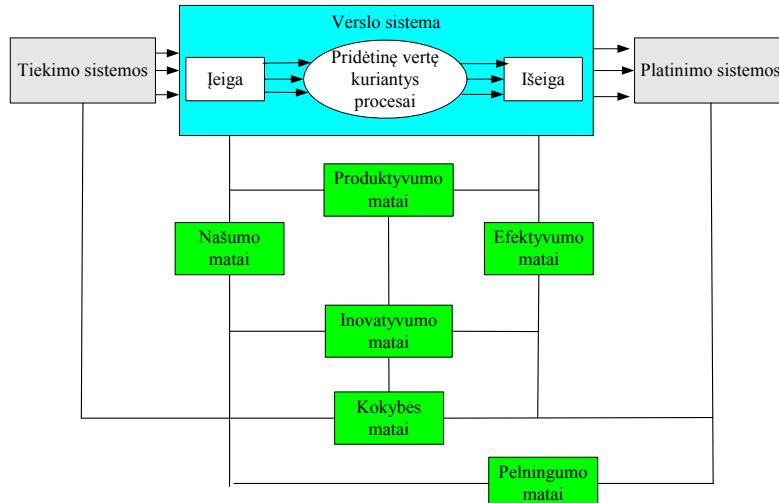
Verslo sistemos sąveika su kitais dviem išoriniais aspektais, įvaizdžio ir teisinio reguliavimo aspektais, analizuojama irgi panašiai. Verslo sistema traktuojama kaip juodoji dėžė, o įvaizdžio ir teisino reguliavimo aspektai yra išskaidomi į *poveikio sluoksnius*. Skaidant į sluoksnius įvaizdžio aspektą, visų pirma išryškinami vertinantieji subjektai, o po to jie grupuojami į pirminius (turinčius tiesioginę sąveiką su verslu), antrinius, t. y. tiesiogiai su verslu nebendraujančius ir vertinančius jį tik pagal juos pasiekiančią informaciją, pavyzdžiu, televizijos laidas ar spaudoje skelbiamus straipsnius, ir t.t. Kiek vertinimo sluoksnių tikslinga aprépti, priklauso nuo verslo pobūdžio ir jo misijos. Gali būti, kad svarbus tik konkrečiame mieste ar miestelyje turimas įvaizdis, o gali prireikti formuoti įvaizdį visos Lietuvos, Europos Sąjungos ar netgi viso pasaulio mastu. Be abejo, įvaizdžio formavimas yra savarankiška veiklos sritis, ja užsiima specialistai ir mes į šią tematiką per daug nesigilinsime. Tačiau neretai įvaizdžiui formuoti gali prireikti specialistai tam skirtų interneto svetainių, portalų ar kitokių programų sistemų. Todėl sisteminio analitiko vaidmenyje veikiantys asmenys (IS inžinieriai, PS inžinieriai ir pan.) neturi ignoruoti šio verslo proceso sąveikos su išorine aplinka aspekto ir, jeigu reikia, kartu su įvaizdžio formavimo specialistais, išsiaiškinti ir suformuluoti, kokių tikslų verslas siekia šioje srityje. Kokie gi galėtų būti su įvaizdžiu susiję verslo tikslai? Gali būti siekiama susikurti solidžios organizacijos ar patikimo verslo partnerio reputaciją; arba modernios šiuolaikinės bendrovės, kuri pirmoji pradeda pardavinėti madingus gaminius ar net kuria madas savo veiklos srityje įvaizdį. Taip pat, gali būti norima išpopuliarinti savo brendus, siekti, jog apie verslą būtų galvojama kaip apie aukštос kokybės paslaugas teikiantį ar aukštос kokybės gaminijus gaminantį verslą, kuris visų pirma vadovaujasi savo klientų ar pirkėjų interesais. Be abejo, beveik kiekvienas verslas taip pat nori užsitikrinti gerus santykius su visais įtaką verslo plėtrai darančiais subjektais, išskaitant valdymo institucijas ir įtakingas interesų grupes. Be abejo, su įvaizdžiu susiję tikslai negali būti atsieti nuo išeigos logistikos tikslų. Vien tik modernizuojant reklamą ar kuriant interneto svetaines, ilgalaikio įvaizdžio sukurti neįmanoma. Tam turi būti pajungti visi verslo proceso aspektai. Tačiau, kita vertus, būtų taip pat netikslinga įvaizdį kurti ignoruojant informacinių technologijų teikiamas galimybes.

Su teisiniu reguliavimu susiję verslo tikslai taip pat gali būti labai įvairialypiai ir saviti kiekvienam verslui. Be abejo, kiekvienas sąžiningas verslas siekia nepažeidinėti įstatymų ir išvengti su tuo susijusių baudų bei kitokiu nemalonumu. Kartais tai gali būti pabrėžta ir pačioje verslo misijoje. Pavyzdžiu, mūsų nagrinėtame Y universiteto ligoninės ir klinikų misijos pavyzdyme buvo pasakyta, kad šios įstaigos veikia sutinkamai su valstijos įstatymais. Tačiau šito dar nepakanka, nes reikia išsiaiškinti, kokius konkretius teisės aktus ir kokiu būdu gali savo veikla pažeisti šios įstaigos. Šitokia teisinė analizė turi būti atlikta bet kuriam verslo procesui. Be abejo, paprastai organizacijos vadovybė gali ne tik išvardinti atitinkamus teisės aktus, bet ir apibūdinti problemas bei grėsmes, su kuriomis ji susiduria šioje srityje. Kita vertus, dažnai verslo siekiai esti gerokai sudėtingesni, t. y. vien tik nepažeidinėti verslą reguliuojančių teisės aktų dar nepakanka. Pavyzdžiu, gali būti siekiama taip organizuoti verslą, kad aplinkosaugos įstatymų įtaką gamybos kaštams būtų sumažinta iki minimumo ar minimizuojami įvairūs su paslaugų ar gaminių pardavimu susiję mokesčiai ir šitaip būtų mažinami pardavimo kaštai. Verslo paslapčių ir duomenų apsauga taip pat yra priskirama prie tikslų, susijusių su teisiniu reguliavimu, nors šie tikslai galėtų būti nagrinėjami ir kaip savarankiškas verslo sistemos sąveikos su išore aspektas.



15 pav. Verslo tikslų įgyvendinimo laipsnio matavimas

Norint nustatyti, kiek sėkmingai yra įgyvendinami su logistika, įvaizdžiu bei teisiniu reguliavimu siejami verslo tikslai, reikia apibrėžti matus tų tikslų įgyvendinamumui pamatuoti (kaip jau minėjome, tokie matai yra vadinami *verslo sėkmės matais*) ir kriterijus matavimų rezultatams vertinti (15 pav.). Kitaip tariant, verslo problemos ir grėsmės dažniausiai yra aptinkamos atliekant verslo tikslų įgyvendinimo laipsnio matavimus ir vertinimus (15 pav.). Taigi, atliekant matavimus, stengiamasi atsakyti į klausimus, kokių mastų yra pasiekiami verslo tikslai ir ar galima teigti, jog verslas vyksta sėkmingai. Su kiekvienu tikslu turi būti susietas bent vienas sėkmės matas, tačiau dažniausiai jų yra keletas. Su kiekvienu tikslu taip pat turi būti susietas vertinimo kriterijus, tiksliau, reitingavimo funkcija, atvaizduojanti matavimų rezultatus į vieną iš lingvistinio kintamojo reikšmių {normalu, grėsmė, problema}. Šie vertinimai apima ne tik pačios verslo sistemos, bet ir jo logistikos partnerių veiklos vertinimus (16 pav.), išskaitant jo sėkmę ar nesėkmę įvaizdžio formavimo ir sąveikos su teisinio reguliavimo sistemomis srityse.



16 pav. Verslo sistemos matavimai

Pavyzdys

Verslo sėkmės matų pavyzdžiai pateikti 17 pav., o matavimų vertinimo pavyzdžiai – 18 pav.

Kas matuojama	Apibrėžtis	Formulė	Matų pavyzdžiai
Pelningumas	Iplaukų ir kaštų savykis	išeiga/jeiga	iplaukos/kaštai pelnas/kaštai
Produktyvumas	Išeigos ir sunaudotų išteklių savykis	išeiga/jeiga	
Efektyvumas	Tikslų įgyvendinimo laipsnis	gauta išeiga/planuota išeiga	faktinė gamybos apimtis per laiko vieną/planuota gamybos apimtis per laiko vieną
Našumas	Tikslingo išteklių panaudojimo laipsnis	planuoti ištekliai/ sunaudoti ištekliai	planuotos kompiuterinio laiko sąnaudos/faktinės kompiuterinio laiko sąnaudos

17 pav. Verslo sėkmės matų pavyzdžiai

Kas matuojama	Matavimo rezultatai	Vertinimo kriterijus	Vertinimas
Efektyvumas	80 %	$\geq 80\%$	Grėsmė
Našumas	75%	$\geq 80\%$	Problema

18 pav. Verslo sėkmės matavimų vertinimo pavyzdžiai

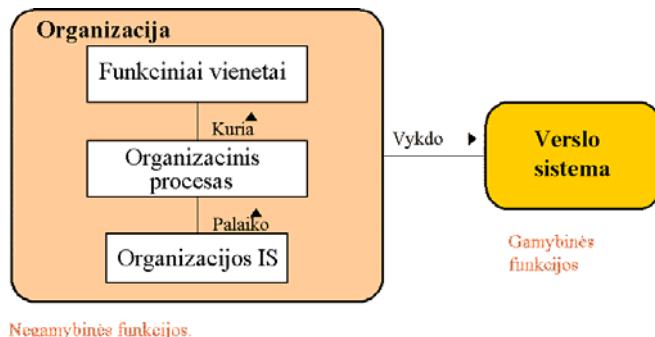
■.

Išryškinus verslo grėsmes, problemas, neišnaudotas galimybes ir stipriasių pusės, pereinama prie vidinės verslo sistemos analizės. Nors kai kurios verslo problemos ir grėsmės gali būti salygotos grynai išorinių veiksnių (pvz., papildomų mokesčių įvedimas), didžioji jų dalys kyla iš pačios verslo sistemos trūkumų. Taigi, reikia pasistengti išsiaiškinti, kokios priežastys salygoja nepakankamą verslo sistemos pelningumą, produktyvumą, našumą, efektyvumą ar kitas iš išorės stebimas problemas bei grėsmes. Savaime aišku, kad tai irgi yra silpnosios verslo pusės, nors galbūt pradinės analizės metu jos ir nebuvovo išryškintos. Be to, netgi jei jos ir buvo išryškintos, jas reikia išanalizuoti daug detaliau negu tai buvo padaryta pradinės analizės metu. Kita vertus, norint pasinaudoti kokiomis nors dar neišnaudotomis galimybėmis, reikia suvokti kiek verslas yra pasirengęs jomis pasinaudoti ir ar tai apskritai yra įmanoma. Visa tai yra verslo sistemos vidinės analizės uždaviniai.

Taigi, pradedant vidinę verslo sistemos analizę, reikia atidaryti juodają dėžę (12 pav.) ir pažiūrėti kas gi yra tos dėžės viduje. Be abejo, tai metafora. Iš tiesų, reikia sukonstruoti verslo sistemos modelį ir, analizuojant operacijų grandines, perdirbančias verslo sistemos įeigą į jos išeigą, išsiaiškinti, kur slypi verslo problemų bei grėsmių priežastys ir kokių mastu sistema yra parengta naujomis galimybėmis diegti. Tai vadinama *vertės ir logistikos grandinių analize*. Tai reiškia, kad turi būti atliekama ne tik pačios verslo sistemos, bet ir verslo logistikos sistemų (t. y. tiekimo ir platinimo sistemų) vidinė analizė. Verslo sistemoje yra analizuojamos ne tik pagrindinės vertės grandinės, bet ir pirkimų, marketingo, pardavimų, žmogiškųjų išteklių vadybos, technologinės plėtros bei infrastruktūros kūrimo ir palaikymo veiklos. Kitaip tariant, reikia analizuoti ne tik bazines, bet ir organizacines bei pagalbinės veiklas. Analizė pradedama bazinėmis veiklomis, judama pridėtinę vertę kuriančia grandine ir analizuojama, ar visos atliekamos operacijos tikrai yra reikalingos, kokie yra jų

našumas, savikaina bei kiti parametrai. Pagalbinėms ir organizacinėms veikloms žiūrima, ar jos tinkamai palaiko bazines veiklas. Šitaip ieškoma išorinės analizės metu rastų problemų bei grėsmių priežasčių. Panašiai analizuojamos ir logistikos grandinės. Vertės ir logistikos grandinių analizė yra pagrindinis vidinės analizės metodas, užduodantis bendrają analizės schemą. Tačiau analizei atliliki yra naudojami labiau specializuoti papildomi metodai. Tai verslo funkcijų analizė, verslo procesų analizė, verslo užduočių analizė, verslo ištaklių auditas, kaštų ir pelno analizė ir lyginamoji analizė. Šie metodai yra tarsi vertės grandinių analizės instrumentai, padedantys aptikti jose esančius nesklandumus bei trūkumus.

Verslo sistemos funkcijomis vadinamos nuosekliai susijusių operacijų grandinės, manipuliuojančios kokia nors viena ar keliomis verslo esybėmis ir tiesiogiai ar netiesiogiai prisidedančios prie vieno ar kelių verslo tikslų įgyvendinimo. Verslo funkcija visuomet duoda organizacijai kokių nors iplaukų. Ji gali arba kurti kokius nors gaminius ar paslaugas, arba valdyti, administruoti, stebėti, fiksuoti ar aprašyti kokias nors organizacijoje vykdomas veiklas, jų būsenas arba kokių nors verslo esybių naudojimo ar veikimo sąlygas. Funkcija yra apibrėžiama veiklų, atsakomybių ir atskaitingumo terminais. Ji turi būti įvardinama ir nusakoma, bet nebūtinai pamatuojama. Paprastai funkcija apima ištisą organizacijos veiklos ar valdymo barą ir gali būti panaudota skirtingoje verslo srityse. Kita vertus, ji gali apimti vieną arba daugiau priklausomų ar nepriklausomų veiklų. Verslo funkcija gali būti sudaryta iš smulkesnių funkcijų. Ją gali vykdyti vienas asmuo, grupė asmenų, padalinys, padalinį grupę ar netgi visa organizacija. Funkcijos vykdytojas yra vadinamas *funkciniu vienetu*. Tipiškas verslo funkcijos pavyzdys yra verslo sistemos kuriamų gaminių bei paslaugų pardavimas. Beje, šiuo atveju pardavimu vadinama tai, ką daro organizacijos pardavimų skyrius (aišku, jei organizacijoje toks skyrius apskritai yra), o ne tai kuo užsiima partneriai, priklausantys išeigos logistikos grandinėms. Bet kurios organizacijos funkcijas galima suskirstyti į gamybines ir negamybines (19 pav.). Gamybinės funkcijos apima veiklas, tiesiogiai susijusias su gamyba, paslaugų teikimu, pelno gavimu arba tokią veiklą valdymu. Gamyba šiuo atveju suprantama labai plačiai ir, pavyzdžiui, mokymo institucijose yra tapatinama su mokymu. Kitaip tariant, gamyba mes vadiname veiklas, kuriomis įgyvendinama verslo sistemos misija. Negamybinės funkcijos apima veiklas, skirtas organizacijos kaip juridinio asmens kasdieniniams darbui organizuoti. Tokių funkcijų pavyzdžiais yra administravimo, apskaitos ir kontrolės funkcijos. Verslo sistemos karkasą sudaro *funkcinės linijos*, apimančios keletą valdymo hierarchijos lygmenų. Kiekviename lygmenyje yra vykdoma ta pati arba artimai su ja susijusios veiklos ir visi lygmenys už savo darbo rezultatus atskaito tuo pat momentu, t. y. tame pačiame funkcinės linijos taške.



19 pav. Gamybinės ir negamybinės verslo funkcijos

Kiekvienai verslo funkcijai reikia nustatyti jos jeigu bei išeigą, apibrėžti sąryšius (materialūs srautai, finansiniai srautai, duomenų srautai, dokumentų mainai ir pan.) su kitomis funkcijomis ir atlikti jos pelningumo, našumo bei kitų savybių vertinimus. Jei tai įmanoma, vertinimai atliekami vadovaujantis atitinkamų matavimų rezultatais. Šitaip verslo problemos bei grėsmės yra lokalizuojamos verslo funkcijose ir nuleidžiamos žemyn, į verslo funkcijų lygmenį. Problemų lokalizavimu verslo funkcijose, mes vadiname problemų susiejimą su verslo funkcijomis arba, kitaip tariant, nustatymą, kurios verslo funkcijos kurias problemas pagimdo. Aišku, kad kokia nors verslo problema gali susidaryti dėl kelių žemesnio lygmens problemų, su kuriomis yra susidurama vienoje ar keliose verslo funkcijose. Taigi, konkreti verslo problema gali būti lokalizuota vienoje ar daugiau verslo funkcijų. Problemų nuleidimu žemyn mes vadiname verslo problemos pakeitimą ją gimdančiomis funkcijų lygmens problemomis. Kalbėdamasis su dalykinės sritys specialistais, analitikas aiškinasi, kaip seniai tos problemos atsirado, ar bandyta jas spręsti ir kaip tai buvo daroma. Jis taip pat renka nuomonės apie galimus išryškintų problemų sprendimo būdus. Vėliau tos nuomonės bus panaudotos formuluojant verslo viziją. Reziumuojant, galima pasakyti, kad analizuojant verslo funkcijas reikia suformuluoti atsakymus į šiuos klausimus:

- Kokie funkciniai vienetai kokias verslo funkcijas vykdo? Kiek darbuotojų yra susiję su kiekvienos verslo funkcijos vykdymu?
- Koks yra vidutinis tų darbuotojų darbo stažas? Kokia yra jų kvalifikacija ir profesinė patirtis? Kiek jie prisideda prie verslo sistemos reputacijos ir įvaizdžio formavimo (ar kontaktuoja su klientais partneriais ir pan.)? Ar yra tam parengti ir suvokia bendravimo kultūros svarbą? Ar yra supažindinti su teisės aktais, reguliuojančiais jų veiklą? Kaip dažnai darbuotojai keičiasi ir kokios yra tos kaitos priežastys? Ar auga su funkcijos vykdymu susijusių darbuotojų skaičius, kaip sparčiai tai vyksta ir kodėl?
- Ką konkrečiai daro kiekvienas darbuotojas? Kodėl darbuotojai daro tai, ką jie daro? Kada darbuotojai daro tai, ką jie daro? Kur darbuotojai daro tai, ką jie daro? Kam darbuotojai daro tai, ką jie daro ir kas konkrečiai naudojasi jų darbo rezultatais? Ką daro tiems darbuotojams vadovaujantys asmenys?
- Su kokiomis problemomis susiduria darbuotojai, kokios grėsmės jiems iškyla ir kokios yra viso to priežastys? Ar darbuotojai yra pajėgūs diegti išorinės analizės metu išryškintas naujas galimybes ir jomis efektyviai pasinaudoti?
- Koks darbuotojų skaičius turi personalinius kompiuterius ir jais naudojasi? Koks yra jų kompiuterinis raštingumas (jų pačių vertinimu)? Kokiai dienos darbo daliai atlikti reikalingas kompiuteris? Kokioms programomis bei duomenų bazėmis yra naudojamasi? Kokiemis tikslams kiekviena iš programų ar bazių yra naudojama? Kaip dažnai naudojamasi kiekviena iš turimų programų ar bazių? Kurios iš jų yra kritinės verslo funkcijos požiūriu? Ar yra naudojamasi kokiomis nors pagalbinėmis programomis (žodynais, antivirusinėmis programomis ir pan.)? Kokie periferiniai įrenginiai (spausdintuvas, skeneris, faksas ir kt.) yra kompiuteriuose darbo vietose? Kaip dažnai ir kokiemis tikslams jie naudojami? Kiek yra kolektyvinio naudojimo įrenginių? Ar iš darbo vietų yra prieinamos kompiuterių tinklų paslaugos ir kokios? Kokiemis tikslams jos yra naudojamos? Kaip yra apsaugotos kompiuteriuotos darbo vietas?
- Kokie yra kiekvieno darbuotojo statusas ir teisės? Kokios problemos, darbuotojų nuomone, kyla dėl jų turimų teisių, jiems nustatytais atsakomybių ir sąveikos su

kitais darbuotojais būdų? Kokios problemos kyla dėl darbuotojų gaunamų dokumentų pobūdžio, turinio ar pateikties? Kokios yra darbuotojų finansinės ir kitokios kontrolės procedūros?

- Kokios verslo taisyklės reguliuoja verslo funkcijos vykdymą ir kokie ribojimai riboja išteklių, personalo, marketingo ir vadybos metodų panaudojimą bei vykdomų darbų trukmes ir atlikimo terminus, išlaidas, investicijas, išskaitant ir investicijas potencialiai skiriamas verslo problemoms spręsti, ir kt.?

Atlikus verslo funkcijų analizę, tos išorinės analizės metu išryškintos problemos ir grėsmės, kurios susidaro dėl vienų ar kitų vidinių verslo sistemos trūkumų, yra nuleidžiamos į verslo funkcijų lygmenį. Kitaip tariant, paaiškėja, kurios verslo funkcijos prisideda prie kiekvienos verslo problemos susidarymo ir kodėl tai vyksta. Gali būti rastos ir papildomos problemos ar grėsmės, kurios dėl atlirkėtų matavimų netikslumo ar kokių nors kitų priežasčių nebuvo aptiktos išorinės analizės metu.

Baigus verslo funkcijų analizę, yra pereinama prie verslo procesų analizės. Verslo procesas – tai iš dalies sutvarkyta tarpusavyje susijusių veiklų seka. Kitaip tariant, verslo procesas gali šakotis ir turėti ciklus. Veikla – tai tarpusavyje susijusių verslo užduočių seka. Veikla negali nei šakotis, nei turėti ciklų. Verslo procese visuomet yra tiksliai žinoma, ką viena veikla perduoda kitai, o veikloje taip pat tiksliai yra žinoma, ką viena užduotis perduoda kitai. Be abejo, veikla nėra bet kokių verslo užduočių seka. Veiklą sudarančios užduotys yra išreiškiamos procedūromis ir taip susietos į vieną visumą, kad jas įvykdžius yra pasiekiamas koks nors konkretus verslo tikslas. Bendruoju atveju, tiek procesai, tiek veiklos priklauso kokiai nors verslo funkcijai. Veiklos yra valdomos duomenimis, jas inicijuoja transakcijos arba reikalavimai pateikti kokius nors duomenis. Veiklos sudaro aktyviąjį verslo funkcijos dalį, yra formalizuotos ir pakartojamos. Analizuojant verslo procesus ir veiklas yra tikslinami verslo funkcijų analizės rezultatai. Pradžioje funkcijų lygmens problemos yra lokalizuojamas verslo procesuose ir nuleidžiamos į verslo procesų lygmenį, po to – procesų lygmens problemos lokalizuojamas veiklose ir nuleidžiamos į veiklų lygmenį. Analizuojant verslo procesus ir veiklas, reikia suformuluoti atsakymus į šiuos klausimus:

- Kokių verslo tikslų yra siekiama vykdant kiekvieną verslo procesą ir kiekvieną veiklą? Kokiomis verslo nuostatomis ir verslo taisyklėmis yra vadovaujamas?
- Kokia yra kiekvieno verslo proceso ir kiekvienos veiklos vidinė struktūra?
- Kaip yra organizuotas kiekvieno verslo proceso ir kiekvienos veiklos vykdymas?
- Kaip kiekvienos verslo funkcijos procesai ir veiklos yra susiję su kitų verslo funkcijų procesais ir veiklomis?
- Su kokiomis problemomis procesų ir veiklų lygmenyse susiduria darbuotojai, vykdymasi verslo transakcijas, kokios grėsmės jiems iškyla ir kokios yra viso to priežastys?

Be to, reikia įvertinti kiekvieno verslo proceso ir kiekvienos veiklos pelningumą, našumą bei kitas savybes. Jei tai įmanoma, vertinimai atliekami vadovaujantis atitinkamų matavimų rezultatais.

Baigus verslo procesų ir veiklų analizę, pereinama prie užduočių analizės. Užduotis – tai smulkiausias diskretinis verslo sistemos elementas. Kaip ir verslo

procesai bei veiklos, užduotys yra tiksliai apibrėžtos, formalizuotos ir skirtos daugkartiniam panaudojimui. Kiekviena užduotis turi įeigą, išeigą ir apdorojančią dalį (t. y. ją realizuojančią procedūrą). Procedūros realizuoja verslo taisyklės.

Užduočių analizės metu problemos ir grėsmės yra lokalizuojamos užduotyse ir nuleidžiamos į verslo užduočių lygmenį. Tai ir yra išorinės analizės metu išryškintų verslo problemų pirminės priežastys. Jos gali būti labai įvairios. Pavyzdžiui su įeigos logistika siejamos problemos gali kilti dėl to, kad organizacija neturi patikimos tiekimų kontrolės sistemos ar patikimos inventoriaus tvarkymo sistemas, kad joje naudojamos neefektyvios žaliavų įsigijimo ir sandėliavimo procedūros arba neefektyvios darbuotojų parinkimo ir mokymo procedūros. Su išeigos logistika susijusios problemos gali kilti dėl to, kad nėra klientams patogiu parduodamos aparatūros dalių įsigijimo ir remontavimo procedūrų, numatyti jiems nepriimtinai garantiniai terminai ir gamintojo įsipareigojimai, į klientų nusiskundimus reaguojama nepakankamai greitai arba ta reakcija yra neadekvati arba gaminami gaminiai bei teikiamos paslaugos yra tobulinti nesiremiant klientų apklausomis. Tokios problemos gali kilti taip pat dėl to, kad yra naudojamos neefektyvios rinkos segmentų ir klientų poreikių analizės procedūros, organizacijoje yra per žemas informacijos apdorojimo procesų kompiuterizacijos laipsnis, prasta kokybės valdymo sistema arba joje vykdoma politika neskatina darbuotojų kūrybiškumo ir inovatyvumo, pavyzdžiui, yra netinkama darbuotojų premijavimo sistema. Kitos tipinės daugelio problemų priežastys yra silpna verslo vertės grandinės veiklų ar užduočių integracija bei prasta koordinacija. Įvaizdžio problemos gali kilti dėl kryptingos brendų populiarinimo sistemos nebuvinimo, pasenusių reklamos formų arba dėl to, kad nevedama klientų nusiskundimų registracija, neatliekama nuolatinė jų analizė ir trūksta aktualios ir tikslios informacijos strateginiams ir operacinio lygmens sprendimams priimti. Daugelis priežasčių gali slypėti netinkamose verslo nuostatose arba jas įgyvendinančiose verslo taisyklėse. Pagaliau teisinio reguliavimo problemos gali kilti dėl to, kad trūksta operatyvios informacijos apie galiojančius teisiinius aktus ir jų pakeitimus; darbuotojai yra neinformuoti apie galimas juridinių aktų pažeidimų priežastis, arba dėl to, kad informacijos apdorojimo procesai yra nepakankamai kompiuterizuoti ir, naudojant rankinio apdorojimo procedūras, nepavyksta išvengti klaidų kontroliuojančioms institucijoms pateikiamuose dokumentuose. Be abejo, tai tik keli pavyzdžiai, iliustruojantys didžiulę verslo problemų priežasčių įvairovę. Bet ir jie parodo, koks įvairiapusis ir sudėtingas yra verslo sistemos vidinės analizės procesas.

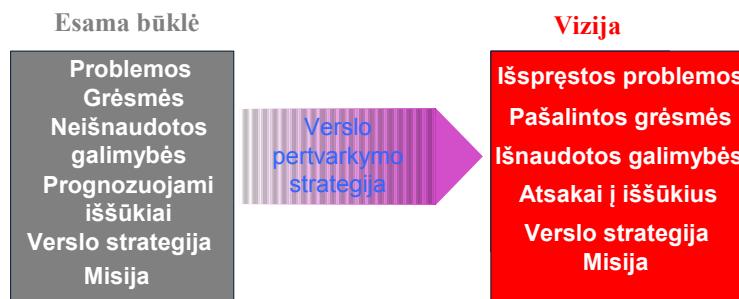
Nors mes aprašėme verslo funkcijų, verslo procesų ir veiklų bei verslo užduočių analizes kaip atskirus vienas po kito atliekamus procesus, iš tiesų visi šie procesai vyksta kartu ir tampriai susipina vienas su kitu. Su išteklių auditu, kaštų ir pelno analize bei lyginamaja analize yra šiek tiek kitaip. Tai savarankiški procesai, naudojami verslo funkcijų, procesų, veiklų ir užduočių analizės rezultatams papildyti.

Atliekant išteklių auditą, yra aiškinamasi, ar kai kurios verslo problemos bei grėsmės nėra susiję su verslo proceso naudojamų išteklių nomenklatūra, kokybe ar apimtimi. Pavyzdžiui, gali paaiškėti, kad versle dirbančių darbuotojų kvalifikacija yra per žema, kad verslas nesugeba tinkamai naudotis turimais intelektinės nuosavybės ištekliais arba kad jam trūksta apyvartinių lėšų ir dėl to kyla vienos ar kitos problemos ar grėsmės. Bendruoju atveju atliekamas fizinių, finansinių, žmogiškųjų ir abstrakciųjų išteklių auditas. Fiziniams ištekliams priskiriamos gamybos bei marketingo priemonės, įskaitant informacines technologijas. Finansinius išteklius sudaro grynieji pinigai, turimi finansiniai fondai ir verslo kreditavimo galimybės.

Abstraktiesiems ištekliams priskiriami brendai, intelektinė nuosavybė ir reputacija. Taigi, šioje vietoje vidinė ir išorinė analizės gali iš dalies persidengti, nes reputacijos auditą yra sunku atskirti nuo įvaizdžio problemų analizės.

Kaštų ir pelno analizė apima esamų pelno šaltinių ir pelno tendencijų analizę, perdirbamų apimčių analizę ir laiko sąnaudų bei darbo imlumo analizę. Nepakankamas vienų ar kitų darbų kompiuterizacijos laipsnis dažniausiai išryškėja būtent kaštų ir pelno analizės metu.

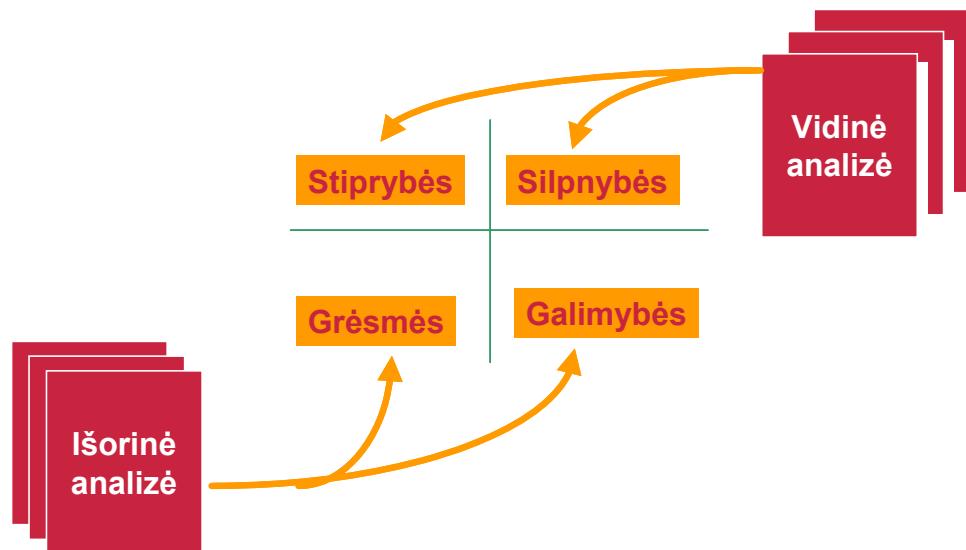
Lyginamoji analizė (angl. *benchmarking*) – tai verslo proceso operacijų našumo analizė, atliekama tikrinant, ar yra pasiekiamas vidiniai ar išoriniai standartais nustatytas tų operacijų našumo lygis, arba lyginant analizuojamo verslo proceso operacijų našumą su konkurentu ar vedančiųjų bendrovių (lyderių) atitinkamų operacijų našumu.



20 pav. Vizijos įgyvendinimas

Baigus vidinę analizę ir apibendrinus jos rezultatus, galima pereiti prie patobulinto verslo vizijos formulavimo (20 pav.). Vizija numato, kurios verslo problemos ir grėsmės turėtų būti pašalintos, kokiomis naujomis galimybėmis norima pasinaudoti, į kokius naujus prognozuojamus iššūkius ir kaip bandys atsakyti verslo sistema. Be abejo, šito dar nepakanka. Vizija turi būti formuluojama atsižvelgiant į organizacijos pasirinktą rinkos strategiją, verslo vykdymo strategiją, organizacijos funkcinio lygmens strategijas, verslo nuostatas ir verslo taisykles. Esti įvairių rinkos ir verslo vykdymo strategijų. Jos trumpai aptartos knygos priede pateiktame terminų žodyne. Nebūtinai kiekviena verslo sistema ketina plėstis ir užkariauti naujas rinkas. Galima verslo stabilizavimo ar netgi jo mažinimo strategija. Kartais vizija gali numatyti, kad rinkos ar verslo vykdymo strategijos bus keičiamos. Tai gali išspręsti kai kurias verslo problemas ar pašalinti kai kurias jam kylančias grėsmes. Tačiau, dažniausiai verslo strategija nekinta. Panašiai yra ir su misija. Vizija gali numatyti, kad misija bus keičiamā, bet dažniausiai išlieka senoji misija. Rinkos ir verslo vykdymo strategijų ar verslo misijos keitimąs yra labai sudėtingi klausimai, tačiau jie turi nedaug ką bendro su informacinių sistemų ir juolab programų sistemų inžinerijos klausimais. Tai verslo savininkų ir verslo konsulantų veiklos laukas. Tačiau šiu klausimų pamiršti negalima, nes gali įvykti taip, kad pakeitus verslo strategiją ar misiją poreikis planuojamai kurti programų sistemai apskritai dings. Pavyzdžiui, jei tarkime analizuojama verslo sistema yra zoologijos sodas ir viena iš išryškintų šio verslo problemų yra bilietų pardavimo ir tikrinimo problema, tai ši problema galėtų būti sprendžiama pastačius bilietų pardavimo automatus ir kompiuteriniu būdu valdomą lankytojų įleidimo į sodo teritoriją sistemą. Tačiau, jeigu vizija numato naują misiją, tarkime, kad zoologijos sodas bus uždarytas ir jo užimamas sklypas panaudotas gyvenamojo kvartalo statybai, aišku, tokios sistemos nebeprireiks. Šiame vadovelyje mes vadovaujamės prielaida, jog verslo misija nebus esminiai keičiamā ir

yra norima tik patobulinti esamą verslo sistemą, siekiant padaryti ją konkurencingesne. Šiek tiek kitaip yra su *verslo nuostatomis* ir *verslo taisyklemis*. Verslo nuostatos – tai nurodymai, kuriais privalu vadovautis visiems organizacijos darbuotojams, pavyzdžiu, „Su klientu visada reikia elgtis mandagiai ir pagarbiai“ arba „Nusiskundimus ir skundus reikia nagrinėti išsamiai ir skubos tvarka“. Šios nuostatos yra realizuojamos verslo taisyklemis, kurios, savo ruožtu, išreiškiamos konkrečiomis procedūromis, įtvirtinamomis pareigybiniemis instrukcijomis. Kita vertus, verslo nuostatos yra įtvirtinamos vadinamosiomis *funkcinio lygmens strategijomis*, nustatančiomis organizacijos padalinių tikslus ir veikimo būdus. Viena vertus, netinkamos verslo nuostatos ar prastos verslo taisykles gali būti viena iš verslo problemų priežasčių ir todėl verslo vizija gali numatyti jas pakeisti. Kita vertus, kai kurios verslo taisykles gali pakisti vien tik todėl, kad verslas bus kompiuterizuotas. Todėl rekomendacija, kad analizė turėtų būti pradedama verslo taisyklių išsiaiškinimu, o po to programų sistemos turėtų visas tas taisykles įgyvendinti, mūsų nuomone nėra visiškai teisinga. Aišku, verslo taisykles apima ne tik organizacijos darbuotojų darbo procedūras, bet ir daugelį kitų verslo aspektų, pavyzdžiu, įvairius ribojimus. Tokios verslo taisykles paprastai nekinta ir turi būti realizuotos verslą palaikančiose programų sistemose.



21 pav. Analizės rezultatų sugretinimas

Formuluojant viziją, patariama sugretinti stipriąsias verslo pusės su neišnaudotomis galimybėmis ir verslo problemas bei grėsmes su silpnosiomis verslo pusėmis (21 pav.). Tam naudojama vidinės analizės metu patikslinta verslo analizės rezultatų matrica (10 pav.). Nors ne visuomet pavyksta vienus su kitais sugretinti iki galo, tačiau vis vien yra labai tikėtina, kad verslo problemų bei grėsmių priežastys slypi kokiose nors jo silpnosiose pusėse. Kita vertus, ne visos silpnosios verslo pusės būtinai gimdo kokias nors problemas ar grėsmes. Tačiau vargu ar verta bandyti diegti tokias naujoves, kurios neišnaudotų stipriųjų verslo pusų ir remtusi jo silpnosiomis pusėmis. Dažniausiai šitaip elgtis yra netikslinga net ir tuomet, kuomet kokia nors naujovė žada duoti didžiausius pelnus. Prtingiau yra diegti tokias naujoves, kurias palaiko stipriosios verslo pusės, nes šitaip galima įgyti didesnius konkurencinius pranašumus. Kita vertus, galima pirmiausiai sustiprinti silpnąsias verslo pusės, šitaip paruošti verslą perspektyviom naujovėm diegti, o po to jau pereiti prie jų diegimo. Tačiau tai yra ilgalaikis procesas ir, pasirinkus tokią verslo pertvarkymo strategiją, viziją palaikančių programų sistemų kūrimą kuriam laikui tenka atidėti.

	Stipriosios pusės	Silpnosios pusės
Galimybės	Nuostata “stipriosios pusės – neišnaudotos galimybės”	Nuostata “silpnosios pusės – neišnaudotos galimybės”
Grėsmės	Nuostata “stipriosios pusės – grėsmės”	Nuostata “silpnosios pusės – grėsmės”

22 pav. Verslo analizės rezultatų siūlomos strateginės nuostatos

Taigi, teoriškai viziją galima grįsti keturių tipų strateginėmis nuostatomis (22 pav.):

- diegti naujoves gerai palaikomas stipriųjų verslo pusiu;
- diegti naujoves, palaikomas silpnųjų verslo pusiu, prieš tai tas puses sustiprinus ir šitaip parengus verslą pageidaujamoms naujovėms diegti;
- panaudoti stipriąsias verslo puses jo grėsmėms bei problemoms pašalinti;
- taip stiprinti silpnąsias verslo puses, kad dėl jų kylančios verslo problemos bei grėsmės išnyktų.



23 pav. Vizijos ištakos ir jos įgyvendinimas

Aišku, vizija retai kada yra grindžiama kuria nors viena strategine nuostata. Dažniausiai renkamasi vienas ar kitas jų derinys. Panašiai yra sprendžiama ir apie tai, kaip verslo sistema turėtų atsakyti į prognozuojamus iššūkius. Be to, formuluojant viziją reikia, aišku, jei tai yra įmanoma padaryti, pasidomėti ir tuo, kokios yra pagrindinių verslo varžovų vizijos, ar jos yra sugretinamos su formuluojama vizija ir, jei taip nėra, analizuoti, kokiais argumentais galima pagrįsti esamus skirtumus. Bet kuriuo atveju, tenka atsakyti į du klausimus: “Ką verslas gali daryti?” ir “Ką verslas gali padaryti?” (23 pav.). Vizijoje turi būti tik tai, ką galima daryti ir padaryti. Be to, vizija turi būti formuluojama tais pačiais aspektais, kuriais buvo analizuojama verslo sistema, t. y. įeigos logistikos, išeigos logistikos, teisinio reguliavimo ir įvaizdžio aspektais.

Pavyzdys

Vizijos pavyzdys pateiktas 24 pav.

Aspektas	Strateginiai tikslai
Ivaizdis	<i>Sukurti ir išpopuliarinti pagrindinių paslaugų brendus</i>
Teisinis reguliavimas	<i>Teikti tik tokias paslaugas, kurių kainas reguliuoti</i>

	<i>draudžia Europos Sajungos teisė</i>
Įeigos logistika	<i>Įsigyt iki aukštos kokybės medžiagų ir pirkti jas be tarpininkų.</i>
Išeigos logistika	<i>Užtikrinti teikiamų paslaugų inovatyvumą ir aukštą kokybę.</i> <i>Integruoti visų verslo procesų valdymą.</i> <i>Pagerinti sprendimų priėmimo procesą.</i>

24 pav. Verslo vizijos pavyzdys



Kaip jau minėjome, vizijos formulavimas yra bene atsakingiausias reikalavimų formulavimo momentas. Jei bus suformuluota nereali ar neįgyvendinama vizija, jei bus pajudėta ne ta kryptimi, jokios informacinės ir programų sistemos nieko gero neduos. Tai bus tiesiog vėjui paleisti pinigai. Bet, ir turint realią verslo viziją, nėra taip paprasta ją įgyvendinti, nes tai yra verslo sistemos pertvarkymo galbūt netgi jos reinžinerijos uždavinys. Todėl turi būti numatyta gerai apgalvota verslo pertvarkymo strategija (20 pav.). Jos nereikėtų painioti nei su ką tik minėta verslo strategija, nei su 7 paveikslėlyje parodyta verslo tobulinimo strategija. Tai trys skirtinių dalykai. Priminsime, kad strategija yra suprantama, kaip veikimo būdas, skirtas tam tikram tikslui pasiekti. Taigi, mūsų atveju visos trys strategijos skirtos skirtiniams tikslams siekti. Verslo strategija nusako kaip bandoma siekti verslo sėkmės: bandant plėstis ir įsitvirtinti naujuose rinkos segmentuose, bandant apginti nuo konkurentų ir išsaugoti turimas pozicijas rinkoje ar dar kaip nors. Vizija formuluojama strateginius tikslus, o verslo tobulinimo strategija kalba apie tai, kokių būdu tą tikslų ketinama siekti. Kitaip tariant, jei vizija atsako į klausimą "Ko ir kodėl yra siekiama?", tai verslo tobulinimo strategija atsako į klausimą "Kaip tai pasiekti?". Būtent apie tai ir kalbėsime kitame skyrelyje. Pagaliau, verslo pertvarkymo strategija kalba apie tai, kaip pereiti nuo esamos būklės prie vizija aprašomos būklės, t. y. ji kalba apie tai, ar verslo sistema turi būti keičiama palaipsniui, ar tai ketinama padaryti staiga, ar kokių nors kitu būdu.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą "Kodėl?" skamba šitaip:

"Sistemų reikia kurti todėl, kad norima įgyvendinti vizija numatytas strategines nuostatas".

Sistemoms, kuriamoms norint parduoti jas rinkoje. Kaip jau minėjome, sistemoms, kuriamoms norint parduoti jas rinkoje, atsakymas į klausimą "Kodėl?" gerokai skiriasi, nes čia yra vadovaujamasi ne užsakovo, bet vykdytojo, t. y. programinę įrangą kuriančios organizacijos, verslo tikslais. Be abejo, programinę įrangą kuriančioms organizacijoms taip pat reikia informacinių sistemų ir jas palaikančių programų sistemų, bet čia yra kalbama ne apie tokias sistemas. Taigi, programų sistemą kurianti organizacija šiuo atveju dažniausiai sprendžia tik vieną iš savo verslo problemų: ji nori kuo daugiau uždirbti, parduodama savo sukurtą produktą, ir galbūt sukurti prielaidas tolimesnei ekspansijai rinkoje. Tai – marketingo problema, sprendžiama tradiciniais marketingo metodais. Sprendžiant šią problemą, analizuojama rinka, randamos joje esančios dar neužpildytos nišos, aiškinamasi, kokių problemų nesprendžia arba blogai sprendžia rinkoje pardavinėjamos sistemos ir kuriama ne verslo, o planuojamo kurti produkto vizija. Ji aprašoma išvardinant ne strategines verslo nuostatas, o vadinamąsias produkto galimybes (angl. *features*).

Programų sistemos galimybe yra vadinamas logiškai tarpusavyje susijusių funkcinių reikalavimų rinkinys, įgyvendinus kurį vartotojui sudaroma galimybė siekti tam tikro jam svarbaus tikslų. Kalbant apie rinkoje parduodamas programų sistemas, šis terminas apibūdina grupę reikalavimų, turinčią potencialiems pirkėjams aiškiai suprantamą dalykinę paskirtį [117]. Pirkėjų pageidaujamą galimybių sąrašas nebūtinai sutampa su vartotojų operacinių poreikių sąrašu. Pageidaujamą galimybių pavyzdžiais yra galimybė išsaugoti interneto naršykleje peržiūrimų tinklalapių adresus, antivirusinės programos galimybė automatiškai atnaujinti virusų katalogą, tekstu redagavimo programos galimybė surasti tekste padarytas rašybos klaidas, galimybė pasinaudoti vadinamaja pagalbos sistema (angl. *help system*). Nei viena iš tų galimybių nesutampa su operacinių poreikiais. Jomis galima pasinaudoti, vykdant daugelį skirtinį užduočių. Kai kurios galimybės gali būti paklausios tik todėl, kad jos yra madingos, nors operaciniu požiūriu jokios didesnės vertės jos neturi. Neretai tokios galimybės yra išpopuliarinamos vykdant intensyvią reklaminę kampaniją. Kitą vertus operacinių vartotojų poreikiai taip pat yra traktuojami kaip galimybės, nes jie yra tenkinami įgyvendinant logiškai tarpusavyje susijusių funkcinių reikalavimų grupes. Kartais programinę įrangą kurianti organizacija gali siekti technologinių tikslų, neatnešančių tiesioginio pelno, bet padedančio įsitvirtinti rinkoje ir tokiu būdu gauti didesnius pelnus ateityje. Tipiniu tokiais tikslais kurto produkto pavyzdžiu yra korporacijos Microsoft sukurta platforma .net. Kuriant tokius produktus, taip pat yra rengiama produkto vizija, nusprenčiant, kokias galimybes tie produktai privalo turėti. Taigi, sistemoms, kuriamoms norint parduoti jas rinkoje, verslo reikalavimai yra suprantami kaip produkto vizijoje aprašytos to produkto galimybės. Vartotojo reikalavimai yra formuluojami produkto galimybių kontekste ir privalo būti suderinti su tomis galimybėmis.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kodėl?” šiuo atveju skamba šitaip:

“Sistemą verta kurti todėl, kad rinkoje nėra produktų, turinčių visas produkto vizijoje aprašytas galimybes ir vartotojams padedančių spręsti rinkos analizės metu išryškintas problemas.”

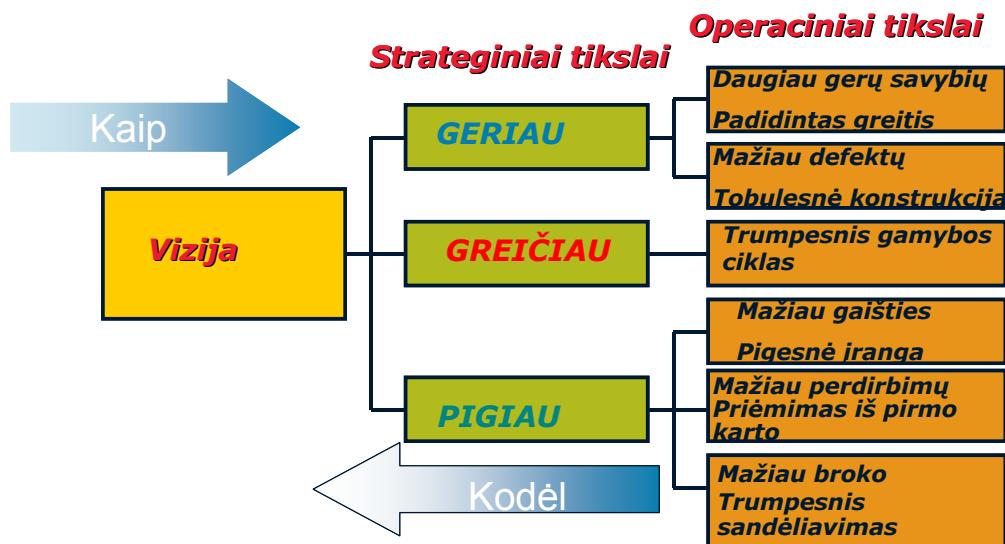
2.4.3 Kaip?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakymą į klausimą “Kodėl?” duoda verslo vizija. Ji pasako, **kodėl** reikia kurti sistemą. Dabar reikia atsakyti į klausimą “Kaip?”, t. y. į klausimą, kokias būdais ketinama įgyvendinti strategines vizijos nuostatas. Kadangi dirbama verslo reikalavimų lygmenyje, atsakymas turi būti suformuluotas verslo terminais. Kitaip tariant, vizija turi būti detalizuota ir sukoncretinta, sukonstruojant jai įgyvendinti skirtą tikslų medį (25 pav.).



25 pav. Pradinės reikalavimų formulavimo pakopos

Paprastas tikslų medžio pavyzdys parodytas 26 paveikslėlyje. Šis medis turi tris lygmenis: vizijos, strateginių tikslų ir operacinių tikslų. Kiekvienas aukštesnysis tikslų medžio lygmuo žemesniojo lygmens atžvilgiu duoda atsakymą į klausimą "Kodėl?" ir kiekvienas žemesnysis tikslų medžio lygmuo aukštesniojo lygmens atžvilgiu duoda atsakymas į klausimą "Kaip?".



26 pav. Vizijai įgyvendinti skirto tikslų medžio pavyzdys

Kaip matome, tikslų medis yra konstruojamas atliekant tikslų dekompoziciją, t. y. vadovaujamas programų sistemų inžinerijoje gerai žinomu dekompozicijos principu. Kaip ir kitais dekompozicijos atvejais, čia kyla keli esminiai klausimai:

"Kokiais kriterijais vadovautis, atliekant tikslo dekomponavimą į žemesniojo lygmens tikslus?", "Kokiais kriterijais vadovautis kokybiškai skirtiniems medžio lygmenims skirti vienas nuo kito?", "Kada baigtis tikslų dekomponavimą?"

Atsakymai į šiuos klausimus nėra paprasti. Jie keičia ir 26 paveikslėlyje parodyto tikslų medžio struktūrą. Iš tiesų medis turėtų būti sudėtingesnis. Visų pirma aukščiausiojo lygmens strateginiai tikslai jau yra suformuluoti pačioje vizijoje. Todėl, norint pradėti nuo vieno bendro tikslo, vizijos lygmenį tenka skaidyti į tris lygmenis. Aukščiausiąjį, pradinį, lygmenį vaizduoja pradine tikslų medžio viršūnė, kurią atitinkantis tikslas yra formuluojamas maždaug šitaip:

"Patobulinti esamą verslo sistemą, taip keičiant jos gebėjimus, kad būtų išspręstos misijai įgyvendinti trukdančios esminės problemas, pašalintos esminės verslo sistemos grėsmės ir duoti atsakymai prognozuojamiems artimiausios ateities verslo sistemos iššūkiams."

Antrame tikslų medžio lygmenyje turi būti keturios viršūnės, atitinkančios keturis vizijos aspektus. Jas atitinkantys tikslai yra formuluojami maždaug šitaip:

"Pašalinti vidinius verslo sistemos trūkumus, salygojančius jeigos logistikos problemas bei grėsmes ir patobulinti jeigos logistiką."

"Pašalinti vidinius verslo sistemos trūkumus, salygojančius išeigos logistikos problemas bei grėsmes ir patobulinti išeigos logistiką."

"Pašalinti vidinius verslo sistemos trūkumus, salygojančius su teisiniu reguliavimu ir verslo paslapčių bei duomenų apsauga susijusias problemas"

bei grėsmes ir pagerinti verslo sistemos atitikimą išorinio reguliavimo reikalavimams”

“Pašalinti vidinius verslo sistemos trūkumus, salygojančius verslo reputacijos ir įvaizdžio problemas bei grėsmes ir pagerinti verslo įvaizdį”

0. Patobulinti esamą verslo sistemą, taip keičiant jos gebėjimus, kad būtų išspręstos misijai įgyvendinti trukdančios esminės problemos, pašalintos esminės verslo sistemos grėsmės ir duoti atsakymai į esminius artimiausios ateities iššūkius verslui.
 1. Pašalinti vidinius verslo sistemos trūkumus, salygojančius jeigos logistikos problemas bei grėsmes ir patobulinti jeigos logistiką.
 - 1.1. Pirmas strateginis potikslis.
 - 1.1.1. Pirmojo strateginio potikslio pirmas operacinis potikslis.
 - 1.1.2. Pirmojo strateginio potikslio antras operacinis potikslis.
 - 1.2. Antras strateginis potikslis.
 - 1.2.1. Antrojo strateginio potikslio pirmas operacinis potikslis.
 - 1.2.2. Antrojo strateginio potikslio antras operacinis potikslis.
 2. Pašalinti vidinius verslo sistemos trūkumus, salygojančius išeigos logistikos problemas bei grėsmes ir patobulinti išeigos logistiką.
 - 2.1. Pirmas strateginis potikslis.
 - 2.1.1. Pirmojo strateginio potikslio pirmas operacinis potikslis.
 - 2.1.2. Pirmojo strateginio potikslio antras operacinis potikslis.
 - 2.2. Antras strateginis potikslis.
 - 2.2.1. Antrojo strateginio potikslio pirmas operacinis potikslis.
 - 2.2.2. Antrojo strateginio potikslio antras operacinis potikslis
 3. Pašalinti vidinius verslo sistemos trūkumus, salygojančius su teisiniu reguliavimu ir verslo paslapčių bei duomenų apsauga susijusias problemas bei grėsmes ir pagerinti verslo sistemos atitikimą išorinio reguliavimo reikalavimams.
 - 3.1. Pirmas strateginis potikslis.
 - 3.1.1. Pirmojo strateginio potikslio pirmas operacinis potikslis.
 - 3.1.2. Pirmojo strateginio potikslio antras operacinis potikslis.
 - 3.2. Antras strateginis potikslis.
 - 3.2.1. Antrojo strateginio potikslio pirmas operacinis potikslis.
 - 3.2.2. Antrojo strateginio potikslio antras operacinis potikslis
 4. Pašalinti vidinius verslo sistemos trūkumus, salygojančius verslo reputacijos ir įvaizdžio problemas bei grėsmes ir pagerinti verslo įvaizdį
 - 4.1. Pirmas strateginis potikslis.
 - 4.1.1. Pirmojo strateginio potikslio pirmas operacinis potikslis.
 - 4.1.2. Pirmojo strateginio potikslio antras operacinis potikslis.
 - 4.2. Antras strateginis potikslis.
 - 4.2.1. Antrojo strateginio potikslio pirmas operacinis potikslis.
 - 4.2.2. Antrojo strateginio potikslio antras operacinis potikslis

27 pav. Pati paprasčiausia tikslų medžio struktūra

Aišku, jei kuris nors aspektas nėra aktualus, atitinkama viršūnė ir tą viršūnę aprašantis tikslas yra praleidžiami. Vizijoje suformuluoti strateginiai tikslai vaizduojami tikslų medžio trečiojo lygmens viršūnėmis. Toliau šie tikslai turi būti dekomponuoti į potikslius ir dekompozicija yra tesiama tol, kol tikslai yra

dekomponuojami į žemiausiojo lygmens operacinius tikslus. Taigi, pačiu paprasčiausiu atveju, tikslų medis turi keturis lygmenis (27 pav.). Tačiau dažniausiai keturių lygmenų nepakanka, nes tiek strateginiams, tiek ir operaciniams tikslams detalizuoti prisireikia kelij lygmenų.

Tikslų dekompozicija į potikslius atliekama norint detalizuoti vizijoje suformuluotą verslo sistemos tobulinimo strategiją. Kitaip tariant, potiksliai parodo, ką reikia padaryti, norint įgyvendinti jų detalizuojamą aukštesniojo lygmens tikslą. Vieno tikslo potiksliai tarpusavyje yra sujungti logine operacija IR (t. y. konjunkcijos operacija). Tai reiškia, kad alternatyvūs tikslo įgyvendinimo būdai tikslų medyje negali būti parodyti. Kita vertus, beveik kiekvieną tikslą galima pasiekti daugeliu skirtingu būdu. Todėl, detalizuojant bet kurio lygmens tikslą, tenka rinktis vieną iš daugelio alternatyvų. Tam reikia atliki alternatyvų įgyvendinamumo analizę ir įvertinti jas pagal pasirinktus vertinimo kriterijus. Kriterijai gali būti įvairūs: kaina, ištaklių poreikis, darbų trukmė, efektyvumas ir t.t. Aišku, retai kada yra vertinama tik pagal kurį nors vieną kriterijų. Paprastai naudojama keletas kriterijų. Taigi, tikslų medžio konstravimas taip pat yra labai sudėtingas ir atsakingas uždavinys. Kadangi planuojamoji kurti programų sistema (arba galbūt netgi tokią sistemą rinkinys) privalo palaikyti numatyta tikslų įgyvendinimą, tai, pabandžius keisti tikslų medži po to, kai tokia sistema ar sistemos bus sukurtos, gali tekti išmesti visas programas ir beveik visą darbą pradėti iš pradžių.

Aptarkime kai kurias tipines tikslų medžio konstravimo klaidas.

Pirma, visų lygmenų tikslai turi būti išreiškiami verslo terminais. Tačiau, konstruodami tikslų medži, programų sistemų inžinieriai dažnai numato kompiuterizuoti kokį nors veiklos barą, sukurti kokią nors internetinę svetainę ar, tarkime, įrengti kompiuterių tinklą. Tai rimta klaida, nes išvardinti dalykai yra ne verslo tikslai, o tų tikslų įgyvendinimo priemonės. Pakeisdami tikslus priemonėmis, pažeidžiame turinių atskyrimo principą ir tikslų medyje tarpusavyje supiname skirtingus turinius (tikslus ir priemones). Taigi, medis nustoja būti *tikslų medžiu* ir praranda savo vertę.

Kita dažna klaida yra tai, kad yra supainiojami operacioniai vartotojų poreikiai ir operacioniai vartotojų tikslai. Kitaip tariant, tikslų medis neturi būti detalizuotas iki vartotojų reikalavimų lygmens. Medyje numatytiems verslo tikslams pasiekti gali prireikti tam tikros informacijos, tam tikrų skaičiavimų arba, trumpai tariant, tam tikrų informacinių, skaičiavimo bei komunikavimo paslaugų. Toms paslaugoms tekti prisireikia kompiuterių tinklų, pačių kompiuterių, periferinių įrenginių, duomenų bazii, duomenų bazii valdymo sistemų, programų paketų ir galbūt kokios nors kitos techninės bei programinės įrangos. Taigi, tuos tris dalykus – verslo tikslus, operacionius vartotojų poreikius ir poreikiams tenkinti reikalingą techninę bei programinę įrangą – reikia vienas nuo kito griežtai atskirti.

Dar viena dažnai pasitaikanti klaida yra tikslo potikslių ir to tikslo įgyvendinimo etapų supainojimas. Tikslo detalizacija negali skambeti „*Atliki analizę...*“, „*Įvertinti ...*“, „*Realizuoti...*““. Nors, įgyvendant tikslą, visa tai gali tekti atliki, tai yra jo įgyvendinimo etapai, o ne jo potiksliai.

Už kiekvieno potikslio įgyvendinimą kas nors turi būti atsakingas. Kadangi verslas vyksta vienoje ar kitoje organizacinėje struktūroje, tai už kiekvieną potikslių turi būti atsakingas koks nors tos organizacijos padalinys. Kitaip tariant, visi tikslų medžio tikslai turi būti lokalizuoti kokiame nors padalinyje arba, jei tai ne žemiausiojo lygmens tikslai, kokiame nors organizacijos valdymo lygmenyje. Tai

viena iš svarbiausių tikslų medžio įgyvendinamumo sąlygų ir, tuo pačiu, vienas iš pagrindinių tikslų dekomponavimo kriterijų. Kitas svarbus tikslų dekomponavimo kriterijus yra potikslų nepriklausomumas vienas nuo kito. Tai reiškia, kad žmonėms, užsiimantiems vieno potikslio įgyvendinimu, nereikėtų domėtis kaip yra įgyvendinami kiti tam potikslui lygiagretūs potiksliai. Bakalauro pakopos studentams skirtame programų sistemų inžinerijos kurse yra kalbama apie modulių rišlumą ir sankibą. Pagal šiuos kriterijus yra vertinama, ar teisingai buvo dekomponuota programų sistema į posistemius. Tie patys kriterijai išlieka ir čia, tik yra nagrinėjami ne modulių, o tikslų rišumas ir sankiba. Gerai sukonstruotame tikslų medyje visi tikslai turi būti vidiniai rišlūs ir minimaliai sukibę vienas su kitu. Tiesa, verslo sistemos šiuo požiūriu yra sudėtingesnės už programų sistemas ir dekomponuoti verslo tikslus į visiškai nepriklausomus potikslius pavyksta labai retai. Tačiau tokia turi būti siekiamybė ir vieno potikslio poveikis kitiems potiksliams turi būti sumažintas tiek, kiek tai yra įmanoma. Pavyzdžiui, įdiegus vieną ar kitą technologinę naujovę, tai atsilieps darbo organizavimo būdai. Tačiau, konstruodami tikslų medį, mes tuos du potikslius traktuojame kaip sąlyginai nepriklausomus.

Tikslai turi būti ne tik lokalizuojami padaliniuose, bet ir nuleidžiami į jų lygmenį. Tai reiškia, kad padaliniuose lokalizuoti potiksliai patikslina, papildo arba netgi keičia tų padalinių darbo strategijas, t. y. vadinamąsias organizacijos funkcinio lygmens strategijas.

Tikslai neturi būti per daug detalizuoti. Per didelis tikslų detalizavimo laipsnis, t. y. susmulkinimas į labai smulkius potikslius, apsunkina tikslų įgyvendinimo kontrolę, nes vadovybei tenka stebėti, kas vyksta, tarkime, padalinyje, atsakingame už dvidešimties potikslų įgyvendinimą. Taigi, visą procesą tampa sunku kontroliuoti ir valdyti.

Tikslų medžiui reikia aprašyti jo ribojimus. Svarbiausieji ribojimai yra verslo sistemos galingumas, prieinamumas ir darbo kokybė. Jei tikslų medžio potiksliai yra traktuotini kaip funkciniai verslo sistemos reikalavimai, tai galingumas, prieinamumas ir kokybė yra traktuotini kaip jos nefunkciniai reikalavimai. Vėliau iš jų yra išvedami informacinės sistemos ir ją palaikančių programų sistemų nefunkciniai reikalavimai. Tiesa, kaip kurie nefunkciniai reikalavimai, būtent apsaugos, našumo ir sistemos išdėstymo reikalavimai, yra aprašomi, atsakant klausimus “Kas?”, “Kada?” ir “Kur?”.

Trumpai aptarkime, kaip yra suprantami verslo sistemos galingumas, prieinamumas ir darbo kokybė. Verslo sistemos galingumu yra vadinamas dydis, nusakantis tos sistemos gaminamų gaminių bei jos teikiamų paslaugų apimtis. Verslo sistemos prieinamumu vadinama tikimybė, kad ta sistema bus galima pasinaudoti visuomet, kai to prireiks. Pavyzdžiui, kalbant apie banko sistemą, tai būtų tikimybė, kad klientas galės atliglioti jam reikalingą banko operaciją, tarkime, paimti pinigus iš savo sąskaitos, visuomet, kuomet jam to prireiks. Verslo sistemos darbo kokybė yra matuojama jau mūsų aptartais vadinamaisiais verslo sėkmės matais. Patobulintai verslo sistemai gali būti nustatytos kitos kritinės sėkmės matų vertės, be to, kai kurie matai gali tapti nebeaktualūs ir būti pakeisti kitaip.

Visi nefunkciniai verslo sistemos reikalavimai, įskaitant ir tuos, kurie yra aprašomi atsakant į kitus klausimus, turi būti lokalizuojami tikslų medžio potiksliuose ir nuleidžiami žemyn. Galų gale jie turi tapti funkcinio lygmens strategijas ribojančiais ribojimais. Beje, visi tie ribojimai gali būti traktuojami ir kaip atitinkamos verslo taisyklės.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kaip?” skamba šitaip:

„*Verslo vizijos strateginės nuostatos yra įgyvendinamos, įgyvendinant visus tikslų medžiu aprašytus potikslius.*“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Produkto vizijoje numatytos galimybės, panašiai kaip ir verslo vizijoje numatyti strateginiai tikslai, turi būti detalizuotos. Tai padaroma sukuriant vadinamąjį *produkto galimybų medį*, savo struktūra primenantį verslo tikslų medį. Kaip ir tikslų medis, galimybų medis turi vieną pradinę viršūnę. Ją atitinka vadinamasis *vizijos esmės aprašas* (angl. *vision statement*). Goffrey Moore [79] pasiūlė vizijos aprašo šabloną. Pritaikius jį lietuvių kalbai, šabloną galima apibrėžti šitaip (apibrėžtyje naudojama Bekaus-Nauro notacija):

<vizijos esmės aprašas> ::= **Produktas** <produkto pavadinimas>, **skirtas**
<potencialūs pirkėjai>, **kuriems reikia** <problemos įvardinimas>. **Tai**
<produkto kategorija>, **kuris (i)** <svarbiausios galimybės, dėl kurios ir
perkamas produktas aprašas>. **Skirtingai negu** <ankstesnė alternatyva,
t. y. esami produktai arba užduoties vykdymas rankiniu būdu>, **šis**
produktas <svarbiausio produkto privalumo aprašas>

Pavyzdys

Ši šabloną panaudojant parengtas vizijos esmės aprašas atrodo šitaip:

„**Produktas** „Dienos balansas“ **skirtas** dideliems prekybos centram, **kuriems reikia** vesti kasdieninę iplaukų ir išlaidų apskaitą. **Tai** su duomenų baze dirbantis programų paketas, **kuris** formuoja ataskaitas apie per darbo dieną atliktus prekių įsigijimus ir pardavimus. **Skirtingai negu** kiti rinkoje parduodami produktai, **šis produktas** duomenis apie iplaukas ima tiesiog iš kasos aparatu ir leidžia matyti einamąjį paros balansą bet kuriuo momentu, o ne tik darbo dienos pabaigoje. Be to, iplaukos yra suskirstytos pagal parduodamų prekių nomenklatūrą, kas leidžia operatyviai stebeti kaip vyksta prekyba vienomis ar kitomis prekėmis.”



Vizijos esmės aprašas yra skaidomas į vizijoje numatytas produkto galimybes, kurios gali būti suskaidytos į smulkesnes galimybes. Norint supaprastinti projekto valdymą, galimybų medis, kaip ir tikslų medis, neturi būti detalizuotas iki per daug smulkų galimybų. Paprastai yra apsiribojama 3-4 lygmenų medžiu. Jei verslo tikslai yra lokalizuojami organizacijos padaliniuose, atsakinguose už atitinkamų potikslų įgyvendinimą, tai produkto galimybės yra lokalizuojamos vykdytojų, atsakingų už galimybų realizavimą, grupėse. Galimybės, atitinkančios tarpinius medžio lygmenis, lokalizuojamos atitinkamuose projekto valdymo lygmenyse, pavedant jiems kontroliuoti tų galimybų realizavimo eigą. Kaip ir tikslų medžiu, galimybų medžiu aprašomi nefunkciniai ribojimai, nusakantys kuriamo produkto galingumą (produktyvumą), prieinamumą ir darbo kokybę. Produkto galingumas suprantamas kaip dydis, nusakantis produkto generuojamų rezultatų apimtis, prieinamumas – kaip tikimybė, kad produktu bus galima pasinaudoti visuomet, kai bus norima juo pasinaudoti, ir kokybė – kaip dydis nusakantis produkto generuojamų rezultatų tikslumą.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kaip?” šiuo atveju skamba šitaip:

„Produkto vizija yra įgyvendinama, įgyvendinant visas galimybių medžių aprašytus galimybes.“

2.4.4 Ką?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakant į klausymą “Ką?”, yra aptariami verslo objektų reikalavimai. Verslo objekto savoka apima ne tik atitinkamus artefaktus, bet ir įvykius, procesus, procedūras bei funkcijas. Kadangi verslo objektai įvardinami dalykinės srities terminais, tai visų pirma, turi būti pateiktos tų terminų apibrėžtys arba, kitaip tariant, paaiškinta, kaip verslo sistemoje yra suprantami “kliento”, “užsakymo”, “užsakymo įvykdymo”, “pirkimo”, “pardavimo” ir kitos savokos. Reikia pabrėžti, kad dažniausiai vieningos terminijos verslas neturi. Vartojami terminai ir jų apibrėžtys priklauso nuo vadinančių *požiūrių*. Tiekių, užsakovai, klientai, pardavimo tinklai, skirtingu organizacijos padalinių darbuotojai bei organizacijos vadovybė išskiria skirtingus verslo objektus ir skirtingai juos apibrėžia. David Hay vadina šiuos požiūrius *diverguojančiais* [47]. Juose skirtinti verslo objektai gali būti įvardinami tuo pačiu terminu ir, atvirkščiai, tie patys verslo objektai gali būti vadinami skirtingai. Be to, gali skirtis ir objektų konceptualizacija, t. y. skirtinguose požiūriuose su tuo pačiu objektu gali būti siejamos skirtintos tų objektų savybės. Formuluodamas verslo objektų reikalavimus, sisteminis analitikas privalo dokumentuoti visus požiūrius ir juos integruoti. Analitiko požiūrį į verslo objektus David Hay vadina *konverguojančiu* [47], nes jis išsprendžia skirtintų požiūrių prieštaravimus ir juos susieja į vieną darnią visumą. Tiesa, tai pavyksta ne visuomet. Jei požiūrių integruoti nepavyksta, skirtintus požiūrius reikia išsamiai dokumentuoti, išryškinant visus jų prieštaravimus bei skirtumus. Nors analitiko vartojami terminai paprastai yra abstraktesni negu kituose požiūriuose vartojami terminai, visi jie turi būti išverčiami į bet kurio požiūrio kalba. Kitaip tariant, formuluojant verslo objektų reikalavimus, reikia aprašyti ir visas terminų atitiktis.

Verslo objektų reikalavimai turi aprašyti ne tik pačius objektus, bet ir leistinas tų objektų būsenas bei ryšius, kuriais jie gali būti susieti. Taip pat turi būti aprašyti visos verslo taisyklės, vienaip ar kitaip ribojančios verslo objektų savybes, būsenas ar tų objektų panaudojimą.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Ką?” skamba šitaip:

„Tikslų medžių aprašyti potiksliai yra įgyvendinami operuojant nurodytais verslo objektais ir darant tą, nepažeidžiant nurodytų verslo taisyklių.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Dauguma programų sistemų apdoroja kokius nors informacinius objektus. Tie objektai gali modeliuoti kokius nors realaus pasaulio objektus ar jokių atitikmenų realiame pasaulyje neturėti ir egzistuoti tik virtualioje erdvėje. Kai kurios programų sistemos valdo realaus pasaulio objektus, tarkime raketas ar skalbimo mašinas, arba stebi jų veikimą. Bet kuriuo atveju programų sistemos dirba su tam tikrais vartotojams prasmingais objektais (arba jų informaciniais modeliais). Atsakant į klausimą “Ką?”, yra sudaromas tokį objektų sąrašas ir išrašomi reikalavimai (leistinos būsenos, kiti ribojimai), kuriuos tie objektai turi tenkinti. Savaime aišku, objektų sąrašas turi būti išvedamas iš galimybių medžio.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Ką?” šiuo atveju skamba šitaip:

„Galimių medžiu aprašytas galimių realizuojanti programinė įranga dirba su nurodytais vartotojams prasmingais objektais ir tą darydama negali taip tų objektų keisti, kad būtų pažeisti nurodyti reikalavimai.“

2.4.5 Kas?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakydami į klausimą “Kas?”, mes nustatome, kas naudosis kuriamos sistemos teikiamomis paslaugomis ir kokios yra jų pareigybų teisės bei įgaliojimai. Visa tai aprašoma verslo sistemos terminais. Bendruoju atveju, reikia nagrinėti šias paslaugų gavėjų grupes: verslo sistemos darbuotojai (pareigybės); verslo sistemos klientai; verslo partneriai, išskaitant verslo sistemą kontroliuojančias valstybines įstaigas; kompiuteriniai procesai ir įrenginiai. Įrenginiai, kaip sistemos paslaugų gavėjai, paprastai esti tik vadinamosiose techninėse informacinėse sistemoje. Kompiuteriniai procesai, arba, paprasčiau kalbant, įvairios kompiuterinės programos, šiuo metu naudojasi daugelio informacinių sistemų paslaugomis. Netgi verslo sistemos klientai ir partneriai gana dažnai naudojasi sistemos teikiamomis paslaugomis per vienus ar kitus programinius tarpininkus. Tačiau, informacinė sistema visų pirma vis tik yra skirta verslo sistemai aptarnauti ir todėl didžioji jos galimių dalis yra skirta toje sistemoje dirbančių darbuotojų poreikiams tenkinti. Kadangi informacinė sistema, o po to ir ją palaikančios programų sistemos yra kuriamos vizijai įgyvendinti, o vizija yra detalizuota tikslų medžiu, tai apatiniai tikslų medžio potikslius reikia susieti su atitinkamomis pareigybėmis, tiesiogiai atsakingomis už tų potikslių įgyvendinimą, nes informacinės sistemos ir ją palaikančių sistemų paslaugos turi būti teikiamos būtent šioms pareigybėms ir padėti joms vykdyti jų atliekamas verslo operacijas. Kita vertus, potiksliai taip pat gali būti siejami ir su klientais, verslo partneriais, kompiuteriniais procesais bei įrenginiais. Kitaip tariant, verslo tobulinimo strategija iš padalinių lygmens dabar turi būti nuleista į konkrečių pareigybų lygmenį, nustatant kokias operacijas su kokiais verslo objektais turi teisę atlikti tos pareigybės, bei susieta su klientais, verslo partneriais ir galbūt kitais paslaugų gavėjais. Apskritai, pareigybinių struktūra yra hierarchinė. Kitaip tariant, tiesioginiai vykdytojai yra kontroliuojami įvairaus lygmens vadovų ir neabejotinai tiems vadovams taip pat prireiks tam tikrų paslaugų, bent jau informacijos apie tai, kiek sėkmingai vyksta vizijos tikslų įgyvendinimas. Taigi, turi būti suformuluoti ne tik reikalavimai, nustatantys, kas tiesiogiai prisidės prie vizijoje suformuluotų tikslų įgyvendinimo, bet ir reikalavimai, nustatantys kas stebės ir kontroluos tiesioginių vykdytojų darbą įvairiuose hierarchijos lygmenyse. Apskritai, turėtų būti aprašytos ne tik hierarchinės, bet ir kitos potiksliams įgyvendinti reikalingos pareigybų sąveikos grandinės. Kitaip tariant, turi būti parodyta, kokias pareigybės tiesiogiai ar netiesiogiai palies kiekvieno potikslio įgyvendinimas.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kas?” skamba šitaip:

„Tikslų medžiu aprašytus potikslius įgyvendina nurodytos pareigybės, turinčios nurodytas teises ir nurodytus įgaliojimus. Nurodytu būdu potiksliai yra susieti su klientais bei verslo partneriais, kuriems taip pat yra suteikti nurodyti įgaliojimai ir nurodytos teisės.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Atsakant į klausimą “Kas?”, produkto galimybės yra susiejamos su vartotojų, kuriems tą galimybių gali prisireikti, klasėmis. Kitaip tariant, nustatomos kuriamo produkto potencialiu vartotojų klasės ir nusprendžiama su kokiais objektais ir ką kiekviena vartotojų klasė galės daryti. Bendruoju atveju, produkto vartotojais gali būti žmonės, kompiuteriniai procesai ir įvairūs įrenginiai.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kas?” šiuo atveju skamba šitaip:

“Galimybių medžiu aprašytomis galimybėmis galės naudotis nurodytos vartotojų klasės ir kiekviena iš jų su vartotojams prasmingais objektais galės daryti tik tai, kas jai yra leidžiama”.

2.4.6 Kur?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakant į klausimą “Kur?”, prieitame skyrelyje aprašytos pareigybės yra susiejamos su atitinkamomis darbo vietomis. Kitaip tariant yra nustatomi reikalavimai, kokiose vietovėse, pastatuose ar kambariuose bus reikalinga prieiga prie kuriamos informacinės sistemos ir ją palaikančią programą sistemą teikiamą paslaugą. Be abejo, sistemos teikiamomis paslaugomis naudosis ne tik organizacijos darbuotojai, bet ir jos verslo partneriai bei jos klientai, o galbūt ir kokie nors kompiuteriniai procesai ar kokie nors įrenginiai. Klientams aptarnauti yra reikalingi jų aptarnavimo terminalai. Todėl reikia numatyti, kur juos planuojama įrengti. Aišku, ne visuomet yra galima tiksliai pasakyti, kokiuose konkrečiuose taškuose bus įrengti terminalai, o be to tiek jų skaičius, tiek ir jų išdėstymas gali kisti. Tačiau, visuomet galima nusakyti kokias nors teritorines ribas vietovės, kur juos galima įrengti. Pavyzdžiui, kuriant banko klientų aptarnavimo sistemą, reikia nuspręsti, kur bus leidžiama įrengti banko bankomatus. Kalbant bendrosios sistemų teorijos terminais, čia yra nustatoma sistemos veikimo zona, t. y. sritis, kurioje gali būti įrengti prieigos prie sistemos mazgai.

Kadangi šiuo metu beveik visos sistemos yra išskirstytos, atsakymas į klausimą “Kur?” yra labai svarbus, nes būtent šitaip yra pradedami formuluoti bet kurios išskirstytosios sistemos reikalavimai. Tačiau, daugeliu šiuolaikinių sistemų, pavyzdžiui, elektroninės bankininkystės sistema, galima pasinaudoti praktiskai iš bet kurios pasaulio vienos. Todėl, atsakant į klausimą “Kur?” reikia, pasakyti, kokiomis tikslų medžiu nusakytomis sistemos galimybėmis bus galima naudotis intranete, kokiomis – ekstranete ir kokiomis - internete. Tiesioginė verslo sistemos veikimo zona apima tik intranetą. Formuluodami verslo tikslų susiejimo su darbo vietomis reikalavimus, mes aprašome kas daroma intranete, ekstranete ir internete, tačiau kalbame tik apie intraneto darbo vietų išdėstymą. Išimtis – organizacijos viduje įrengti klientų aptarnavimo terminalai. Nesvarbu, ar tai kompiuteris, kuriuo gali pasinaudoti klientas, bankomatas, koks nors bilietu pardavimo automatas ar tiesiog klientus aptarnaujantis padalinys. Aišku, net ir būdamas organizacijos viduje, klientas neturi teisės naudotis intraneto teikiamomis galimybėmis. Tačiau klientų aptarnavimo terminalai organizacijoje turi būti įrengti ir todėl reikia numatyti, kiek jų bus ir kaip jie bus išdėstyti. Tai darant, negalima pamiršti, kad klientais gali būti ir senyvo amžiaus bei neįgalieji asmenys ir klientų aptarnavimo terminalai turi būti įrengiami tokiemis asmenims nesunkiai pasiekiamose vietose.

Ekstraneto ir interneto vartotojai taip pat turi būti aptarnaujami. Jų darbo vietas yra jų pačių rūpestis, tačiau sistemoje turi būti numatytos jiems aptarnauti

skirtos aparatūrinės bei programinės priemonės. To reikia ir sistemos aptarnaujamiems kompiuteriniams procesams bei įrenginiams. Tokius aparatūrinius programinius kompleksus galima traktuoti kaip savo iškasas darbo vietas. *Mes jas vadinsime prieigos prie sistemas mazgais.* Savaime aišku, kad, formuluojant sistemas reikalavimus, reikia suformuluoti ir prieigos prie sistemas reikalavimus. Nors šie reikalavimai nebūtinai turi nustatyti, kur konkrečiai turi būti įrengti prieigos prie sistemas mazgai, skaitysime, kad tokie reikalavimai vis vien turi būti formuluojami atsakant į klausimą “Kur?”, nes jie tiesiogiai siejasi su informacinių sistemų išskirstymu. Formuluojant verslo reikalavimus, pakanka pasakyti tik kiek skirtingų tipų prieigos prie sistemas mazgų reikia ir su kokių tikslų medžio potikslių įgyvendinimu jie yra siejami.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kur?” skamba šitaip:

„Tikslų medžiu aprašytiems potiksliams įgyvendinti, reikalingos nurodytos darbo vietas bei klientų aptarnavimo terminalai, išdėstyti nurodytoje sistemas veikimo zonoje, ir prieigos prie sistemas mazgai, kurie, be kita ko, leidžia pasinaudoti nurodytomis sistemas galimybėmis per ekstranetą ir per internetą.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Šiuo atveju, atsakant į klausimą “Kur?”, yra nusprenčiama ar, norint pasinaudoti produkto galimybėmis, reikia turėti specialią kliento programinę įrangą, ar tą galima padaryti pasinaudojant interneto naršykle, ar paprasčiausiai pakanka produktą instaliuoti savo kompiuteryje. Galimi ir kitokie darbo vietas reikalavimai. Be to, turi būti pasakyta, ar produktas gali būti išskirstytas kompiuterių tinkle ir, jeigu taip, tai kiek darbo vietų jis gali aptarnauti.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kur?” šiuo atveju skamba šitaip:

„Galimių medžiu aprašytomis galimybėmis galima pasinaudoti tik darbo vietose, tenkinančiose nurodytus reikalavimus.“

2.4.7 Kada?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakant į klausimą “Kada?”, yra formuluojami verslo sistemas našumo reikalavimai, iš kurių po to yra išvedami informacinių sistemų ir atitinkamų programų sistemų našumo reikalavimai. Našumas gali būti suprantamas įvairiai. Dažniausiai jis traktuojamas taip, kaip tai yra įprasta bendrojoje sistemoje inžinerijoje, t. y. kaip laikas, reikalingas atitinkamems verslo operaciniams tikslams pasiekti, iškaitant laiką sugaištamą logistikos, pridėtinės vertės ir galbūt kitose grandinėse. Priminsime, kad operacinius vadiname einamuosius, trumpalaikius verslo sistemas tikslus.

Norint suformuluoti našumo reikalavimus, visų pirma reikia išsiaiškinti, kokiui mastui verslo sistema priskirtina vienai ar kitai verslo sistemos klasei. Vienose verslo sistemoje svarbiausi yra verslo įvykiai. Tokios sistemos vadinamos įvykiais valdomomis sistemomis. Kitose sistemoje įvykiai yra antriniai, o pirmaelis vaidmuo tenka verslo situacijoms. Tai situacijoms valdomos sistemos. Dar kitose – svarbiausias vaidmuo tenka nurodymams, direktyvoms ar netgi verslo sistemos viduje kylančioms iniciatyvoms. Apskritai, verslo sistemos pobūdį dažniausiai apsprendžia formalūs ir neformalūs jos įsipareigojimai (misija, sudaryti sandoriai ir kt.).

Ivykiais valdomų verslo sistemų našumas apibūdinamas reikalavimais, nustatančiais, kiek gali ilgiausiai užtrukti įvykių apdorojimas (t. y. verslo transakcijos) sistemoje. Anglų kalboje tai apibūdinama terminu *timing*. Įvykiai yra skirtomi į kalendorinius ir verslo įvykius. Kalendorinių įvykių pavyzdžiais yra finansinių metų pabaiga, ketvirčio pabaiga, teisės aktais numatyta kokio nors balanso ar ataskaitos pateikimo data, naujos darbo dienos pradžia ir pan. Verslo įvykių pavyzdžiais yra įvykiai "Gautas užsakymas", "Sandėlyje baigesi ten saugomų žaliavų atsargos", "Klientui išsiusta sąskaita faktūra už suteiktas paslaugas" ir kiti panašūs į juos įvykiai. Taigi, šiuo atveju, verslo sistemos našumo reikalavimai gali būti traktuojami kaip verslo taisyklės, nustatančios leistinas įvykių apdorojimo trukmes. Pavyzdžiui, taisyklė "Išnagrinėti kliento skundą ir raštu pranešti klientui apie nagrinėjimų rezultatus privalu ne ilgiau kaip per 3 dienas.". Reikia atkreipti dėmesį į tai, kad tokiomis taisyklėmis yra apribojamas visos verslo sistemos reakcijos į vykstančius įvykius laikas, o ne konkrečių verslo užduočių vykdymo laikai. Aišku, gali taip atsitikti, kad, pradėjus nagrinėti užduočių vykdymo laiką, paaiškės, jog kuris nors našumo reikalavimas yra neįgyvendinamas. Tada teks grįžti atgal ir ši reikalavimą peržiūrėti. Taip gali atsitikti ne tik su našumo, bet ir su kitokiais reikalavimais. Siekiant to išvengti, kiekvienam reikalavimui formulavimo lygmenyje reikia atlirkti pirminę formuluojamą reikalavimų įgyvendinamumo analizę. Išsamiau apie tai kalbėsime kituose šios knygos skyriuose.

Grįžtant prie atsakymo į klausimą "Kada?", reziumuosime, kad, formuluojant įvykiais valdomų verslo sistemų našumo reikalavimus, reikia sudaryti sistemai svarbių įvykių sąrašą ir išsiaiškinti bei dokumentuoti verslo taisykles (tai gali būti ir teisiniai aktais iš išorės primestos taisyklės), nustatančias tų įvykių apdorojimo trukmę ir tvarką.

Situacijomis valdomų verslo sistemų našumo reikalavimai yra formuluojami panašiai. Aišku, šiuo atveju reikia sudarinti ne verslui svarbių įvykiui, bet verslui svarbių situacijų sąrašą. Be to, išsiaiškinti visas verslo sistemai svarbias situacijas gali būti sunkiau, negu išsiaiškinti visus jai svarbius įvykius. Kartais situacijos esti labai sudėtingos, salygotos daugelio technologinių, socialinių, rinkos ir kitų vidinių bei verslo aplinkos veiksnių [78]. Bendruoju atveju, bet kuri situacija gali būti aprašyta atitinkamu predikatu. Situacija gali būti pageidautina arba nepageidautina. Verslo sistema siekia kuo ilgiau išlaikyti palankias situacijas ir keisti nepalankias situacijas. Tieki vienu, tiek kitu atveju situacija gali salygoti tam tikrus veiksmus. Taigi, šiuo atveju našumo reikalavimai aprašomi verslo taisyklėmis, nustatančiomis per kokį laiką reikia atlirkti verslo situacijų inicijuojamus veiksmus. Gali būti formuluojami ir sudėtingesni reikalavimai, nusakantys tų veiksmų efektyvumą, t. y. kiek jie prisideda prie palankių situacijų išsaugojimo ir kaip greitai ir kryptingai keičia nepalankias situacijas.

Nurodymais valdomų verslo sistemų, jos dar yra vadinamos komandinėmis sistemomis, našumo reikalavimai nusakomi verslo taisyklėmis, nustatančiomis per kiek laiko turi būti įvykdyti kiekvieno iš galimų tipų nurodymai. Taigi, reikia sudaryti potencialiai galimų verslo sistemai svarbių nurodymų tipų sąrašą ir suformuluoti atitinkamas verslo taisykles.

Vidinėmis iniciatyvomis valdomos verslo sistemos pavyzdžiu gali būti bendrovė, kurianti parduoti rinkoje skirtą programinę įrangą. Jos darbuotojai siūlo pradėti kurti Tačiauią ar kitokią įrangą, manydami, kad ji bus paklausiai rinkoje. Tos iniciatyvos yra analizuojamos, svarstomos ir arba aprobuojamos, arba atmetamos.

Aprobuotos iniciatyvos yra įgyvendinamos. Taigi, šiuo atveju našumą apsprendžia du veiksniai: per kiek laiko iniciatyva turi būti įvertinta ir per kiek laiko ją reikia realizuoti. Formuluoti našumo reikalavimus tokioms verslo sistemoms yra labai sudėtinga, nes iniciatyvas yra sunku tipizuoti ir dėl to beveik neįmanoma pasakyti per kiek laiko jos turi būti įgyvendinamos. Tai priklauso nuo to, kas konkrečiai yra siūloma. Nustatyti ribojimus laikui, per kurį iniciatyva turi būti įvertinta, yra žymiai paprasčiau. Pagrindinė klaida, daroma formuluojant tokius reikalavimus, yra ištaklių analizės ignoravimas. Tarkime, nusprendžiama, kad bet kuri iniciatyva turėtų būti įvertinta per dvi savaites, bet neatsižvelgiama į tai, kad iniciatyvas vertinančią ekspertų grupę sudaro penki asmenys, o iš karto gali būti pateikta 10 ar daugiau iniciatyvų. Taigi, šitaip suformuluotas reikalavimas tampa neįgyvendinamu.

Realios verslo sistemos retai kada esti aiškiai išreikšto pobūdžio. Jose vyksta įvykiai, susidaro tam tikros situacijos, keliamos iniciatyvos, joms duodami nurodymai. Tai reiškia, kad ir našumo reikalavimai yra daugialypiai, apimantys tiek įvykių, tiek situacijų, tiek iniciatyvų apdorojimą, tiek ir nurodymų vykdymą.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kada?” skamba šitaip:

„Igyvendinant tikslų medžiu aprašytus potikslius, nurodyti įvykiai, situacijos bei iniciatyvos turi būti apdorojami ir nurodytos direktyvos turi būti įvykdomos ne ilgiau kaip per nurodytą laiką.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Šiuo atveju yra aprašoma, koks maksimalus vartotojams priimtinis laikas gali būti sugaištas, norint pasinaudoti kiekviena iš galimybių medžiu numatytu produkto galimybių.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kada?” šiuo atveju skamba šitaip:

„Vartotojas turi turėti galimybę pasinaudoti kiekviena iš galimybių medžiu aprašytų galimybių, sugaišdamas ne daugiau laiko, negu yra nurodyta.“

2.4.8 Baigiamosios pastabos

Šiandien mažai kas beabejoja, kad programų sistemos reikalavimų formulavimą reikia pradėti verslo problemų bei tikslų analize. Pavyzdžiui, Karl Wiegers rašo:

„...reikalavimų grandinėje verslo reikalavimai yra aukščiausiai abstrakcijos lygmenyje: jie nusako programų sistemos viziją ir veikimo sritį. Vartotojo reikalavimai ir funkciniai programinės įrangos reikalavimai turi būti formuluojami verslo reikalavimų kontekste ir sutinkamai su tuo reikalavimų nustatytais tikslais. Programų sistemai neturi būti formuluojami jokie reikalavimai, nepadedantys siekti verslo tikslų.“ [117]

Panašiai mano ir David Hay:

„Yra labai sunku suformuluoti sistemos reikalavimus, jeigu organizacijoje nebuvvo atliktas strateginis jos veiklos planavimas arba jeigu tas planavimas buvo atliktas neteisingai. Kuriant bet kurią šiek tiek didesnę sistemą, visi su tuo susiję asmenys, išskaitant ir pačią aukščiausią organizacijos vadovybę, privalo tiksliai suvokti, kodėl šią sistemą verta kurti. Galų gale projekto sėkmė ar nesėkmė yra matuojama tuo, kiek sukurtoji sistema prisideda prie organizacijos misijos ir vizijos įgyvendinimo. Tačiau to išmatuoti negalima, jei niekas tiksliai nežino, kokios yra organizacijos misija ir vizija.“ [47]

Tiesa, dauguma autorių verslo reikalavimų formulavimo nepriskiria reikalavimų inžinerijai. Verslo reikalavimų formulavimą jie traktuoja kaip savarankišką veiklą, vadinamą strateginiu planavimu.

Vadovaudamiesi šiuo požiūriu, labiau tradiciniai reikalavimų inžinerijos vadovėliai, pavyzdžiui, [8], [70], [95], verslo reikalavimų formulavimo klausimų nenagrinėja arba juos nagrinėja labai paviršutiniškai, tačiau ir jie teigia, kad tokie reikalavimai turi būti suformuluoti. Pavyzdžiui, Dean Leffingwell ir Don Widrig rašo:

„Šiandien, išaugus programų sistemoms kurti skirtų instrumentinių sistemų produktyvumui, patenkinti realius verslo poreikius tapo lengviau negu bet kada anksčiau. Tačiau, kaip buvo parodyta, tyrimų duomenys rodo, kad teisingai suprasti ir patenkinti tuos poreikius vis dar tebéra mums iššūkis, su kuriuo mes nesugebame susidoroti. Galbūt yra paprastas šio fakto paaiškinimas, atskleidžiantis “problemos problemą”. Vykdymo kolektyvai per mažai laiko skiria tam, kad suprastų realias verslo problemas, vartotojų ir kitų suinteresuotų subjektų poreikius ir aplinkos, kurioje veiks programų sistema, pobūdį. Vietoje to mes, vykdymo kolektyvai, veržiamės pirmyn, siūlydami technologinius sprendimus, skirtus neteisingai suprastoms problemoms spręsti.“ [70]

Panašias mintis galima rasti visuose vadovėliuose.

Kai kurie žinomi reikalavimų inžinerijos specialistai, pavyzdžiui, Michael Jackson ir Pamela Zave, kritikuoją éjimą nuo tikslų prie reikalavimų. Jie rašo:

„Reikalavimų inžinerija yra apie tai, kaip pasiekti tikslus [21], tačiau tikslai patys savaime nėra tinkamas reikalavimų inžinerijos proceso pradžios taškas. Kad suprastume, kodėl taip yra, panagrinėkime projektą, skirtą sukurti kompiuterinę sistemą, valdančią barjerą, užtveriantį jėjimą į zoologijos sodą [53]. Jei inžinieriams buvo pasakyta, kad yra siekiama sutrukdyti praeiti žmonėms, nesusimokėjusiems nustatyto jėjimo mokesčio, tai toks tikslas yra aiškiai per siauras. Ar iš tiesų nėra norima padidinti zoologijos sodo pelningumą? Ar jie turėtų nagrinėti ir kitus pelno didinimo būdus, tarkime, jėjimo mokesčio sumažinimą? O ką daryti su tuo, kad dar daugiau pinigų galima gauti uždarius zoologijos sodą ir pardavus jo užimamą sklypą? Ir kodėl yra siekiama pelno? Jei, siekiant pelno, norima padaryti laimingesniu zoologijos sodo savininką, tai ar ne geriau tai daryti kitaip būdais, pavyzdžiui, įtraukiant jį į religiją arba išaiškinant jam, kad laimingiausiu jis gali būti šeimoje? Akivaizdu, kad čia kažkas ne taip. Beveik kiekvienas tikslas yra kokio nors aukštessnio tikslo potikslis. Ir inžinerija, ir religija užsiima tikslų įgyvendinimu, bet daro tai skirtingesnėmis būdais. Taigi, inžinieriui reikia pasakyti ne tik tikslą, bet ir tai, kad jo nagrinėjimų objektas ribojamas jėjimu į zoologijos sodą. Ši informacija turi nusakyti fenomenus, stebimus prie jėjimo į sodą, tarkime, lankytojus, monetas ir veiksmus, kuriuos reikia atliliki, norint patekti į sodą. Ši informacija apribuja tikslo įgyvendinimo alternatyvų rinkinį ir, kita vertus, sukuria bazę formaliam reikalavimų vaizdavimui.“ [123]

Be abejo, šios pastabos yra teisingos. Nei informacinių sistemų inžinierius, nei juolab programų sistemų inžinierius neturi išeiti iš savo kompetencijos ribų ir pradėti projektuoti svetimą verslo sistemą. Jis visuomet lieka tik ekspertas ir patarėjas, o jo ekspertizės ribas nubrėžia užsakovas. Užsakovas privalo išspręsti ir galimus tikslų konfliktus. Skirtingi verslo procesų dalyviai siekia skirtingų tikslų. Pavyzdžiui,

akcininkai nori plėtoti verslą, didinti jo produktyvumą ir gauti daugiau pinigų. Dalykinės srities specialistai nori palengvinti savo darbą, augti, vystyti savo gebėjimus. Klientams reikia platesnio teikiamų paslaugų assortimento, aukštesnės jų kokybės ir trumpesnio užsakymų vykdymo laiko. Šie tikslai gali būti bent jau iš dalies prieštaringi. Konfliktais išryškėja labai anksti, konstruojant verslo sėkmės matų sistemą. Priklausomai nuo to, kieno požiūris vyrauja, gali būti parinkti skirtingi sėkmės matai. Gali gautis ir blogiau, sėkmės matų sistema gali būti prieštaringa. Sisteminio analitiko pareiga išryškinti tokius prieštaravimus ir išaiškinti užsakovui, kieno požiūrius jie atspindi. Tačiau galų gale sprendžia tas, kas moka pinigus, t. y. užsakovas. Sisteminis analitikas gali tiktais paauskinti galimas priimto sprendimo pasekmes.

Nors verslo reikalavimams aprašyti mes skyrėme gana daug vietas. Tačiau tai nereiškia, kad pačių reikalavimų apimtis yra didelė. Paprastai jie užima vos keletą puslapių. Tačiau, tam, kad galėtų parašyti tuos kelis puslapius, analitikas privalo turėti pakankamą kompetenciją, idéti daug darbo ir sugaišti nemažai laiko. Neretai, spaudžiant užsakovui ir terminams, tam skiriama nepakankamai laiko arba ir apskritai atsisakoma verslo reikalavimų formulavimo. Apie šitokio sprendimo pasekmes mes jau kalbėjome. Belieka tiktais pridurti, kad liaudies išmintis tokias situacijas apibūdina labai taikliu posakiu "Pigią mėsą šunys éda", o buvęs Vokietijos kancleris Gerhard Schröder yra pasakęs, kad labai svarbu yra laiku spėti į traukinį, bet dar svarbiau įsesti į traukinį, važiuojantį būtent ten, kur norima nuvažiuoti. Jeigu jūs nežinote kur norite nueiti, tikslo nepasieksite, kad ir kokį kelią tam bepasirinktumete.

2.5 Vartotojo reikalavimai

2.5.1 Vartotojo reikalavimų formulavimo ypatumai

Vartotojo reikalavimai, jie dar vadinami operaciniais vartotojo poreikiais, atspindi dalykinės srities specialistų požiūrį į kuriamą informacinię sistemą ir jos programinę įrangą (t. y. į tą sistemą palaikančias programų sistemas). Dalykinės srities specialistai informacinię sistemą suvokia per jos teikiamų paslaugų prizmę. Pereinant nuo tradicinių informacinių sistemų, kuriose dominavo rankinio informacijos apdorojimo procedūros, prie šiuolaikinių kompiuterizuotų informacinių sistemų, teikiamų paslaugų nomenklatūra pasikeitė gana nežymiai, tačiau esminiai pasikeitė tų paslaugų pobūdis ir kokybė. Ir vienose ir kitose sistemose dalykinės srities specialistai gali gauti informaciją, reikalingą kokioms nors verslo operacijoms atlikti ar kokiems nors sprendimams priimti. Rengiant tokią informaciją, ir vienose ir kitose sistemose atliekamas duomenų filtravimas, agregavimas ir apibendrinimas. Tačiau, rengiant tokią informaciją rankiniu būdu, tai užtrukdavo labai ilgai, įsiveldavo daug klaidų, neretai informacijos patikimumas būdavo gana ribotas. Ribotos buvo ir galimybės pasinaudoti nutolusiais informacijos šaltiniais. Kompiuterizacija visa tai pakeitė iš esmės. Turėdami modernią informacinię sistemą, šiandien dalykinės srities specialistai tiesiogiai iš savo darbo vietų gali pareikalauti bet kurios jiems reikiamos informacijos, gauti ją per kelias minutes ir būti įsitikinę tos informacijos aktualumu, korektiškumu bei patikimumu. Be to atsirado galimybės naudotis erdvine informacija (žemėlapiai, teritorijų bei pastatų planai ir pan.), gauti informaciją apie įvairių objektų (komunikacijų, sklypų, degalinių ir pan.) geografinį išdėstymą, įvairių reiškiniių teritorinių pasiskirstymų, dirbtį su vaizdais ir garso įrašais, operatyviai rinkti analitinę informaciją interne, naudotis įvairiomis duomenų analizės ir žinių išgavimo paslaugomis, ko praktiškai nebuvo tradicinėse informaciniše sistemose. Taip pat drastiškai pakito įvairios skaiciuojamosios paslaugos, išaugo jų operatyvumas ir

patikimumas. Šiuolaikinėse informacinėse sistemose galima atlikti tokijų apimčių skaičiavimus, apie kokius tradicinėse sistemose niekas negalėjo netgi ir svajoti. Tai duoda galimybę modeliuoti priimamą sprendimą pasekmes, tiksliau prognozuoti įvairių procesų, pavyzdžiui, rinkos kainų kitimo, eiga, kurti imitacinius modelius, pavyzdžiui, apžiūrinėti statomą pastatą kompiuterio ekrane. Kompiuterių tinklai iš esmės pakeitė ir komunikavimo, koordinavimo bei grupinio darbo paslaugų pobūdį. Atsirado visiškai naujo tipo – žinių valdymo (angl. *knowledge management*) ir turinio tvarkymo (angl. *content management*) paslaugos. Tačiau daugelis dalykinės srities specialistų, o ir dalis sisteminių analitikų, vis dar neįsisavino naujų galimybių ir dėl to yra kuriamos daug prastesnės informacinės sistemos, negu jos iš tiesų galėtų būti. Visa tai reikia turėti omenyje aiškinantis vartotojų reikalavimus ir stengtis ne tik perprasti operacinius dalykinės srities specialistų poreikius, bet ir išaiškinti jems, kokiomis naujomis galimybėmis jie gali pasinaudoti.

Vartotojų reikalavimai išsiaiškinami dviem būdais. Viena vertus, vartotojo reikalavimai yra išvedami iš verslo reikalavimų, juos detalizuojant ir konkretizuojant. Kita vertus, juos aiškinamasi, apklausiant būsimus sistemos vartotojus. Užsakovo vadovybės atstovai, dalyvavę formuluojant verslo reikalavimus, atstovauti vartotojų negali, nors kartais tai ir bando daryti, nes jie vartotojų tikrujų poreikių detaliai nežino. Kuriant rinkoje parduoti skirtas programų sistemas, situacija yra dar sudėtingesnė, nes jokio konkretaus užsakovo apskritai nėra. Kartais vartotojus bando atstovauti rinkos analitikai ar marketingo specialistai, tačiau iš tiesų vartotojų tikruosius poreikius jie suvokia dar blogiau, negu verslo organizacijų vadovai. Todėl ir šiuo atveju reikia susirasti kokius nors tikrus būsimos sistemos vartotojus ir pabandyti su jais aptarti reikalavimus. Kitaip, kaip rašo Karl Wiegers

„...būkite pasirengęs skaityti žurnalų apžvalgas, aprašinėjančias jūsų produkto trūkumus, kurių būtų padėjęs išvengti tiesioginis bendravimas su vartotojais.“
[117]

Kaip jau kalbėjome, vartotojo reikalavimams skirta antroji 1 lentelės eilutė yra pirmoji tiesiogiai su reikalavimų inžinerija siejama šios lentelės eilutė, nes verslo reikalavimai su reikalavimų inžinerija siejami netiesiogiai. Tai strateginio planavimo rezultatas, naudojamas vartotojo, o vėliau ir žemesnių lygmenų reikalavimams pagrįsti. Priminsime, kad naudojant 1 lentele aprašytą Zachmano metodiką, visi reikalavimai yra išvedami iš dviejų šaltinių: reikalavimų, gautų atsakant į eilute aukščiau tame pat stulpelyje esantį klausimą, ir reikalavimų, gautų atsakant į tos pat eilutės kairiau esančio lanelio klausimą. Aišku, ši taisyklė negalioja pirmajai lentelės eilutei ir pirmajam lentelės stulpeliui. Ten aprašyti reikalavimai išvedami tik iš vieno šaltinio. Pirmąjį lentelės eilutę atitinkantys reikalavimai yra išvedami tik iš reikalavimų, gautų atsakant į tos pat eilutės kairiau esančio lanelio klausimą. Pirmajį stulpelį atitinkantys reikalavimai yra išvedami tik iš reikalavimų, gautų atsakant į eilute aukščiau tame pat stulpelyje esantį klausimą.

Vartotojo reikalavimai susieja tarpusavyje verslo ir informacinės sistemos reikalavimus, užtikrina sklandų perėjimą nuo verslo prie tą verslą palaikančių informacinių technologijų. Vartotojo reikalavimai turi būti suformuluoti, kuriant bet kurią sistemą. Formuluojant šiuos reikalavimus yra nusileidžiama iki verslo procesų ir juos sudarančių verslo užduočių lygmens ir yra nusprendžiama, kurias iš tų užduočių ir kokių mastu turi palaikyti informacinė sistema. Kadangi kokio nors informaciniu palaikymu reikia praktiskai visoms verslo užduotims, tai vartojo reikalavimai aprašo kokiems tikslams ir kokių mastu vartotojai galės pasinaudoti kuriamos IIS žemesnių lygmenų teikiamomis paslaugomis.

Kas turi formuluoti vartotojo reikalavimus, t. y. kas šiuo atveju turi veikti kaip sisteminis analitikas? Atsakymas labai paprastas. Tai turi būti būtent sisteminis analitikas², t. y. sistemiškai mąstantis asmuo, veikiantis kaip tarpininkas tarp dalykinės srities specialistų ir informacinių bei programų sistemų inžinierių. Kitaip tariant sisteminis analitikas turi gerai suprasti, kokiomis sąvokomis operuoja tiek dalykinės srities specialistai (t. y. verslo žmonės), tiek sistemas kuriantis inžinerinis personalas (t. y. informacinių sistemų inžinieriai ir programų sistemų inžinieriai) ir sugebėti tas sąvoką sistemas išreikšti vieną per kitą. Sisteminis analitikas greičiau yra pareigybė arba vaidmuo, o ne profesija. Kol kas bandymai rengti sisteminius analitikus nebuvo labai sėkmingi. Sisteminiais analitikais paprastai tampama, jie išauga tiek iš informacinių sistemų inžinierių, tiek iš programų sistemų inžinierių, tiek ir iš verslo konsultantų. Kokius reikalavimus turėtų tenkinti sisteminis analitikas ir kokius gebėjimus bei mokėjimus jis turi turėti, išsamiai buvo aptarta bakalauro pakopos studentams skirtame programų sistemų inžinerijos kurse.

Kaip ir verslo reikalavimus, vartotojo reikalavimus skaito tiek dalykinės srities specialistai, tiek ir informacinių sistemų specialistai, todėl šie reikalavimai turėtų būti parašyti tiek vieniems, tiek ir kitiems suprantama kalba. Kadangi to padaryti praktiškai yra neįmanoma, tai pirmenybė yra teikiama dalykinės srities specialistams. Pirmenybė dalykinės srities specialistams yra teikiama dėl to, kad būtent jie turi aprobuoti vartotojo reikalavimus. Taigi, reikalavimai turi būti parašyti, vengiant specialių informatikos terminų ir, jei tokią terminų išvengti nepavyksta, išsamiai ir dalykinės srities specialistams suprantamai juos paaškinant. nes reikalavimus skaitys taip pat ir inžinerinis personalas, kuriems tie terminai gali būti nežinomi.

Vartotojo reikalavimams aprašyti naudojamos įvairios metodikos. Svarbiausiomis iš jų yra užduočių <Trm81> aprašai [13], darbo scenarijų aprašai ir vadinamosios reakcijų į įvykius lentelės.

Kartais verslo konsultantų ar verslo inžinierų ir dalykinės srities specialistų (vartotojų) požiūriai gali gerokai išsisiskirti ir netgi konfliktuoti. Tuo pačiu gali kilti ir verslo bei vartotojų reikalavimų konfliktai. Dažniausiai taip yra todėl, kad verslo reikalavimai atspindi strateginius tikslus ir bendruosius viso verslo ribojimus, kurie kasdieninį operacinį darbą dirbantiems dalykinės srities specialistams gali būti mažai žinomi ir sunkiai suvokiami. Todėl, konfliktų atveju, sisteminis analitikas privalo dalykinės srities specialistams išaiškinti verslo reikalavimus. Sprendžiant konfliktus pirmenybė teikiama verslo reikalavimams, aišku, jeigu nepaaiškėja, kad tie reikalavimai yra nepakankamai pagrįsti. Paaiškėjus verslo reikalavimų nepagrįstumui, tuos reikalavimus reikia peržiūrėti.

2.5.2 Kodėl?

Sistemoms, kuriamoms konkrečiam užsakovui. Vizija ir jos tikslų medis nubréžia ilgalaikę verslo sistemos tobulinimo perspektyvą. Vizijai įgyvendinti gali prireikti ne vienerių metų ir ne vieno projekto. Todėl tikslų medyje reikia numatyti potikslių prioritetus ir tų prioritetų pagrindu suskaidyti tikslų medžį į konkrečių projektų tikslus aprašančius pomedžius. Taigi, atsakant į klausimą „Kodėl?“, yra išskiriamas einamojo projekto tikslus aprašantis tikslų medžio pomedis ir vartotojo reikalavimai, išskaitant visų žemesnių lygmenų reikalavimus, yra formulojami

² Nereikėtų painioti terminų *sistemų analitikas* ir *sisteminis analitikas*. Pirmasis analizuoją kokias nors sistemas, antrasis analizuoją ką nors, nebūtinai sistemas, tarkime, kokią nors problemą, panaudodamas vadinamojo sisteminio mąstymo principus.

remiantis būtent tuo pomedžiu. Pasirinktasis tikslų medžio pomedis tiksliai nubrėžia einamojo projekto ribas arba, kitaip tariant, nusako jo apimtį (angl. *project scope*).

Formuluojant verslo reikalavimus, tikslų medžio potiksliai susiejami su padaliniais, atsakingais už tą potikslų įgyvendinimą, ir netgi su tuose padaliniuose esančiomis darbo vietomis. Po to yra formuluojami reikalavimai, kaip reikia pakeisti vadinamąsias funkcinio lygmens strategijas, arba, kitaip tariant, kaip reikia keisti padalinių siekius. Tačiau vien šito nepakanka. Šiuolaikinėse verslo sistemos pagrindinis dėmesys yra skiriamas ne tiek funkcionams padaliniams, kiek vadinamiesiems *verslo procesams* ir *verslo transakcijoms*. Dažniausiai verslo procesas yra suprantamas [42] kaip iš dalies sutvarkytas *verslo užduočių* rinkinys, naudojantis vieno ar kelių skirtingų tipų įeigą ir išeigoje kuriantis ką nors, kas yra vertinga klientui ar užsakovui. Verslo procesai yra vykdomi siekiant tam tikrų verslo tikslų, tai yra jie realizuoja tam tikras verslo transakcijas. Kai kurioms verslo transakcijoms realizuoti gali prireikti ir kelių verslo procesų. Transakcijas inicijuoja atitinkami verslo ar kalendoriniai įvykiai. Procesai yra veikiami verslo sistemas išorėje ar kituose jos procesuose vykstančių įvykių. Taigi, atsakant į klausimą „Kodėl?“, reikia vykdomo projekto tikslų medžio potikslius susieti su konkrečiomis verslo transakcijomis bei konkrečiais verslo procesais ir suformuluoti patobulintų verslo procesų reikalavimus. Kitaip tariant, turi būti aprašyta, kaip ir kokius verslo procesus norima patobulinti vykdant einamąjį projektą ir kokios verslo transakcijos gali būti realizuotos panaudojant tuos procesus. Reikia aprašyti ir iš kokių verslo užduočių tie procesai turi būti sudaryti. Verslo užduotimis <Trm81> vadinamos verslo sistemos vykdomos užduotys. Jos yra aprašomas verslo proceso terminais. Verslo transakcijos taip pat gali būti traktuojamos kaip užduotys. Beje, nereikia painioti sistemos vykdomų transakcijų ir sistemos funkcijų. Tiktai labai retais atvejais transakciją galima realizuoti pasinaudojant kokia nors viena sistemos funkcija. Dažniausiai tam prisireikia gana sudėtingos kelių funkcijų kompozicijos.

Tariant Karl Wiegers žodžiais [117], patobulintų verslo procesų reikalavimai nusako *projekto apimtį*, nes nusako, kokį verslo transakcijų vykdymą turi palaikyti tame projekte kuriamo sistema. Reikalavimų inžinerijoje panašaus pobūdžio reikalavimų sudarymui apibūdinti yra vartojamas bendresnis terminas *bazinio reikalavimų komplekto* (angl. *requirements baseline*) *sudarymas*. Kas konkrečiai yra vadinama baziniais reikalavimais, priklauso nuo konkretaus projekto.

Vien tik sudaryti palaikomų verslo transakcijų sąrašą nepakanka. Transakcijoms dar reikia nustatyti jų prioritetus, t. y. reikia pasakyti, kokia eilės tvarka reikia kurti jas palaikančias priemones. Šito prireikia tuomet, kuomet projekto tikslų medžio pomedžiu numatytos galimybės realizuojamos ne iš karto ir projektas yra skaidomas į kelias viena po kitos vykdomas dalis (subprojektus), kiekviena iš kurių sukuria atitinkamą kuriamos sistemos prieaugi³.

Patobulintus verslo procesus, tiksliau, perėjimą nuo esamų prie patobulintų verslo procesų, galima aprašyti vadinamosiomis *verslo procesų diagramomis* [42, 115]. Tam taip pat gerai tinkta ir UML *veiklos diagramos* [113].

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kodėl?“ skamba šitaip:

³ Tai, kas yra sukuriama vykdant patį pirmąjį subprojektą, taip pat yra vadinama prieaugiu (angl. *increment*).

„Sistemą reikia kurti todėl, kad norima patobulinti nurodytus verslo procesus ir šitaip igyvendinti tikslų medžiu išskleistą verslo viziją.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Panašiai kaip verslo vizija ir jos tikslų medis, produkto vizija ir jos galimybių medis irgi nubrėžia ilgalaike produkto raidos perspektivą. Jie aprašo produkto paskirtį ir tai, kaip turi atrodyti galutinė produkto versija. Todėl projekto galimybių tikslų medžių taip pat reikia suskaidyti į pomedžius ir, pasirenkant atitinkamą pomedžių, nubrėžti griežtas projekto ribas arba, kitaip tariant, nustatyti *projekto apimtį*. Projekto apimties aprašas yra detalizuojamas aprašant tipiniam produkto vartotojui naudingą funkcionalumą. Kitaip tariant, detalizuojant projekto apimtį, yra sudaromas vartotojui prasmingų užduočių sąrašas. Prasmingomis vadinamos užduotys, padedančiomis vartotojui siekti jam prasmingų dalykinio pobūdžio tikslų. Tai verslo užduočių analogas. Jų aprašai traktuotini kaip galimų produkto panaudojimo būdų <Trm81> aprašai. Pagalbinės, techninio pobūdžio užduotys kol kas dar nėra nagrinėjamos. Taigi, sudarant vartotojui prasmingų užduočių sąrašą, yra sukuriamas *bazinis produkto reikalavimų komplektas*. Sąrašas sudaromas detalizuojant ir sukonkretinant produkto galimybių aprašus. Taigi, į jį turi būti įtrauktos ne tik vartotojui prasmingos užduotys, bet ir kitos galimybių medžio pomedžiu numatyta galimybių gimdomos užduotys (t. y. tokios, kuriomis naudojamas kaip pagalbinėmis, vykdant daugelį vartotojui prasmingų užduočių). Baziniai reikalavimai turėtų būti prioretizuoti, nes projekto galimybių pomedžiu numatytos galimybės gali būti realizuojamos ne visos iš karto ir produktas gali būti kuriamas kuriant jo priaugius.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kodėl?“ šiuo atveju skamba šitaip:

„Produktą reikia kurti todėl, kad būtų galima vykdyti vartotojui prasmingas dalykines užduotis, leidžiančias pasinaudoti produkto vizijoje numatytomis galimybėmis.“

2.5.3 Kaip?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakant į klausimą „Kaip?“, yra aprašomas vadinamasis *projekto gylis*. Kitaip tariant, nustatoma kokių informacinių, skaičiavimo, komunikavimo ar kokių nors kitokiu informacinės sistemos paslaugų prireikia vykdant aptariamąsias verslo transakcijas ir kokiui mastui tai turi būti kompiuterizuota. Tai vadinama vartotojo operacinių poreikių formulavimu (28 pav.). Verslo procesus sudarančios užduotys gali būti kompiuterizuojamos įvairiai. Vienos gali būti paliktos vykdyti rankiniu būdu, vykdant kitas, gali būti numatyta vienaip ar kitaip pasinaudoti kompiuteriu ar kompiuterių tinklu, dar kitos gali būti visiškai kompiuterizuotos (t. y. vykdomos automatiškai). Kokio gylio kompiuterizaciją pasirinkti, priklauso tiek nuo užduočių pobūdžio, tiek nuo užsakovo pageidavimų, tiek ir nuo konkrečių projekto ribojimų (per kiek laiko reikia padaryti, kiek tam yra skirta pinigų ir pan.). Be to, skirtiniems sistemoms priaugiamas tų pačių verslo užduočių kompiuterizavimo gylis gali būti skirtinas. Pavyzdžiui, gali būti nuspręsta, kad pradiniame priaugyje visos užduotys yra kompiuterizuojamos minimaliai, o paskutiniame priaugyje turi būti pasiektas maksimalus kompiuterizacijos gylis. Gali būti nuspręsta ir kitaip, pavyzdžiui, kai kurias užduotis nusprenčiama apskritai nekompiuterizuoti, nors tai ir būtų galima padaryti. Be kita ko reikia žiūrėti, kiek konkrečios užduoties kompiuterizavimas arba nekompiuterizavimas yra esminis verslo vizijos požiūriu ir kokiui mastui atsiperka tam tikslui skirtos finansinės investicijos.



28 pav. Vartotojų operaciniai poreikiai

Reikia pastebėti, kad verslo transakcijoms vykdyti reikia labai daug pačių įvairiausią paslaugų: buhalterinių, sąskaitybos, duomenų analizės ir t.t. Be jų verslo transakcijų vykdyti negalima, nes tai draudžia kokie nors teisiniai aktai arba tokia yra organizacijos politika. Kitaip tariant, atliekant bet kurią verslo transakciją prisireikia tam tikros informacijos, greičiausiai reikia kažką paskaičiuoti, galbūt reikia kam nors ką nors pranešti ir, be abejo, reikia kaupti informaciją apie transakcijos vykdymo rezultatus ir galbūt netgi apie jos vykdymo eigą. Kadangi sprendimų priemimą, planų sudarymą ir kitus panašaus pobūdžio darbus mes taip pat traktuojame kaip verslo sistemai teikiamas paslaugas, tai neabejotinai prieikia atliliki duomenų analizę, agreguoti duomenis, juos apibendrinti ir kt. Taigi, verslo sistemai teikiamų paslaugų nomenklatūra yra labai plati. Tipiniai paslaugų pavyzdžiai yra:

„Paskaičiuoti darbuotojo mėnesinį atlyginimą“, „Kaupti duomenis apie kliento sąskaitoje esančios sumos pokyčius“, „Pateikti analitinę ataskaitą su atitinkamomis išvadomis apie praėitame ketvirtuje atliktus pirkimus“ ir kt.

Visa tai yra paslaugos, kurias organizacijos informacinė sistema teikia jos verslo sistemai, tiksliau, verslą vykdantiems bei administruojantiems padaliniams. Darbuotojas, atlyginimas, klientas, kliento sąskaita, pirkimas – visa tai yra verslo objektai, kuriais operuoja verslo sistema. Informacinė sistema operuoja informaciniais tų objekto modeliais. Paslaugos gali būti teikiamos rankiniu būdu arba vienu ar kitu mastu panaudojant kompiuterius bei jų tinklus. Kitaip tariant, tai priklauso nuo informacinės sistemos teikiamų paslaugų kompiuterizavimo gylio.

Dažnai informacinės sistemos teikiamos paslaugos yra priskiriamos verslo sistemai, nes be jų neįmanoma verslo vykdyti. Tada skirtumo tarp verslo sistemos ir informacinės sistemos nėra daroma ir viskas vadinama verslo sistema, terminą informacinė sistema rezervuojant informacinės sistemos programinei įrangai pavadinči. Mūsų nuomone tokia praktika yra ydinga keliais požiūriais. Pirma, nėra daroma skirtumų tarp verslo objektų ir jų informacių modelių, dėl ko nebegalvojama, kaip geriau tuos objektus modeliuoti. Antra, pamirštama, kad informacinė sistema taip pat turi būti projektuojama ir ji arba paliekama „biurokratams“ arba stichiškai klostosi savaimė. To pasėkoje programinė įranga atitrūksta nuo realių verslo poreikių ir pradedą palaikyti galbūt visiškai netikusią informacinię sistemą. Šitaip galima palengvinti „biurokratų“ (sekretorių, vadybininkų, buhalterių ir kt.) darbą, bet negalima išspręsti jokių realių verslo problemų.

Reikia aprašyti ne tik kokių informacinės sistemos paslaugų reikia verslo transakcijoms bei užduotims vykdyti (funkciniai reikalavimai), bet ir tų paslaugų ribojimus, t. y. pateikiamų rezultatų tikslumo, rezultatų ir pačių paslaugų patikimumo

ir jų saugos reikalavimus (nefunkciniai reikalavimai). Beje, saugos reikalavimai dažniausiai esti svarbūs tik techninėms bei technologinėms informacinėms sistemoms, pavyzdžiui, technologinių procesų valdymo sistemoms.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kaip?“ skamba šitaip:

„Patobulinti verslo procesai realizuojami panaudojant nurodytas informacines, skaičiavimo, komunikavimo ir galbūt kokias nors kitokias paslaugas.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Šiuo atveju aprašoma, kokias informacines, skaičiavimo, komunikavimo ir galbūt kitokias paslaugas turi teikti produktas, kad būtų galima įgyvendinti bazinius produkto reikalavimus (t. y. realizuoti vartotojui prasmingas užduotis). Kitaip tariant, analizuojamos vartotojui prasmingos užduotys ir sprendžiama, kokios informacijos, kokių skaičiavimų, duomenų perdavimo, komunikavimo ar kitokių paslaugų prieiks vykdant tas užduotis. Turi būti aprašytos ir paslaugos, reikalingos kitoms produkto galimybėms įgyvendinti (kartais jos sutampa su pačiomis galimybėmis). Šitaip suformuluojami aukščiausiojo abstrakcijos lygmens produkto funkciniai reikalavimai. Juos reikia papildyti nefunkciniais reikalavimais (tikslumas, patikimumas, sauga), išvedamais iš produkto galimybų reikalavimų.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kaip?“ šiuo atveju skamba šitaip:

„Vartotojui prasmingos užduotys realizuojamos panaudojant nurodytas informacines, skaičiavimo, komunikavimo ir galbūt kokias nors kitokias paslaugas, kurias privalo teikiti kuriamas produktas.“

2.5.4 Ką?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakant į klausimą „Ką?“, yra formuluojami koncepciniai verslo objektus modeliuojančių informacinių objekto reikalavimai. Kaip jau minėjome, visi verslo objektai yra modeliuojami atitinkamais informaciniais objektais. Dažniausiai jie aprašomi tam tikru rodiklių rinkiniu. Tokie aprašai yra vadinti *žrašais* apie atitinkamą verslo objektą. Tačiau gali būti ir kitaip. Nuotrauka, piešinys, žemėlapis, grafikas, natos taip pat yra informaciniai atitinkamų verslo objektų modeliai. Pavyzdžiui, paveikslų galerijoje greta pačių paveikslų (verslo objektai) esti katalogai su tų paveikslų nuotraukomis (informaciniai verslo objektų modeliai).

Kiekvienam įrašu modeliuojamam verslo objektui, o daugumoje organizacijų tokį objektą esti daugiausia, reikia aprašyti kokiais rodikliais tas objeketas yra aprašomas, kokius ribojimus tie rodikliai turi tenkinti ir kokiais ryšiais objekto aprašas yra siejamas su kitų objektų aprašais. Tai padaryti nėra paprasta, nes verslo objektai gali būti naudojami skirtingose verslo transakcijose ir vaidinti jose skirtinguose vaidmenis. Tarkime, organizacijos padalinys gali būti projekto vykdytojo, kokias nors paslaugas perkančio pirkėjo ir daugelyje kitų vaidmenų. Kiekvienam vaidmeniui aprašyti yra reikalingi specialūs, jie yra vadinti *roliniai*, rodikliai. Todėl, atsakant į klausimą „Ką?“, reikia verslo objekto *bazinius rodiklius* atskirti nuo *rolinių rodiklių* ir *bazinius rodiklius* bei kiekvieno vaidmens *rolinius rodiklius* aprašyti atskirai. Baziniai vadinami rodikliai, kurių prieikia aprašinėjant bet kurį objekto vaidmenį ir bet kurią jo būseną. Pavyzdžiui, tokiais yra identifikaciniai objekto duomenys, aprašantys jo tapastį. Jeigu sistema nėra kuriamą kokiai nors naujai steigiamai

organizacijai, tai verslo sistema ir kokia nors ją palaikanti informacinė sistema, galbūt ir labai prasta, jau funkcionuoja ir yra tiktais tobulinamas. Todėl informaciniai verslo objektų modeliai (t. y. įrašų struktūra) jau yra žinomi ir sisteminiam analitikui reikia tik išsiaiškinti, kokie jie yra. Kitas dalykas, ar visi ten numatyti rodikliai iš tiesų yra reikalingi ir ar kokių nors rodiklių tuose įrašuose netrūksta. Neretai organizacijoje viena ar kita informacija yra kaupama ir netgi apdorojama tik „dėl visa ko“, „todėl, kad to reikalauja kokie nors vidiniai dokumentai“ arba „todėl, kad visi šitaip daro“, nors ta informacija nėra reikalinga jokiam verslo procesui palaikyti. Todėl, atsakant į klausimą „Ką?“, reikia atliskti išsamią organizacijoje naudojamų informacinių objektų analizę. Tai yra vadinama *duomenų analize*. Ši analizė sutelkta į du klausimus: su kokiais duomenimis yra dirbama šiuo metu naudojamoje informacinėje sistemoje ir kokių duomenų prieiks naujoje patobulintoje informacinėje sistemoje. Kiekvienai verslo transakcijai reikia aprašyti, kokių duomenų ją vykdant prisireikia ir kurius duomenis ji pati sukuria. Taip pat reikia nurodyti ir tuos tarpinius duomenis, kuriuos privalu kaupti. Be abejo, reikia suformuluoti ir visus ribojimus, kuriuos turi tenkini transakcijos duomenys.

Svarbi verslo objektų kategorija yra kalendoriniai ir verslo įvykiai. Visi verslo įvykiai yra registruojami arba, kitaip tariant, apie kiekvieną verslo įvykį yra daromas įrašas vadinamojoje *organizacijos atmintyje* (angl. *company memory*). Organizacijos atmintį sudaro visas organizacijos duomenų saugyklos, įskaitant kompiuterizuotas duomenų bazes, įvairias registravimo knygas bei visus kitus dokumentus. Įrašai apie kai kuriuos svarbius kalendorinius įvykius, pavyzdžiui, apie finansinių metų pabaigą, taip pat yra rašomi į organizacijos atmintį. Taigi, įrašas apie įvykį – tai duomenys, aprašantys ką nors svarbaus, kas įvyko versle ir apie ką organizacijai reikia sužinoti ir įsiminti savo atmintyje. Irašas gali būti kuriamas tiek organizacijos viduje, tiek ir jos išorėje. Atsakant į klausimą „Ką?“, reikia nuspręsti, kokius duomenis apie įvykį reikia fiksuoti, kad informacinė sistema tą įvykį galėtų apdoroti reikalaujamu būdu. Taip pat reikia nustatyti, kas ir kur tą įrašą sukurs ir kaip įrašas pateks į organizacijos atmintį. Kadangi verslo transakcijas inicijuoja įvykiai, tai įvykiai ir transakcijos yra glaudžiai susiję. Todėl jie turi būti analizuojami kartu.

Organizacijos atmintyje saugomiems vaizdams, nuotraukoms, grafikams, žemėlapiams, garso įrašams ir kitai *daugialypei (multimedia) informacijai* turi būti formuluojami panašūs reikalavimai bei ribojimai, kaip ir verslo objektus modeliuojantiems įrašams. Visų pirma turi būti pasakyta, kur ir kaip tokie informaciniai objektai yra kuriami ir kaip jie patenka į organizacijos atmintį. Gali būti, ir gana dažnai iš tiesų taip esti, kad organizacijos atmintyje iki šiol tokių informacinių objektų apskritai nebuvo. Pavyzdžiui, tobulinant paveikslų galerijos verslo sistemą, vizija gali numatyti sukurti galimybę peržiūrinėti galerijoje pardavinėjamus paveikslus interne. Nors iki tol paveikslų nuotraukos ir būdavo spausdinamos kataloguose, nieko daugiau su jomis nebūdavo daroma ir jokie specialūs reikalavimai nebuvo suformuluoti. O gali būti ir taip, kad galerija katalogų nespausdino ir paveikslų nuotraukos apskritai nebūdavo daromos. Taigi, atsakant į klausimą „Ką?“, reikia sudaryti išsamų netradicinių informacinių objektų sąrašą ir suformuluoti jų kokybės reikalavimus. Kadangi jau yra žinoma, kokios užduotys ir kokių mastu turi būti kompiuterizuotos, tai reikia nurodyti su kuriais objektais bus manipuliuojama kompiuteriniu būdu ir, formuluojant kokybės reikalavimus, atsižvelgti į tai, ar informacinis objektas bus tiktais rodomas ekrane, ar ir spausdinamas, ir į tai, ką su juo dar numatoma daryti.

Dar viena nejprasta informacinių objektų kategorija yra vadinamieji *turiniai* (angl. *content*). Turiniai yra tvarkomi kompiuteriniu būdu. Vartotojo požiūriu jie primena skelbimų lentas, enciklopedijas ar žynus, kuriuose galima rasti pačią įvairiausią nuolat atnaujinamą informaciją. Taigi, vartotojams yra svarbūs du dalykai:

- kokie turiniai bus prieinami informacinėje sistemoje;
- kaip tie turiniai bus organizuoti arba, kaip įprasta sakyti, kokia bus kiekvieno turinio informacinė architektūra (angl. *information architecture*).

Atsakant į klausimą “Ką?”, reikia atsakyti į abu tuos klausimus.

Dalykinės srities specialistų požiūriu, visiems informaciniams objektams labai svarbūs yra *tikslumo reikalavimai*. Kitaip tariant, reikia pasakyti, kiek tikslus turi būti informacinis verslo objekto modelis, kad būtų galima atskirti vienas nuo kito du skirtingus verslo objektus modeliuojančius informacinius objektus, tarkime, įrašą apie Lietuvos Respublikoje registruotą bendrovę „Venta“ ir įrašą apie Latvijos Respublikoje registruotą bendrovę „Venta“. Informacinių objektų yra skiriami vienas nuo kito pagal atitinkamų objektų *tapastis* (identitetus) modeliuojančius identifikacinius duomenis. Todėl, šiuo požiūriu, verslo objektų informacinių modeliavimo tikslumas yra nusakomas suformuluojant jų identifikavimo reikalavimus arba, kitaip tariant, nurodant, kokie identifikacinių duomenys apie objektą turi būti pateikti jį registruojant. Be abejo, vien to nepakanka. Reikia aprašyti ir tai, kaip tiksliai turi būti modeliuojamos objekto savybės. Pavyzdžiui, reikia žinoti, ar asmens gimimo datai aprašyti pakanka nurodyti tiktais gimimo metus, ar reikia nurodyti ir mėnesį bei dieną, ar galbūt ir gimimo valandą. Dažnai verslo objektų tapastys yra modeliuojamos specialiai tam tikslui sukurtais kodais, vadinamaisiais *identifikatoriais*. Pavyzdžiui, asmens tapastis yra modeliuojama vadinamuoju asmens kodu. Labai svarbu nekurti kokių nors savų, organizacijos masto kodavimo sistemų, o pasidomėti, ar nėra tam tikslui jau sukurtų ir kokiai nors standartais įteisintų sistemų. Sukūrus savą identifikavimo sistemą, anksčiau ar veliau teks susidurti su informacijos mainų su kitomis organizacijomis problema ir, jei ta sistema netenkins galiojančių formalinių ar faktinių standartų, gali tekti ijdėti daug darbo ir išleisti didžiules pinigų sumas pertvarkant visą organizacijos atmintyje saugomą informaciją.

Dar viena svarbi reikalavimų grupė yra *duomenų darnos reikalavimai*. Anglų kalboje vartojamas terminas *data integrity*, o lietuviškai tai dar yra vadinama *duomenų vientisumo reikalavimais*. Šie reikalavimai nusako įrašo rodiklių tarpusavio sąryšius arba, kitaip tariant, kokius ribojimus turi tenkinti įrašas, kad jis būtų korekтиškas. Tarkime, paprastas tokio reikalavimo pavyzdys yra ribojimas, kad darbuotojas negali būti jaunesnis kaip aštuoniolikos metų. Darnos reikalavimai gali riboti ne tik vieną informacinių objektą, bet ir tokiai objektų grupes. Jie gali riboti ir skirtingo tipo informacinių objektų rodiklių priklausomybes. Duomenų darnos reikalavimai yra tiesiogiai išvedami iš atitinkamų verslo taisyklių.

Dar viena svarbi reikalavimų grupė yra *informacinių objektų apsaugos reikalavimai*. Formuluojant šiuos reikalavimus, visus informacinius objektus reikia suskirstyti į grupes pagal reikalaujamą tų objektų apsaugos laipsnį ir nurodyti, nuo kokių grėsmių kiekviena informacinių objektų grupė turi būti saugoma.

Pagaliau, dalykinės srities specialistams labai svarbūs yra ne tik patys informacinių objektai, bet ir jų pateikties būdas arba, kitaip tariant, dokumentai, kuriuose yra pateikiami duomenys. Dokumentų struktūra yra aprašoma *pateikties reikalavimais*. Tai reiškia, kad turi būti pateikti visų verslo transakcijose naudojamų

dokumentų pavyzdžiai ir aprašyti visi tų dokumentų laukai. Priminsime, kad yra kalbama ne apie visus organizacijoje naudojamus dokumentus, o tik apie tuos, kurie siejami su duotojo projekto tikslu medžio pomedžiu. Atsakant į klausimą „Ką?“, reikia išsamiai išanalizuoti visus senojoje verslo sistemoje naudotus dokumentus jų tikslingo, prasmingo, išsamumo ir patogumo dalykinės srities specialistams požiūriais ir suformuluoti reikalavimus, kaip tuos dokumentus reikia keisti. Kadangi jau yra nustatyta, kokios verslo užduotys ir kokiui mastui turi būti kompiuterizuotos, tai yra aprašomas ir ekrainės duomenų įvesties ir rezultatų pateikties formos bei kiti kompiuteriniu būdu formuojami dokumentai. Formuluojant tokius reikalavimus, būtina atsižvelgti į atitinkamais teisiniais nustatytus reikalavimus.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Ką?“ skamba šitaip:

„Verslo transakcijoms ir verslo užduotims vykdyti reikia apdoroti nurodytus informacinius objektus. Vykdant transakcijas bei užduotis, negalima pažeisti ribojimų, kuriuos tie objektai turi tenkinti.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Produkto vartotojas mato tik dviejų tipų duomenis: tuos, kurių reikia jo užduotims vykdyti, ir tuos, kuriuos jis pats kuria, tas užduotis vykdydamas. Taigi, atsakant į klausimą „Ką?“, šiuo atveju reikia išanalizuoti, kokių duomenų reikia vartotojo užduotims vykdyti ir kokie duomenys yra tomis užduotimis kuriami bei suformuluoti ribojimus, kuriuos tie duomenys turi tenkinti. Ribojimai yra aprašomi duomenų tikslumo, darnos ir pateikties reikalavimais. Be to, visus duomenis reikia sugrupuoti į grupes pagal reikalaujamą jų apsaugos laipsnį ir kiekvienai grupei nurodyti nuo kokių grėsmių ji turi būti saugoma.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Ką?“ šiuo atveju skamba šitaip:

„Vartotojo prasmingoms užduotims vykdyti yra reikalingi nurodyti duomenys. Vykdydamas šias užduotis, vartotojas kuria kitus nurodytus duomenis. Visi nurodyti duomenys privalo tenkinti suformuluotus reikalavimus.“

2.5.5 Kas?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakant į klausimą „Kas?“ yra detalizuojami, konkretizuojami ir tikslinami vykdytojų reikalavimai, suformuluoti, formuluojant verslo reikalavimus. Galutinis tikslas yra suformuluoti programų sistemas vartotojo interfeisų⁴, panaudojamumo ir apsaugos reikalavimus. Nors šis tikslas turi būti realizuotas tik perėjus prie ketvirtos 1 lentelės eilutės, jo siekti

⁴ Kai kurie lietuviškos terminijos kūrėjai siūlo angliską terminą *interface* versti kaip *sąsaja*. Tam yra pritarusi ir Lietuviai kalbos komisija, tačiau toks pritarimas reiškia tik tiek, kad terminas *sąsaja* nepriestarauja lietuvių kalbos terminų darybos taisyklėms. Tačiau, terminas *sąsaja* turi porą rimtų trūkumų, dėl kurių, mano nuomone, jis yra neteiktinas. Pirma, jis neatspindi tikrosios termino esmės. Interfeisas reiškia dviejų vienetų sąveikos būdą, o ne juos siejančią *sąsaja*. Antra, žodžio *sąsaja* vartojimas termino prasme, prieistarauja terminologijos mokslo nuostatomis, sutinkamai su kuriomis žodžiai, plačiai vartojami kasdieninėje kalboje, yra netinkami būti terminais. Pažeidus šias nuostatas, gauname įvairias nepageidautinas pasekmes. Viena iš jų yra tekstai, kuriuose žodis yra vartojamas abiem prasmėm, pavyzdžiu, „vartotojo sąsajos ir duomenų bazės sąsajos sąsajos“ (paimta iš vieno bakalauro baigiamojo darbo), kita – vertimo į kitas kalbas klaidos, padaromos dėl to, kad vertėjui tą patį žodį skirtingose vietose reikia versti skirtingai (kaip kasdieninės kalbos žodį ir kaip terminą). Atlirkas eksperimentas parodė, kad net keturi skirtinti vertėjai, versdami informatikos tekstus į anglų kalbą, terminą *sąsaja* išvertė kaip *association*.

pradedama jau dabar, formulujant dalykinės srities specialistų požiūrį aprašančius reikalavimus. Visų pirma reikia suformuluoti kvalifikacinius reikalavimus, t. y. pasakyti, ką privalo mokėti ir gebeti informacinės sistemos teikiamomis paslaugomis besinaudojantys dalykinės srities specialistai, kad jie iš tiesų būtų pajęgūs tomis paslaugomis pasinaudoti. Ypatingas dėmesys skiriamas kompiuterinio raštingumo reikalavimais. Paprastai kvalifikaciniai reikalavimai yra formuluojami nurodant, kokį profesinį išsilavinimą ir išsilavinimą kompiuterinio raštingumo srityje privalo turėti vieną ar kitą pareigybę užimantis asmuo. Gali būti reikalaujama ir tam tikros praktinės patirties. Kvalifikaciniai reikalavimai turėtų būti įrašomi į pareigybines instrukcijas. Taigi, jei organizacijoje yra parengtos pareigybines instrukcijos ir jos yra tokios, kokios turėtų būti, tai kvalifikaciniai reikalavimai yra formuluojami peržiūrint pareigybines instrukcijas ir jas atitinkamai koreguojant. Suformulavus kvalifikacinius reikalavimus, juos reikia papildyti, konkretizuojant verslo reikalavimų lygmenyje aprašytus įgaliojimus ir nusakant prie kokių informacinių objektų turi teisę prieiti kiekviena iš pareigybų ir ką su tais objektais ji turi teisę daryti (peržiūrėti, kurti, keisti, perduoti kitiems ir t.t.). Be abejo, šios teisės turi būti suderintos su verslo reikalavimų lygmenyje aprašytais įgaliojimais. Jos yra nustatomos atitinkamomis verslo taisyklėmis.

Dar viena reikalavimų grupė yra siejama su informacinės sistemos panaudojamumu. Formulujant šiuos reikalavimus, yra aprašoma, ar atitinkama paslauga pareigybė naudojasi reguliarai, ar tik kartas nuo karto ir, jei naudojasi reguliarai, kokią darbo dienos dalį tam skiria. Nuo to priklauso, kokių kriterijumi reikia vadovautis, projektuojant interfeisą, per kurį yra užprašoma paslauga: teikti pirmenybę naudojimosi paprastumui ar paslaugos gavėjo darbo našumui. Beje, kartais gali būti svarbūs ir kokie nors kiti kriterijai. Taip pat reikia nurodyti, kokių mastu gaunamos paslaugos yra kritinės tos paslaugos gavėjo vykdomų verslo užduočių požiūriu. To reikia paslaugos prieinamumo reikalavimams suformuluoti.

Nors, formulujant vartotojų reikalavimus, yra aprašomas verslo sistemos viduje esančių dalykinės srities specialistų požiūris į kuriamą sistemą, jokiu būdu negalima pamiršti ir verslo sistemos klientų. Dar daugiau, jiems turi būti teikiama pirmenybė. Tarkime, nauja kompiuterizuota banko informacinė sistema būtų niekam tikusi, jeigu ji tik palengvintų darbą banko darbuotojams, bet niekaip nepagerintų klientų aptarnavimo. Kadangi sisteminis analitikas visų pirma bendrauja su kompiuterizuojamas organizacijos darbuotojais, o ne su jos klientais, tai pavojujus pajudėti netinkama linkme yra labai didelis. Nuo to pavojaus turėtų saugoti tinkamai sudarytas tikslų medis. Šiaip ar taip, būtina išnagrinėti, kokios informacinės sistemos paslaugos reikalingos verslo sistemos klientams ir suformuluoti atitinkamus reikalavimus. Jeigu dalykinės srities specialistams yra formuluojami kvalifikaciniai reikalavimai, t. y. nustatoma, ką jie turi gebeti ir mokėti, kad galėtų pasinaudoti informacinės sistemos teikiamomis paslaugomis, tai verslo sistemos klientams tokį reikalavimų kelti negalima. Atvirkščiai, reikia reikalauti, jog klientams teikiamos paslaugos būtų taip įgyvendintos, kad jomis galėtų pasinaudoti bet kuris vidurinį išsilavinimą turintis asmuo, išskaitant ir neigaliuosius (t. y. akluosius, nebylius ir kt.).

Reikia išnagrinėti ir verslo partnerių poreikius. Tiekėjams, verslo kuriamų produktų platintojams, valstybinėms įstaigoms ir kitiems verslo partneriams taip pat gali reikėti tam tikrų informacinės sistemos paslaugų. Todėl reikia nustatyti, kokiems verslo partneriams kokias paslaugas privalo teikti informacinė sistema, kokios yra kiekvieno verslo partnerio teisės ir kokius panaudojamumo reikalavimus jis kelia.

Specifinė paslaugų gavėjų klasė yra programiniai produktai. Kitaip tariant, realizuojant kokias nors kompiuterizuotas informacinės sistemos paslaugas gali prireikti naudotis kitomis kompiuterizuotomis paslaugomis. Be to, tiek verslo partneriai, tiek ir klientai paslaugų gali prašyti per aparatūrinius ar programinius tarpininkus. Todėl turi būti nustatyta, kuriomis informacinės sistemos teikiamomis paslaugomis gali naudotis ne tik žmonės, bet ir kompiuteriniai procesai, ir nustatytos tų procesų teisės. Beje, kai kurios informacinės sistemos, paprastai jos yra vadinamos techninėmis informacinėmis sistemomis, gali valdyti kokius nors įrenginius ar procesus arba stebeti jų darbą. Tokie įrenginiai bei procesai irgi turi būti traktuojami kaip speciali paslaugų gavėjų klasė ir jiems taip pat turi būti formuluojami atitinkami sistemos panaudojamumo reikalavimai bei nustatomos jų teisės.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kas?“ skamba šitaip:

„Kiekviena informacinės sistemos paslauga gali naudotis nurodytos tos paslaugos gavėjų grupės, išskaitant ir antrinius vartotojus. Visi paslaugos gavėjai gali naudotis paslauga tik nustatytu įgiliojimų ribose. Bet kuri sistemos paslauga turi būti projektuojama, atsižvelgiant į nurodytus jos naudojimo ypatumus, teikiant pirmenybę arba naudojimosi paslauga paprastumui, arba paslaugos gavėjo darbo našumui, arba kitiems nurodytiems kriterijams.. Projektuojant organizacijos darbuotojams skirtas paslaugas, reikia remties prielaida, kad tie darbuotojai tenkina nurodytus kvalifikacinius reikalavimus. Projektuojant klientams skirtas paslaugas, privalu vadovautis prielaida, kad paslaugos turi būti tokios, jog jomis gebėtų pasinaudoti bet kuris vidurinį išsilavinimą turintis asmuo, išskaitant ir neįgaliuosius.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Panašiai kaip ir dalykinės srities specialistai, produkto vartotojai gali būti skirstomi pagal įvairius kriterijus: pagal tai, kaip dažnai jie naudojasi produkту; pagal jų kompiuterinį raštingumą ir turimą darbo su kompiuteriu patirtį; pagal tai, kokios produkto galimybės juos domina; pagal jų įgiliojimus ir teises. Atsakant į klausimą „Kas?“, visų pirma vartotojus reikia suklasifikuoti pagal jų įgiliojimus bei teises ir suformuluoti reikalavimus, kokiai vartotojų grupei kokiomis produkto galimybėmis bus leidžiama naudotis. Antra, reikia nuspręsti, ar produktas bus pritaikytas tik pakankamą kompiuterinį raštingumą ir pakankamą darbo su kompiuteriais patirtį turintiems vartotojams, ar ir pradedantiesiems ir suformuluoti reikalavimus, kiek skirtinę patirtį turintiems vartotojams pritaikytų darbo su produktu lygmenų privalo palaikyti produktas. Trečia, reikia išsiaiškinti, kaip dažnai vartotojai naudosis produkту ir suformuluoti reikalavimus, kurie vartotojo interfeisai turi būti kuriami, teikiant pirmenybę jų suprantamumui ir paprastumui, o kurie, teikiant pirmenybę vartotojo darbo našumui. Ketvirta, reikia nuspręsti, ar produktas bus pritaikytas vartotojams su negalia ir suformuluoti atitinkamus reikalavimus. Penkta, turi būti numatyta, ar produkto bus galima naudotis tik per vartotojo interfeisus, ar ir kokiai nors kitaip būdais, tarkime, užprašant jo teikiamų paslaugų iš kompiuterinių programų vidaus, ir aprašyti visus tuos būdus. Gali būti reikalaujama įmontuoti produktą į kito produkto vidų, tarkime, į automobilį ar į skalbimo mašiną, ir tada jam turi būti suformuluoti specialūs panaudojamumo reikalavimai. Pagaliau, kai kuriems produktams yra svarbūs ir vadinamieji *antriniai vartotojai*, t. y. tokie vartotojai, kurie nors tiesiogiai ir nesinaudoja produkту, bet vienaip ar kitaip naudojasi jo kuriamais rezultatais. Atsakant į klausimą „Kas?“, reikia pasakyti, ar tokiu vartotoju yra ir kokius produkto rezultatų panaudojamumo reikalavimus jie kelia.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kas?“ šiuo atveju skamba šitaip:

„Produkto galimybėmis gali naudotis nurodytos vartotojų grupės, turinčios nurodytas teises ir keliančios nurodytus produkto panaudojamumo reikalavimus.“

2.5.6 Kur?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakant į klausimą „Kur?“, reikia patikslinti verslo reikalavimų lygmenyje suformuluotus darbo vietų reikalavimus. Tai reiškia, kad reikia pasakyti, kokios verslo užduotys kokiose darbo vietose bus vykdomos ir kokių informacinės sistemos paslaugų ten prireiks. Kitaip tariant, informacinės sistemos teikiamas paslaugas reikia lokalizuoti dalykinės srities specialistų pareigybiniėse darbo vietose, klientų aptarnavimo terminaluose bei kitiemis paslaugų gavėjams (verslo partneriams, kompiuteriniams procesams, įrenginiams) skirtuose prieigos prie sistemos mazguose. Atsakymas į klausimą „Kur?“ turi būti išplėstas, pasakant ne tik kur yra vykdomos vienos ar kitos verslo užduotys, bet ir kur (vietos prasme) yra saugomi vieni ar kiti informacinių objektai. Taip pat turi būti pasakyta, koks dalykinis komunikavimas (duomenų perdavimas, duomenų mainai, susirašinėjimas ir t.t.) reikalingas tarp darbo vietų.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kur?“ skamba šitaip:

„Atitinkamomis informacinės sistemos paslaugomis galima pasinaudoti tik nurodytose darbo vietose arba per nurodytus prieigos prie sistemos mazgus.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Skirtingoms vartotų kategorijoms gali reikėti skirtingų darbo vietų. Taigi, atsakant į klausimą „Kur?“, šiuo atveju reikia pasakyti, kiek skirtingų darbo vietų reikės dirbant su produkту, kokiomis produkto galimybėmis bus galima naudotis kiekvienoje iš jų, o taip pat, kur (produkto komponentų prasme) bus saugomi bendro naudojimo duomenys ir bendro naudojimo programos, jei tokią yra. Kitaip tariant, turi būti pasakyta, kad „darbo vietų“ prireiks ne tik vartotojams, bet ir tarkime, duomenų bazių, pašto ar kitokiemis serveriams.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kur?“ šiuo atveju skamba šitaip:

„Atitinkamomis produkto galimybėmis galima naudotis tik nurodytose darbo vietose. Iš tų darbo vietų produkto galima pasinaudoti tik tuo atveju, jei yra įrengti ir kiti produktui veikti reikalingi funkciniai mazgai.“

2.5.7 Kada?

Sistemoms, kuriamoms konkrečiam užsakovui. Atsakant į klausimą „Kada?“ yra formuluojami verslo užduočių našumo reikalavimai. Jie išvedami iš verslo reikalavimų lygmenyje suformuluotų našumo reikalavimų. Jei, tarkime, aprašant įvykių apdorojimo reikalavimus, ten buvo nustatytos maksimaliai leistinos verslo transakcijų trukmės, tai dabar pagal jas reikia nustatyti kiekvienos tų transakcijų vykdymui reikalingų verslo užduočių trukmes. Tokie patys ribojimai turi būti suformuluoti ir visoms kitoms verslo užduotims. Analogiškus reikalavimus reikia suformuluoti ir klientų, verslo partnerių bei kitų sistemas paslaugų gavėjų vykdomoms užduotims.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kada?“ skamba šitaip:

„Informacinė sistema privalo užtikrinti, kad dalykinės srities specialistų, verslo klientų, verslo partnerių ir kitų sistemos paslaugų gavėjų vykdomos užduotys bus įvykdytos, neviršijant nurodytų laiko ribojimų.“

Sistemoms, kuriamoms norint parduoti jas rinkoje. Sistemoms, kurios yra kuriamos tikslu parduoti jas rinkoje, į klausimą „Kada?“ yra atsakoma labai panašiai. Skirtumas tiktais tokis, kad laiko ribojimai šiuo atveju nustatomi ne verslo užduotims, bet produkto vykdomoms vartotojui prasmingoms užduotims. Savaime aišku, kad tie ribojimai yra nustatomi vadovaujantis verslo reikalavimų lygmenyje nustatytais produkto galimybų laiko ribojimais.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kada?“ šiuo atveju skamba šitaip:

„Produktas privalo užtikrinti, kad vartotojams prasmingos užduotys būtų vykdomos, neviršijant nurodytų laiko ribojimų.“

2.5.8 Baigiamosios pastabos

Vartotojo reikalavimų svarbą pripažista visų reikalavimų inžinerijos vadovėlių autoriai. Tačiau, tų reikalavimų formulavimo problemas traktuojama gana skirtingai. Pavyzdžiui, Ian Sommerville ir Pete Sawyer mano, kad bene svarbiausia vartotojų reikalavimų formulavimo problema yra skirtingu požiūriu integravimas. Jie rašo:

„Daug kas daro įtaką kompiuterizuotos sistemas reikalavimams. Prie tokiu veiksnių priskiriami tiesioginiai sistemas vartotojai, bet arpiškai vykdantys procesus, kuriems palaikyti ir yra kuriama sistema, vadovai, tie organizacijos darbuotojai, kurių darbą keičia sistema, nors jie tiesiogiai ja ir nesinaudoja, ir sistemas užsakovai, perkantys tą sistemą.“

Visi jie ir daugelis kitų yra potencialūs sistemas reikalavimų šaltiniai ir kiekvienas iš jų turi savą požiūrį į tai, kokias paslaugas turėtų teikti sistema ir kokiui būdu ji tą turėtų daryti. Reikia pripažinti tą požiūrių įvairovę ir renkant reikalavimus neignoruoti nei vieno požiūrio.“[107]

David C. Hay nuomone, svarbiausia yra perprasti operacinus vartotojų poreikius, o ne stengtis iškvosti iš jų, ko jie tikisi iš kuriamos sistemas. Jis rašo:

„Kalbėkis su žmonėmis. Pasitenk pasikalbėti su kiekvienu dalykinės srities specialistu, kurį tu žinai. Klausinėk jų, kaip jie dirba. Neklausk, ko jie norėtų iš naujos sistemas, neklausk bent jau iki pokalbio pabaigos. Reikia susidaryti aiškų vaizdą apie jų veiklą ir poreikius, vaizdą, kuris nėra iškreiptas išankstinių nuostatų, kaip sistema jiems turėtų padėti.“[47]

Panašios nuomonės laikosi ir Karl Wiegers:

„Vienintelis būdas išvengti nepasiteisinusių lūkesčių, produkto, kurį tikisi gauti užsakovas, ir vykdytojų kuriamo produkto disharmonijos yra įtraukti užsakovą. Negalima pradėti rašyti programą vien tik paprasčiausiai paklausus kelių užsakovo atstovų, ko gi jiems reikia. Jei vykdytojas sukurs tiksliai tai, ko iš pradžių jo paprašė užsakovas, tai jam tikriausiai teks viską perdaryti, nes dažniausiai užsakovas nežino, ko jam iš tiesų reikia.“

Produkto galimybės, kurias vartotojai nusako kaip savo "norus", visiškai nebūtinai sutampa su tuo funkcionalumu, kurio jiems reikia naudojantis naujuoju produktu savo užduotims vykdyti. Norėdamas tiksliau suvokti vartotojų poreikius, analitikas privalo surinkti vartotojų nuomonės, jas išanalizuoti ir išgryninti ir nuspresti, ką gi reikia sukurti, kad palengvinti vartotojams jų darbą. Būtent analitikas privalo suformuluoti, kokios sistemos galimybės yra reikalingos ir kokias kitas savybes ji turi turėti, ir pateikti tą informaciją visoms sistema suinteresuotoms šalims. Tai iteracinis procesas ir tam reikia laiko. Jei jūs neskirsite pakankamai laiko susitarti dėl bendros nuomonės, dėl bendros kuriamo produkto vizijos, to rezultatais tikrai bus būtinybė perdirbinėti produkta, sužlugdyti terminai ir užsakovo nepasitenkinimas."[117]

Be to, praktiškai visi autoriai pabrėžia, kad užsakovai paprastai nesuprantą vartotojų reikalavimų formulavimo svarbos, yra užsiémę ir nenori gaišti tam laiko. Todėl bene sunkiausia sisteminio analitiko užduotis yra išaiškinti užsakovui potencialias tokią nuostatą pasekmes ir įtikinti jį, kad dalykinės srities specialistai, kad ir kaip jie bebūtų užsiémę, privalo reikalavimų formulavimui skirti pakankamai laiko ir dėmesio. Tačiau net ir tais atvejais, kai tą padaryti pavyksta, sėkmė dar nėra garantuota, nes organizacijos darbuotojai yra sutelkę savo dėmesį į einamąsių problemas ir retai kada susimąsto apie savo ilgalaikius informacinius poreikius. Vartotojo reikalavimams išsiaiškinti Gordon Davis ir Margrethe Olson siūlo [22] naudotis net keturiais skirtingais būdais: apklausti vartotojus; „ištraukti“ reikalavimus iš šiuo metu naudojamos informacinių sistemų; ivesti reikalavimus iš verslo procesų charakteristikų ir „atrasti“ reikalavimus eksperimentuojančiuose kuriama sistemoje. Paskutinysis reikalavimų aiškinimosi būdas reikalauja daug laiko ir lėšų. Taip pat jis kainuoja daug nervų vartotojams. Todėl juo naudotis nerekomenduotina. Tačiau kai kurios Lietuvos organizacijas kompiuterizuojančios bendrovės šiuo būdu naudojasi gana plačiai arba, kitaip tariant, pradeda diegti „žalias“ sistemas ir laukia kol jos „savaime susikratys“. Aišku, šitaip dirbtį įmanoma tik labai nebrandžioje rinkoje.

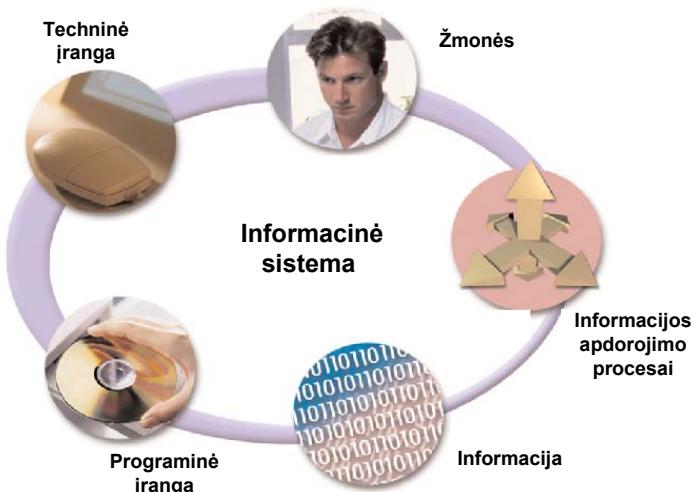
2.6 Informacinių sistemų reikalavimai

2.6.1 IS reikalavimų formulavimo ypatumai

Kuriant konkrečiam užsakovui skirtas programų sistemas, tos sistemos dažniausiai yra skirtos užsakovo informacinei sistemai kompiuterizuoti. Todėl, net ir turint jau suformuluotus vartotojo reikalavimus, pradėti formuluoti reikalingų programų sistemų reikalavimus dar nėra galima. Prieš tai reikia suformuluoti tarpinius, informacinių sistemų reikalavimus. Dalis rinkoje parduoti skirtų produktų, vadinamieji verslo procesų valdymo (angl. *ERP*⁵) paketai, taip pat yra kuriami informaciniems sistemoms palaikyti. Todėl jiems irgi yra formuluojami informacinių sistemų reikalavimai. Kadangi šie paketai skirti ne konkrečiam užsakovui ir yra manoma, kad jais naudosis daugelis skirtingo pobūdžio organizacijų, tai informacinių sistemų reikalavimų negalima suformuluoti remiantis konkrečios verslo sistemos poreikiais. Verslo procesų valdymo paketams informacinių sistemų reikalavimus formuluojant ekspertai, remdamiesi, jų nuomone, pažangiaisiais teoriniaisiais informacinių sistemų modeliais. Be abejo, tokie modeliai nėra, kaip sakoma, iš piršto laužti. Jie kuriami apibendrinant daugelio skirtingų organizacijų patirtį ir paprastai yra pritaikyti tam tikros rinkos (Azijos šalių, JAV, Vakarų Europos šalių, Rytų Europos šalių ir kt.)

⁵ ERP yra anglisko termino *enterprise resource planning* trumpinys.

poreikiams. Reikalaujama, kad, įsigijusi tokį paketą, organizacija pertvarkytų savo informacinę sistemą, o tuo pačiu ir savo darbo procesus, pagal įsigyto paketo realizuojamą modelį. Tai motyvuojama tuo, kad kartu su paketu bus įdiegti nauji, progresyvūs darbo metodai ir organizacija dėl to tik išloš. Tačiau, neretai tikrovė esti šiek tiek kitokia. Apibendrintas teorinis IS modelis negali atsižvelgti į konkrečios organizacijos poreikius bei jos vadovybės darbo stilių ir, nors verslo procesų valdymo paketai numato įvairius parametrizavimo, adaptavimo bei plėtros mechanizmus, pritaikyti tą modelį organizacijos poreikiams ne visuomet pavyksta. Kita vertus, net ir tais atvejais, kai paketas iš tiesų atneša pažangius darbo metodus, įdiegti juos organizacijoje gali būti labai sunku ar netgi apskritai neįmanoma. Organizacija turi būti tam pribrendusi. Kaip sakoma, aukščiau bambos neiššoksi. Reikalas tas, kad diegiant bet kurią inovaciją iškyla vadinamoji *inovacinių slenksčių problema*. Tai reiškia, kad organizacijos darbuotojai turi vienus mokėjimus, gebėjimus, įgūdžius bei įpročius, o inovacija reikalauja kitokių mokėjimų, gebėjimų, įgūdžių bei veikimo būdų. Tie skirtumai vadinami inovacinių slenksčiais. Jei skirtumai yra pakankamai dideli, perkopti per inovacinius slenksčius gali prireikti ne vienerių metų, nekalbant jau apie tai, kiek tam prireiks lėšų. Tai pagrindinė priežastis, dėl ko verslo procesų valdymo paketą diegimas neretai esti nesėkmingas. Panašiai gali atsitikti ir kuriant programų sistemą konkrečiam užsakovui, jei informacinės sistemos reikalavimai nebus suformuluoti ir programų sistemą kuriantys inžinieriai turės omenyje vieną informacinės sistemos modelį, o užsakovo organizacija dirbs kitaip. Nepaisant šio akivaizdaus fakto, per kelis dešimtmečius sukauptos kompiuterizuotų informacinių sistemų kūrimo patirties ir to, kad šiandien praktiškai visuose pasaulio universitetuose yra skaitomi informacinių sistemų ir informacinių sistemų inžinerijos kursai, informacinių sistemų reikalavimams vis dar yra skiriama per mažai dėmesio. Retai kuris vadovėlis išsamiau aptaria informaciniės sistemos reikalavimus ir jų formulavimo problemas. Viena iš priežasčių yra tai, kad organizacijų informaciniės sistemos, išskyrus verslo procesų valdymo paketą diegimo atvejus, labai retai yra pakankamai formalizuotos ir projektuojamos išreikštiniu būdu. Paprastai jos klostosi savaime per daugelį metų. Todėl daugumos vadovėlių autoriai vadovaujasi prielaida, kad informaciniės sistemos reikalavimų negalima atskirti nuo verslo reikalavimų ir juos reikia formuluoti kartu. Mūsų nuomone, ši prielaida yra klaidinga, nes tai pačiai verslo sistemai galima sukurti skirtinges informacines sistemas. Taip pat yra galima gana radikaliai pertvarkyti organizacijos informacinię sistemą, visiškai nekeičiant jos palaikomos verslo sistemos. Informaciniė sistema yra savarankiška verslo sistemos dalis ir jos pobūdis priklauso ne tik nuo verslo sistemos poreikių, bet ir nuo organizacijos vadybos metodų, jos vadovybės darbo stiliaus bei nuo vadinamosios *korporacinių kultūros*. Be to, minėtoji prielaida yra ir logiškai ydinga, nes vartotojų poreikiai, o tuo pačiu ir jų keliami informaciniės sistemos reikalavimai yra formuluojami jau suformulavus verslo reikalavimus ir, jei šie reikalavimai nusakys, kokia turi būti informaciniė sistema, tai kokia prasmė bus aiškintis su vartotojais, kokia gi ši sistema jų požiūriu turėtų būti. Aišku, galima teigti, kad suformulavus vartotojo reikalavimus reikia grąžti atgal ir keisti verslo reikalavimus. Tačiau vargu ar šitaip elgtis yra verta. Mūsų nuomone, tikslingiau yra informaciniės sistemos reikalavimus atskirti nuo verslo reikalavimų ir formuluoti juos atskirai. Beje, šitaip mano ir John Zachman[122].



29 pav. Informacinė sistema (paimta iš [97])

Tačiau, ką gi mes vadiname *informacinėmis sistemomis*⁶? Atsakymas gali būti labai paprastas: informacijos rinkimo, kaupimo, organizavimo, saugojimo ir apdorojimo sistemas. Tačiau, pradėjus gilintis į šio termino prasmę ir jo vartoseną, paaiškėja, kad viskas yra ne taip paprasta. Programų sistemų inžinieriai ir programuotojai yra linkę susiaurinti šio termino prasmę ir informacinėmis sistemomis vadina tik kompiuterines informacines sistemos. Dažniausiai, informacine sistema vadinama kokia nors duomenų bazė ir joje saugomiems duomenims apdoroti skirtų programų rinkinys. Panašia, bet šiek tiek platesne prasme ši terminą vartoja inžinieriai, medikai bei kai kurių kitų sričių specialistai. Jie techninėmis informacinėmis sistemomis, medicinos informacinėmis sistemomis ar kitokiomis informacinėmis sistemomis vadina programų rinkinius, skirtus kokiems nors specialaus pobūdžio duomenims, nebūtinai saugomiems duomenų bazėse, apdoroti. Iš tiesų ir vienu ir kitu atveju tokios sistemos vadintinos tam tikros paskirties *duomenų apdorojimo sistemomis* (angl. *data processing system*). Kalbant apie organizacijų informacines sistemas, o šioje knygoje mes kalbame būtent apie jas, terminas *informacinė sistema* suprantamas šiek tiek labiau vienareikšmiškai, bet gerokai plačiau. Didžioji dauguma specialistų sutaria, kad, tariant Gordon Davis ir Margrethe Olson žodžiais, tai sistema

“teikianti informaciją, reikalingą kasdieninei operacinei veiklai vykdyti, organizacijai valdyti ir reikiams sprendimams priimti.” [22]

Šitaip suprantama informacinė sistema apima ne tik informacijos apdorojimo bei komunikavimo technologijas, kaip kompiuterizuotas, taip ir ne, bet ir (29 pav.):

- informacijos saugyklas,
- informacijos srautus ir jų valdymą,
- informacijai apdoroti ir perduoti naudojamus procesorius, išskaitant ne tik programų sistemas, kompiuterius, telefonus, kopijavimo aparatus bei kitą techninę įrangą, bet ir informaciją apdorojančius bei tvarkančius žmones,

⁶ Lietuvoje vis dar yra diskutuojama, kaip versti angliskąjį terminą *information system*: *informacijos sistema* ar *informacinė sistema*. Mes pasisakome už pastarąjį vertimą, nes yra kalbama ne apie informacijos organizavimo, o apie jos rinkimo, organizavimo, kaupimo, saugojimo ir apdorojimo sistemą. Tarybiniais metais, mėgdžiojant rusiškąją terminiją, buvo vartojoamas terminas *automatizuota valdymo sistema*. Tačiau, pradėjus dominuoti anglų kalbai, šis terminas tapo nebevartojamas. Be to, jo prasmė buvo netiksli ir per daug siaura.

- visa tai aptarnaujantį bei prižiūrintį personalą bei viso to organizavimo mechanizmus.

Natūralu, kad, viena vertus, suprojektuoti ir sukurti tokią sistemą yra labai sudėtinga. Kita vertus, Tačiau ar kitokia informacinė sistema, galbūt nekompiuterizuota ir labai primityvi, veikia bet kurioje organizacijoje. Netgi oficialiai registruoto ūkininko ūkyje. Šiais laikais jokia legali veikla negali būti vykdoma vienaip ar kitaip neapdorojant kokių nors informacinių objektų. Didžiojoje daugumoje Lietuvos organizacijų informacinės sistemos niekuomet nebuvo projektuotos ir kurtos. Jos susiklostė savaime. Projektai, kuriuose bandoma vien tik perprasti ir kompiuterizuoti tokias savaime susiklosčiusias informacines sistemas, gali palengvinti bei pagreitinti organizacijos darbuotojų darbą, padidinti informacijos patikimumą bei korekтиškumą, bet jie negali išspręsti jokių realių verslo problemų. Tam reikia pradėti nuo verslo sistemos analizės, suformuluoti verslo reikalavimus, optimizuoti informacinę sistemą ir iš tiesų pritaikyti ją verslo poreikiams. Visų informacinę sistemą palaikančių programų sistemų bei informacinių ir komunikavimo technologijų reikalavimai turi būti formuluojami informacinės sistemos reikalavimų kontekste, nes tos technologijos ir sistemos yra informacinės sistemos komponentai. Šitaip suprantama informacinė sistema apima visą organizaciją, išskaitant visas joje veikiančias duomenų ir informacijos apdorojimo sistemas, kurias, kaip jau minėjome, taip pat dažnai vadina informaciniems sistemomis. Ji organizacijoje yra tikta viena ir informacijos apdorojimo aspektu IS specifikacija aprašo visą organizaciją. Tačiau tokią specifikaciją sukurti yra labai sunku, praktiskai beveik neįmanoma, nors kai kurie autorai ir siūlo informacines sistemas kurti būtent šitaip. Be to, tokia specifikacija, netgi jei ją ir pavyktų sukurti, labai greitai pasentų ir taptų mažai naudinga. Tai yra viena iš priežasčių, kodėl verslo sistemos pertvarkymas atliekamas ne iš karto, o realizuojant atskirus projektus, kiekvienas iš kurių įgyvendina tam tikrą verslo tikslų medžio pomedį. Taigi, kiekviename atskirame projekte informacinės sistemos reikalavimai taip pat yra formuluojami tiktais tai jos daliais, kuri reikalalinga tuo pomedžiu numatytiems tikslams realizuoti. Aišku, skaidant informacinę sistemą į atskiras dalis, iškyla tų dalių integravimo problema. Šiai problemai išspręsti vien verslo reikalavimų nepakanka. Formuluojant atitinkamas informacinės sistemos dalies reikalavimus, juos privalu derinti tiek su reikalavimais, suformuluotais vykdant ankstesnius projektus, tiek ir su reikalavimais, formuluojamais lygiagrečiai vykdomuose projektuose. Tai sudėtingas procesas, galintis pareikalauti dar kartą pertvarkyti vykdant ankstesnius projektus jau pertvarkytas IS dalis. Priminsime, kad verslo tikslų medžiui realizuoti reikalingi informacinės sistemos pertvarkymai apima ir jos kompiuterizavimą. Pakartotinų pertvarkymų bandoma išvengti dekomponuojant tikslų medži į kuo mažiau vienas nuo kito priklausančius pomedžius ir šitaip minimizuojant atskirai kuriamų informacinės sistemos dalių sankibą. Be to, informacinės sistemos pertvarkymai paprastai yra pradedami pertvarkant vadinamuosius organizacijos registrus ir kadastrus, t. y. duomenų saugyklas, kuriose yra saugomi tik tie informacinių objektų duomenys, kurie yra bendri visoms informacinės sistemos dalims. Tai taip pat sumažina informacinės sistemos dalių sankibą, nes, pritaikant turinių atskyrimo principą, bendro pobūdžio duomenys (baziniai rodikliai) ir duomenys, reikalingi tiktais vieniems ar kitiems specifiniams tikslams pasiekti (roliniai rodikliai) yra atskiriami.

Kuo gi skiriasi vartotojo reikalavimai ir informacinės sistemos reikalavimai? Vartotojo reikalavimai aprašo, kokių informacinės sistemos paslaugų reikia dalykinės srities specialistams ir kitiems jos vartotojams. Nors mes juos ir vadiname reikalavimais, iš tiesų tai yra tik vartotojams reikalingų informacinės sistemos

paslaugų aprašas. Čia trūksta dar daugelio detalių, reikalingų tam, kad informacinę sistemą būtų galima pradėti įgyvendinti. Tas detales aprašo informacinės sistemos reikalavimai, atspindintys informacinių sistemų inžinieriaus požiūrį ir galutinai susiejantys verslo ir programų sistemų bei informacinių ir komunikavimo technologijų reikalavimus į vieną darnią visumą. Formuluojant pertvarkytos informacinės sistemos reikalavimus, be abejo, reikia detalai išsiaiškinti esamą informacinię sistemą, iškaitant joje naudojamas programų sistemas bei informacines ir komunikavimo technologijas, ir gerai perprasti nuostatas, kuriomis esama informacinių sistema yra grindžiama. Taip pat reikia surinkti ir išsiaiškinti taisykles, vienaip ar kitaip ribojančias informacijos kaupimo metodus, informacijos saugyklų organizavimą, informacijos apdorojimo metodus, informacinių srautų organizavimą, perdavimą bei valdymą ir kitus darbo su informacija ypatumus. Išsiaiškinti ir suprasti esamą situaciją yra ne mažiau svarbu, negu turėti aiškią pertvarkytos sistemos viziją.

Kadangi, išskyrus verslo procesų valdymo paketus, rinkoje parduoti skirtiems produktams informacinės sistemos reikalavimų formuluočių nereikia, tai tolimesniuose šio poskyrio skyreliuose apie rinkoje parduoti skirtus produktus nekalbėsime.

2.6.2 Kodėl?

Atsakant į klausimą “Kodėl?”, yra formuluojami tikslai, kurių siekiama kuriant informacinių sistemų. Šie tikslai formuluojami, viena vertus, vadovaujantis projektui parinktu tikslų medžio pomedžiu ir, kita vertus, vadovaujantis vartotojo reikalavimais. Kadangi informacinių sistemos yra produktas, tai tikslai yra aprašomi suformuluojant to produkto viziją ir detalizuojant tą viziją sistemos galimybių medžiu⁷. Be to, išreikštiniu būdu yra suformuluojamos organizacijoje galiojančios darbo su informaciją taisykles, vienaip ar kitaip ribojančios galimybių medžiu numatyty galimybių įgyvendinimą.

Pageidautina, jog informacinių sistemos turėtų tokias galimybes, kad būtų patenkinti visi vartotojų operacinių poreikių ir visi jų pageidavimai. Tačiau dėl laiko ir lėšų ribojimų šito pasiekti dažniausiai yra neįmanoma. Todėl, formuluojuojant informacinius reikalavimus, vartotojų reikalavimus tenka *prioreitizuoti*, t. y. suskirstyti juos į grupes pagal jų svarbą. Visų pirmą reikia atskirti vartotojų operacinius poreikius ir jų pageidavimus. Tai nereiškia, kad pageidavimai yra mažiau svarbūs ir juos galima atmesti. Pageidavimai atspindi organizacijos vadovybės darbo stilių, naudojamų vadybos procedūrų ypatumus, organizacijos korporacinių kultūrų ir daugelį kitų svarbių dalykų. Todėl juos reikia detalai išanalizuoti, suskirstyti į grupes pagal jų svarbą ir žiūrėti, kokią jų dalį realiai galima įgyvendinti. Operacinių poreikių turėtų būti tenkinami visi. Jei paaiškėja, kad šito pasiekti neįmanoma, reikia grįžti prie vartotojo reikalavimų lygmenyje suformuluoto atsakymo į klausimą “Kodėl?”, peržiūrėti projektui parinktą tikslų medžio pomedį, susiaurinti projekto apimtį ir tuo pačiu kai kurių operacinių poreikių tenkinimą atidėti vėlesniams laikui. Šitaip atrinkus vartotojų reikalavimus, pagal juos yra formuluojama naujos informacinių sistemos vizija, joje išryškinant svarbiausių senos ir naujos sistemos skirtumus.

Pavyzdys

Bibliotekos verslo sistema operuoja dviem verslo objektais: knygomis ir skaitytois. Knygos yra įsigyjamos, skolinamos skaitytojams ir nurašomos.

⁷ Nors mes vartojaame terminus *tikslų medis* ir *galimybių medis*, grafų teorijos prasme tai nėra tikri medžiai, nes du tikslai gali turėti bendrą pomedį ir dvi skirtinges galimybės gali būti įgyvendinamos pasinaudojant kokia nors bendra žemesnio lygmens galimybe.

Paklausias knygas norintys gauti skaitytojai yra statomi į eilę. Tai verslo užduotys. Knygų išdavimas skaitytojams ir jų gražinimas bibliotekai yra verslo transakcijos. Bibliotekos skaityklose skaitytojai gali imti knygas tiesiai iš lentynų ir naudotis jomis skaityklos patalpose.

Bibliotekos informacinė sistema operuoja trijų tipų informacinius objektais: bibliografiniai knygų aprašais, įrašais apie knygas ir įrašais apie skaitytojus. Skaitytojai turi skaitytojų pažymėjimus. Tai taip pat yra informacinius objektas.

Įsigytos knygos yra registruojamos. Tai reiškia, kad apie kiekvieną įsigytą knygą yra sukuriamas ją aprašantis įrašas ir įrašomas į bibliotekos turimų knygų registrą. Tai vadinama knygų inventorizavimu. Nurašomas knygos yra išregistruojamos, t. y. įrašai apie nurašytas knygas yra paskelbiami negaliojančiais. Inventorizavus įsigytą knygą, yra sudaromas jos bibliografinis aprašas. Tas aprašas yra įtraukiamas į bibliotekos katalogą. Nurašant knygą, bibliografinis knygos aprašas iš bibliotekos katalogo yra pašalinamas.

Nauji skaitytojai yra registruojami, t. y. apie kiekvieną naują skaitytojų yra sukuriamas atitinkamas įrašas, kuris yra įrašomas į bibliotekos skaitytojų registrą. Registruotam skaitytojui išduodamas skaitytojo pažymėjimas. Asmeniui nustojus būti skaitytojui, jis yra išregistruojamas, t. y. skaitytojų registre apie jį saugomas įrašas yra paskelbiamas negaliojančiu.

Skaitytojui pageidaujant gauti knygą, kuri yra išduota kitam skaitytojui, jis yra įrašomas į tos knygos laukiančių skaitytojų eilę. Išduodant jam knygą, kurios jis laukė eilėje, skaitytojas iš eilės yra išbraukiamas.

Visos vykdomos verslo transakcijos (t. y. knygų išdavimas ir gražinimas) taip pat yra registruojamos. Ileidžiant asmenį į skaityklą, tikrinama ar jis turi skaitytojo pažymėjimą.

Visoms skaitytojams išduotos knygos yra registruojamos paskolintų knygų registre. Gražinus knygą, ji iš registro yra išbraukiamas.

Biblioteka turi informacinię vitriną, kurioje skelbiama informacija apie naujai gautas knygas ir įdomiausių knygų recenzijos.

Visa tai yra informacinių sistemos teikiamos paslaugos. Tų paslaugų reikia atitinkamoms verslo užduotims vykdysti.

Bibliotekos organizacijos atmintį sudaro penki registrai (knygų, skaitytojų, eilėje laukiančių skaitytojų, įvykdytų transakcijų ir paskolintų knygų), knygų katalogas ir bibliotekos informacinė vitrina. Knygų ir skaitytojų registrai yra pirminiai. Juose saugomi tie duomenys apie knygas ir skaitytojus, kurie reikalingi vykdant bet kurias verslo užduotis. Eilėje laukiančių skaitytojų, transakcijų ir paskolintų knygų registrai yra antriniai, t. y. jų įrašuose yra daromos nuorodos į atitinkamus pirminių registrų įrašus. Knygų katalogas ir informacinė vitrina taip pat yra antrinės informacijos saugyklos, nes jose saugomi informacinių objektų turi nuorodas į knygų registrą.

Informacinių sistemos užduotis vykdant procesorių operuoja bibliotekos organizacinių atmintyje saugomais informacinius objektais. To procesoriaus dalis yra reikiama knygų paieškos kataloge sistema (kalbant, programų sistemų inžinerijos terminais, *paieškos kataloge mašina*)

Bibliotekos informacinė sistema yra nekompiuterizuota. Bibliotekos registrai yra realizuoti kaip atitinkamos registravimo knygos, knygų katalogas – kaip atitinkamai suklasifikuotas kortelių su knygų bibliografiniais aprašais rinkinys, informacinė vitrina – kaip po stiklu esanti skelbimų lenta, kurioje kabinami reikiams skelbimai. Informacinės sistemos procesorių realizuoja bibliotekos darbuotojai, rankiniu būdu tvarkantys registravimo knygas ir knygų katalogą. Kitaip tariant, verslo užduotis vykdantys darbuotojai jiems reikalingas informacines paslaugas teikia patys sau. Paieškos kataloge mašina realizuota kaip dvi medinės spintos su stalčiais. Vienos spintos stalčiai pažymėti abécélės raidėmis, kiekviename stalčiuje sudėtos kortelės su ta raide prasidedančių autorų parašytų knygų bibliografiniais aprašais. Stalčiaus viduje kortelės sudėtos abécéline tvarka. Kitos spintos stalčiai pažymėti atitinkamais UDK⁸ kodais. Stalčiuje sudėtos kortelės su ta tema parašytų knygų bibliografiniais aprašais. Stalčiaus viduje kortelės sutvarkytos pagal žemesnius UDK klasifikavimo medžio lygmenis. Jėjimą į bibliotekos skaityklas kontroliuoja bibliotekos darbuotojas. Jis tikrina įeinančių asmenų skaitytojų pažymėjimus ir žiūri, ar jie nėra paskelbti negaliojančiais. Jis taip pat stebi, ar iš skaityklų išeinančios skaitytojai nebando išsinešti atviruose fonduose saugomų leidinių.

Esamą informacinę sistemą aprašo šitoks funkcinių galimybių medis:

0. Galimybė tvarkyti visą darbui su skaitytojais ir darbui su knygomis reikalingą informaciją
 1. Galimybė tvarkyti knygų registrą
 - 1.1. Galimybė registruoti naujai įsigytas knygas
 - 1.2. Galimybė išregistruoti nurašytas knygas
 2. Galimybė tvarkyti knygų katalogą
 - 2.1 Galimybė papildyti katalogą naujai įsigytų knygų bibliografiniais aprašais
 - 2.2. Galimybė pašalinti iš katalogo nurašytų knygų bibliografinius aprašus
 - 2.3. Galimybė vykdyti reikiamas knygos paiešką kataloge
 3. Galimybė tvarkyti skaitytojų registrą
 - 3.1. Galimybė registruoti naujus skaitytojus
 - 3.1.1. Galimybė registruoti skaitytojų skaitytojų registre
 - 3.1.2. Galimybė išduoti skaitytojui skaitytojo pažymėjimą
 - 3.2. Galimybė išregistruoti skaitytojus
 - 3.2.1. Galimybė paskelbti skaitytojų registro įrašą apie skaitytoją negaliojančiu
 - 3.2.2. Galimybė paskelbti skaitytojo pažymėjimą negaliojančiu
 4. Galimybė tvarkyti bibliotekos transakcijų registrą
 - 4.1. Galimybė registruoti knygų išdavimus skaitytojams
 - 4.2. Galimybė registruoti knygų grąžinimą
 5. Galimybė tvarkyti paskolintų knygų registrą
 - 5.1. Galimybė registruoti paskolintas knygas
 - 5.2. Galimybė išregistruoti grąžinamas knygas
 6. Galimybė tvarkyti eilėse laukiančių skaitytojų registrą

⁸ UDK – universalusis dešimtainis klasifikatorius.

- 6.1. Galimybė statyti į eilę paklausią knygą norintį gauti skaitytoją
- 6.2. Galimybė pašalinti iš eilės norimą knygą gavusį skaitytoją
7. Galimybė tvarkyti bibliotekos informacinię vitriną
 - 7.1. Galimybė skelbti informaciją apie naujai gautas knygas
 - 7.2. Galimybė skelbti trumpas įdomiausių knygų recenzijas
 - 7.3. Galimybė šalinti iš vitrinos pasenusią informaciją
8. Galimybė kontroliuoti, kad į bibliotekos skaityklas patektų tik galiojančius skaitytojo pažymėjimus turintys asmenys

Ši medžiagą papildo tokios bibliotekos informacijos tvarkymo taisyklės:

1. Knygų katalogas visą laiką turi būti aktualus, t. y. jame turi būti visų knygų registre įregistruotų knygų bibliografiniai aprašai ir negali būti iš knygų registro išregistruotų knygų aprašų.
2. Įrašuose apie transakcijas nuorodos gali būti daromas tik į knygų registre įregistruotas knygas ir tik į skaitytojų registre įregistruotus skaitytojus.
3. Transakcijų registre darant įrašą apie knygos išdavimą, atitinkamas įrašas turi būti daromas ir paskolintų knygų registre. Registruojant knygos grąžinimą, atitinkamas įrašas iš paskolintų knygų registro turi būti pašalintas.
4. Įrašuose apie eilėse laukiančius skaitytojus nuorodos gali būti daromas tik į knygų registre įregistruotas knygas ir tik į skaitytojų registre įregistruotus skaitytojus.
5. Informacija apie naujai gautas knygas bibliotekos vitrinoje skelbiama iš karto po to, kai jų bibliografiniai aprašai yra įtraukiami į knygų katalogą. Informacija pašalinama iš vitrinos po dviejų savaičių.
6. Skelbiamos tik tų įdomiausių knygų recenzijos, kurių bibliografiniai aprašai yra bibliotekos knygų kataloge. Recenzijos šalinamos iš vitrinos po dviejų savaičių.

Verslo sistemos analizės metu išryškintos šios verslo problemos:

- laikas, praeinantis nuo knygos įsigijimo iki galimybės ją išduoti skaitytojui yra per ilgas;
- per ilgos norinčių gauti ar gražinti knygas skaitytojų eilės;
- sulaukus eilės norimai knygai gauti, skaitytojas sužino apie tai tik per kelias dienas;
- katalogu ir informacine vitrina galima naudotis tik bibliotekos darbo valandomis, prie jų susidaro spūstis;
- norint rasti kataloge norimą knygą, tenka sugaišti per daug laiko, be to, kataloge lieka kai kurių bibliotekos nebeturimų (nurašytų, negražintų, pavogtų) knygų kortelės ir trūksta kai kurių turimų knygų kortelių;
- į bibliotekos skaityklas dažnai praeina asmenys, neturintys tam teisės, iš viešai prieinamų fondų dingsta daug leidinių.

Visų tų problemų priežastis yra prasta bibliotekos informacinė sistema.

Suformulavus ir išanalizavus vartotojų reikalavimus, buvo suformuluota šitokia naujos bibliotekos informacinės sistemos vizija:

„Produktas „Naujoji bibliotekos informacinė sistema“ skirtas viešosioms bibliotekoms, kurioms reikia tvarkyti visą jų darbui su skaitytojais reikalingą informaciją. Tai kompiuterizuota sistema, kuri automatizuotai tvarko informaciją apie skaitytojus, informaciją apie turimas knygas, informaciją apie knygų išdavimą bei gražinimą ir bibliotekos informacinę vitriną. Skirtingai negu senoji informacinė sistema, šis produktas leidžia naudotis bibliotekos katalogu ir peržiūrėti jos informacinę vitriną per internetą, tą leidžia daryti bet kuriuo paros metu, sutrumpina reikiama knygų paiešką kataloge iki kelių sekundžių ir leidžia skaitytojams užsisakinėti norimas knygas per internetą. Naujoji bibliotekos informacinė sistema, panaudodama knygų brūkšninius kodus, pakeisdama tradicinius skaitytojų bilietus kortelėmis su įmontuotais luistais ir panaudodama skenavimo įrenginius, vidutiniškai daugiau kaip 10 kartų sutrumpina knygų išdavimo ir gražinimo bei gautų knygų katalogo aktualizavimo procedūras. Ji taip pat užtikrina informacijos apie knygas ir apie skaitytojus šimtaprocentinį patikimumą ir kontroliuoja, kad i bibliotekos skaityklas nepatektų asmenys, neturintys galiojančių skaitytojo kortelių.“

Kaip matome, naujosios sistemos viziją beveik nekeičia informacinės sistemos funkcinių galimybių, tačiau ji esminiai keičia tų galimybių turinį. Sistemos galimybių reikalavimai yra formuluojami šitaip:

,,1.1. Bibliotekos naujoji informacinė sistema turi turėti šias galimybes:

- 0. Galimybė tvarkyti visą darbui su skaitytojais ir darbui su knygomis reikalingą informaciją*
 - 1. Galimybė tvarkyti knygų registrą kompiuteriniu būdu
 - 1.1. Galimybė kompiuteriniu būdu registruoti registre naujai įsigytas knygas*
 - 1.2. Galimybė kompiuteriniu būdu išregistruoti iš registro nurašomas knygas**
 - 2. Galimybė tvarkyti knygų katalogą ir dirbtį su juo kompiuteriniu būdu
 - 2.1 Galimybė kompiuteriniu būdu papildyti katalogą naujai įsigytų knygų bibliografiniais aprašais*
 - 2.1.1. Galimybė skenuoti knygų bibliografinius aprašus ir rašyti juos į katalogą*
 - 2.2. Galimybė kompiuteriniu būdu pašalinti iš katalogo nurašytų knygų bibliografinius aprašus*
 - 2.3. Galimybė bet kuriuo paros metu visiems norintiems per internetą pasinaudoti katalogu
 - 2.3.1. Galimybė peržiūrinėti visą katalogą*
 - 2.3.2. Galimybė ieškoti kataloge reikiamo leidinio pagal autoriaus pavardę, pagal knygos pavadinimą, pagal dalykinės srities, kuriai priskirtina knyga, pavadinimą***

- 2.3.3. Galimybė tarpusavyje derinti 2.3.2
nurodytus paieškos būdus
3. Galimybė tvarkyti skaitytojų registrą kompiuteriniu būdu
 - 3.1. Galimybė registruoti naujus skaitytojus
 - 3.1.1. Galimybė kompiuteriniu būdu rašyti į registrą įrašus apie naujus skaitytojus
 - 3.1.2. Galimybė ruošti lustines registre registruotų skaitytojų kortelės ir įregistruoti jas registre
 - 3.2. Galimybė išregistruoti skaitytojus
 - 3.2.1. Galimybė kompiuteriniu būdu paskelbti negaliojančiu skaitytojų registre esantį įrašą apie skaitytoją
 - 3.2.2. Galimybė kompiuteriniu būdu paskelbti negaliojančia skaitytojo kortelę
 4. Galimybė kompiuteriniu būdu tvarkyti bibliotekos transakcijų registrą ir palaikyti pačias transakcijas
 - 4.1. Galimybė užsisakyti norimą knygą per internetą
 - 4.2. Galimybė transakcijos metu skenuoti ir kompiuteriniu būdu apdoroti knygų brūkšninius kodus ir skaitytojų kortelėse įrašytus duomenis
 - 4.3. Galimybė kompiuteriniu būdu registruoti knygų išdavimus skaitytojams transakcijų registre ir išduodamas knygas paskolintų knygų registre
 - 4.4. Galimybė kompiuteriniu būdu registruoti knygų gražinimus transakcijų registre ir išregistruoti gražinamas knygas iš paskolintų knygų registro
 5. Galimybė kompiuteriniu būdu tvarkyti eilėse laukiančių skaitytojų registrą
 - 5.1. Galimybė paklausią knygą norinčiam gauti skaitytojui bet kuriuo paros metu per internetą užsirašyti į tos knygos laukiančių skaitytojų eilę
 - 5.2. Galimybė, skaitytojui gražinant jo skaitytą paklausią knygą, informuoti apie tai pirmąjį eilėje tos knygos laukiantį skaitytoją, automatiškai pasiunčiant pranešimą į jo kompiuterinio pašto dėžutę
 - 5.3. Galimybė, išduodant paklausią knygą eilėje jos laukusiam skaitytojui, automatiškai išbraukti tą skaitytoją iš eilės
 - 5.4. Galimybė skaitytojams per internetą stebėti savo judėjimą eilėse prie paklausiu knygų, kurių jie laukia
 6. Galimybė kompiuteriniu būdu tvarkyti bibliotekos informacinę vitriną
 - 6.1. Galimybė, įtraukiant knygos bibliografinį aprašą į knygų katalogą, automatiškai paskelbti jos bibliografinį aprašą informacinės vitrinos skyriuje „Naujai gautos knygos“

- 6.2. Galimybė informacinės vitrinos skyriuje „Idomiausių knygų recenzijos“ kompiuteriniu būdu skelbti trumpas idomiausių knygų recenzijas
 - 6.3. Galimybė automatiškai šalinti iš vitrinos pasenusią informaciją
 - 6.4. Galimybė informacinės vitrinos skyriuje „Kiti skelbimai“ kompiuteriniu būdu skelbti įvairius bibliotekos skelbimus
 - 6.5. Galimybė bet kuriuo paros metu visiems norintiems skaityti per internetą informacinėje vitrinoje paskelbtą medžiagą
 - 7. Galimybė kontroliuoti bibliotekos skaityklų skaitytojų srautą
 - 7.1. Galimybė kompiuteriniu būdu tikrinti jų skaityklas einančių skaitytojų kortèles ir praleisti tik teisę naudotis skaityklomis turinčius skaitytojus
 - 7.2. Galimybė automatiškai kontroliuoti (taip, kaip tai daroma savitarnos parduotuvėse), ar skaitytojai neišsineša iš skaityklų atviruose fonduose saugomų leidinių, ir neįleisti iš skaityklos tą bandančių daryti skaitytojų.
-

Dar turėtų būti suformuluotos darbo su informacija taisykles. Jos lieka tos pačios, kaip ir senojoje sistemoje. Pridedami tiktais papildomi reikalavimai, kad vienu metu bibliotekos informacinės sistemos paslaugomis, išskaitant ir per internetą teikiamas paslaugas, gali naudotis iki 500 asmenų, ir kad paieška kataloge neturėtų trukti ilgiau kaip pusė minutės.

Pastaba. Pavyzdys apima tik dalį bibliotekos verslą palaikančios IS. Sistema dar turėtų teikti buhalterines, žmogiškųjų išteklių tvarkymo ir kitas paslaugas. Tačiau, pavyzdys yra gana tikroviškas, nes, kaip minėjome, paprastai vienu projektu pertvarkoma tik tam tikra organizacijos informacinės sistemos dalis.



Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kodėl?” skamba šitaip:

”Esamą informacinę sistemą todėl reikia pertvarkyti, kad būtų įgyvendinta naujos informacinės sistemos vizija ir taptų prieinamomis jos galimybių medžiu numatytos galimybės.”

2.6.3 Kaip?

Atsakant į klausimą kaip įgyvendinti naujosios informacinės sistemos viziją, reikia suformuluoti informacinės sistemos architektūros reikalavimus (t. y. nustatyti iš kokių funkcinių komponentų turi būti sudaryta informacinė sistema) ir galimybių medžiu numatytas sistemos galimybes susieti su tas galimybes įgyvendinančiais informacinės sistemos komponentais. Kitaip tariant, informacinės sistemos viziją ir tą viziją detalizuojančiu galimybių medžiu numatytas galimybes reikia *lokaliizuoti* tos sistemos komponentuose ir *nuleisti žemyn*, į sistemos komponentų lygmenį, t. y. reikia aprašyti kiekvieno komponento viziją ir sudaryti to komponento galimybių

medij. Reikia turėti omenyje, kad kai kurios sistemos galimybės, jos vadinamos *aspektais*, gali išsibarstyti po daugelį informacinės sistemos komponentų.

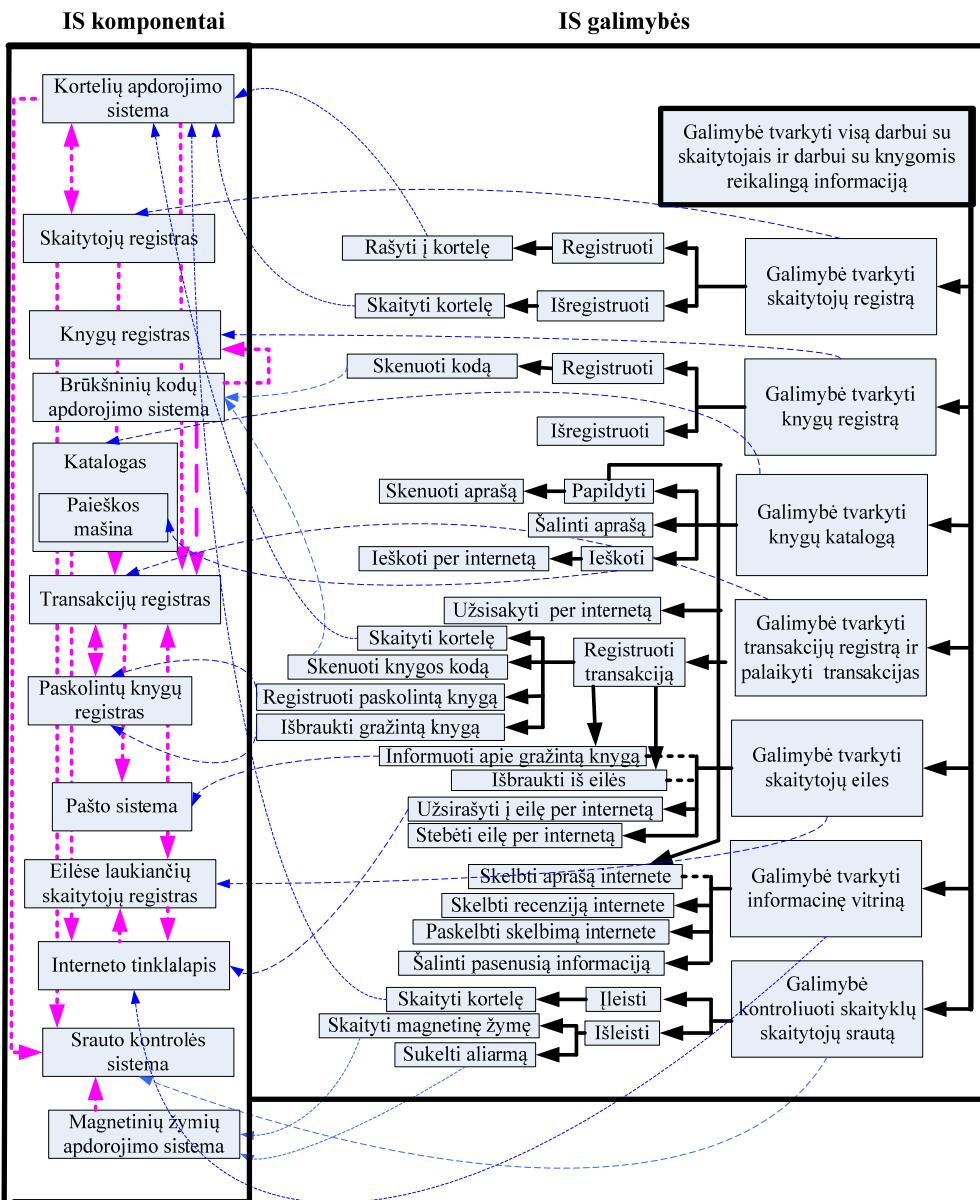
Informacinės sistemos požiūriu, informacinės sistemos suskaidymas į komponentus yra projektavimo uždavinys, tačiau, programų sistemos požiūriui, tai – reikalavimų inžinerijos uždavinys, nes, nežinant kokias galimybes turi įgyvendinti programų sistema, negalima suformuluoti detalių tos sistemos reikalavimų ir pradėti ją projektuoti. Šis faktas aiškiai parodo, kad reikalavimų formulavimo ir projektavimo atskyrimas yra gana salyginis ir kad iš tiesų reikalavimų inžinerijos procesai tampriai susipina su kitais sistemos inžinerijos procesais.

Reikia turėti omenyje, kad, atsakant į klausimą „Kaip“, sistemos galimybės turi būti lokalizuotos ne tik kompiuterizuotuose, bet ir nekompiuterizuotuose informacinės sistemos komponentuose, aišku, jeigu tokį komponentų sistemoje apskritai lieka. Beje, terminą *komponentas* čia mes vartojaime nebūtinai komponentinės paradigmos prasme, nors ir yra patartina į informacinės sistemos komponentus bandyti žvelgti būtent iš komponentinės paradigmos požiūrio taško, t. y. traktuoti juos kaip realizacijos požiūriu vienas nuo kito nepriklausomus funkcinius vienetus, teikiančius nustatytas paslaugas per tiksliai apibrėžtą interfeisą. Tačiau, kai kurioms informacinėms sistemoms komponentinės paradigmos reikalavimai gali būti sunkiai įgyvendinami ir komponentai jose gali būti mažiau formalizuoti.

Taip pat reikia turėti omenyje, kad čia yra kalbama tik apie *funkcinius* informacinės sistemos komponentus arba, kitaip tariant, informacijos saugykļų reikalavimai kol kas néra nagrinėjami. Registrų, kadastrų bei kiti informacijos saugyklas aptarnaujantys komponentai traktuojami kaip funkciniai komponentai, nes jie teikia tam tikras funkcinės, pavyzdžiui, registravimo, paslaugas. Taigi, juos reikia įtraukti į funkcių komponentų sąrašą ir jiems suformuluoti tokius pačius reikalavimus, kaip ir kitiems informacinės sistemos funkciniams komponentams. Beje, funkcių komponentų įvairovė taip pat yra gana didelė. Ji apima *duomenų apdorojimo sistemas, transakcijų apdorojimo sistemas, užklausų sistemas, informavimo sistemas, duomenų analizės sistemas, ekspertines sistemas, sprendimus priimti padedančias sistemas, planavimo sistemas, stebėjimo sistemas, valdymo sistemas, grupinio darbo organizavimo sistemas, darbų srautų tvarkymo sistemas, išteklių tvarkymo sistemas, turinio tvarkymo sistemas, žinių tvarkymo sistemas, dokumentų tvarkymo sistemas, biuro sistemas, ryšio sistemas, gamybos procesų valdymo sistemas, įvairias paieškos mašinas* ir daugelį kitų dalykų. Visi paminėti sistemų tipai yra trumpai aptarti knygos gale pridėtame terminų žodynėlyje.

Nors IS komponentams apibūdinti mes vartojaime terminus *sistema* ir *mašina*, tačiau nebūtinai turime omenyje kokias nors programų sistemos. Kalbama apie tam tikros paskirties darbo su informacija sistemas, išskaitant ir nekompiuterizuotas. Be to, IS gali turėti kelis to paties tipo, bet skirtinges paskirties komponentus ir kiekvienas iš jų gali būti kompiuterizuotas skirtingu laipsniu. Kadangi informacinės sistemos teikiamų paslaugų kompiuterizavimo laipsnis jau buvo apibrėžtas formuluojant vartotojo reikalavimus, tai dabar, vadovaujantis tais reikalavimais, yra nustatomas kiekvieno informacinės sistemos funkcionio komponento kompiuterizavimo laipsnis.

Pavyzdys



30 pav. Informacinė sistemos galimybių susiejimas su jos komponentais

30 paveikslėlyje parodyta, kaip praeitame pavyzdyje nagrinėtos bibliotekos informacinės sistemos galimybės yra lokalizuojamos jos komponentuose. Bibliotekos naujosios informacinės sistemos architektūros reikalavimai skamba šitaip:

“2.1. Bibliotekos naujoji informacinė sistema turi būti sudaryta iš šių funkcinių komponentų:

- kompiuterinio skaitytojų registro;
- kompiuterinio knygų registro;
- kompiuterinio paskolintų knygų registro;
- kompiuterinio transakcijų (knygų skolinimo ir gražinimo operacijų) registro;

- kompiuterinio paklausų knygų laukiančių skaitytojų registro;
 - kompiuterinio knygų katalogo;
 - paieškos kataloge mašinos;
 - skaitytojo kortelių apdorojimo sistemos;
 - knygų brūkšninių kodų apdorojimo sistemos;
 - magnetinių žymių, kuriomis pažymeti skaityklų atviruose fonuose saugomi leidiniai, apdorojimo sistemos;
 - skaitytojų srauto kontrolės sistemos, tikrinančios jeinančių skaitytojų korteles, valdančios jėjimą į skaityklą kontroliuojantį įrenginį ir pakeliančios aliarmą, skaitytojui bandant išsinešti magnetine žyme pažymėtą leidinį;
 - kompiuterinio pašto sistemos, per kurią automatiškai yra siunčiami pranešimai skaitytojams;
 - Interneto tinklalapis⁹, per kurį yra prieinama prie knygų katalogo, užsirašoma į knygų laukiančių skaitytojų eiles, galima stebėti skaitytojų judėjimą tose eilėse ir skaitoma informacinėje vitrinoje paskelbta medžiaga.
-

Visi šiaiems reikalavimais numatyti registrai yra transakcijų apdorojimo sistemos, apdorojančios atitinkamas duomenų bazių transakcijas (t. y. registruose registruojamų objektų registravimo ir išregistravimo operacijas). Kitaip tariant, tai programos, aptarnaujančios atitinkamas duomenų bazės lenteles. Katalogas taip pat yra transakcijų apdorojimo sistema, o paieškos mašina – užklausų sistema. Skaitytojo kortelių, knygų brūkšninių kodų ir knygų magnetinių žymių apdorojimo sistemos yra specializuotos duomenų apdorojimo sistemos, dirbančios su specialios paskirties duomenų skaitymo ir rašymo įrenginiais. Kompiuterinio pašto sistema yra ryšio sistema. Skaitytojų srauto kontrolės sistema yra valdymo sistema. Bibliotekos tinklapis yra turinio tvarkymo sistema. Visiems šiems informacinės sistemos komponentams turi būti suformuluoti patikimumo, saugos ir tikslumo reikalavimai. Jie formuluojami, remiantis vartotojo reikalavimais, kuriuose pasakyta, kokie turi būti informacinės sistemos teikiamų paslaugų patikumas, sauga ir tikslumas. Tai tik preliminarūs ir gana abstraktūs reikalavimai. Išsamūs reikalavimai bus formuluojami kiekvienam komponentui atskirai. Tai bus daroma, perėjus prie programų sistemų reikalavimų formulavimo.



Lokalizuojant informacinės sistemos galimybes jos komponentuose yra suformuluojami patys abstrakčiausių tų komponentų funkciniai reikalavimai. Tačiau, atsakymui į klausimą „Kaip?“ vien funkcinių reikalavimų nepakanka, nes reikia pasakyti ne tik tai, kokias paslaugas privalo teikti komponentai, bet ir tai, kiek patikimos bei saugios tos paslaugos turi būti ir koks turi būti jų pateikiamų rezultatų tikslumas. Komponentų patikimumo, saugos ir tikslumo reikalavimai yra gaunami iš

⁹ Kol kas tinklalapis yra nagrinėjamas kaip funkcinis informacinės sistemos komponentas, o ne kaip informacijos saugykla. Todėl yra kalbama tik apie tai, kaip juo galima pasinaudoti.

vartotojų reikalavimų lygmenyje suformuluotų informacinės sistemos teikiamų paslaugų ribojimų, lokalizuojant tuos ribojimus informacinės sistemos komponentuose ir nuleidžiant juos žemyn į komponentų lygmenį.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kaip?“ skamba šitaip:

„Informacinės sistemos galimybių medžiu numatytos galimybės yra įgyvendinamos sukuriant nurodytus informacinės sistemos komponentus, tenkinančius duotaisiais patikimumo, saugos ir tikslumo reikalavimais numatytais ribojimus.“

2.6.4 Ką?

Atsakant į klausimą „Ką?“, yra išvardinamos visos informacijos saugyklos (duomenų bazės¹⁰, dokumentų bazės, skirtingų paskirčių repozitorijai ir kt.), iškaitant ir nekompiuterizuotas popierinių dokumentų saugyklas, kurios privalo būti informacinėje sistemoje, ir vartotojo lygmenyje suformuluoti koncepciniai verslo objektus modeliuojančių informaciinių objektų reikalavimai yra lokalizuojami tose saugyklose bei nuleidžiami į jų lygmenį. Tai reiškia, kad kiekvienai informacijos saugyklai reikia nurodysti, kokie informaciniai objektai joje bus saugomi, kokiui tikslumu tie objektai turi būti vaizduojami, kokie yra jų darnos reikalavimai ir kokie yra jų apsaugos reikalavimai. Kiekvienai informacijos saugyklai reikia nurodysti, ar ji turi būti centralizuota, ar išskirstyta. Išskirstytoms informacijos saugykloms reikia aprašyti jose saugomos informacijos replikavimo reikalavimus.

Vartotojo reikalavimuose suformuluoti informaciinių objektų pateikties reikalavimai yra konkretizuojami, nurodant kokiose laikmenose (popieriuje, interneto puslapyje, kompiuterio ekrane ir t.t.) turi būti pateikiami tie informaciniai objektai.

Pavyzdys

Naujajai bibliotekos informacinei sistemai reikalavimai gali būti formuluojami šitaip:

“3.1. Bibliotekos naujoje informacinėje sistemoje turėtų būti numatytos šios informacijos saugyklos:

- *skaitytojų duomenų bazę;*
- *knygų duomenų bazę;*
- *paskolintų knygų bazę;*
- *transakcijų duomenų bazę;*
- *paklausių knygų laukiančių skaitytojų bazę;*
- *bibliografinių aprašų bazę;*
- *informacinės vitrinos turinio saugykla.*

3.1.1. Skaitytojų duomenų bazę skirta įrašams apie skaitytojus saugoti. Ji turi būti realizuota kaip kompiuterizuota duomenų bazę. Kiekvienam skaitytojui turi būti suteiktas unikalus registravimo numeris, naudojamas kaip kodas

¹⁰ Terminas *duomenų bazę* čia suprantamas kaip saugykla skirta saugoti įrašo tipo informaciniams objektams. Vėliau, projektuojant programinę įrangą, tokia duomenų bazę dažniausiai realizuojama kokios nors reliacinės duomenų bazės lentele.

įrašams apie skaitytojus identifikuoti. Rašyti naujus įrašus iš skaitytojų bazę bei keisti ten saugomų įrašų turinį, išskaitant ir įrašo paskelbimą negaliojančiu, leidžiamą tik bibliotekos registratoriaus darbo vietoje dirbantiems darbuotojams. Skaityti įrašus iš bazės gali visi bibliotekos darbuotojai. Įrašai pateikiami darbuotojo darbo vėtos kompiuterio ekrane. I bazę galima rašyti tik įrašus, kuriuose visi laukai yra užpildyti. Už įraše pateiktos informacijos korektiškumą atsako tą įrašą sukūrės darbuotojas.

3.1.2. Knygų duomenų bazę skirta įrašams apie bibliotekos turimas knygas saugoti. Įrašas aprašo konkretų knygos egzempliorių, tai yra, jei biblioteka turi kelis tos pačios knygos egzempliorius, kiekvienas iš jų yra aprašomas savu įrašu. Bazę turi būti realizuota kaip kompiuterizuota duomenų bazė. Kiekvienai knygai turi būti suteiktas unikalus inventorinis numeris, naudojamas kaip kodas įrašams apie knygas identifikuoti. Rašyti naujus įrašus i knygų bazę bei keisti ten saugomų įrašų turinį, išskaitant ir įrašo paskelbimą negaliojančiu, leidžiamą tik buhalterijos darbuotojams. Skaityti įrašus iš bazės gali visi bibliotekos darbuotojai. Įrašai pateikiami darbuotojo darbo vėtos kompiuterio ekrane. I bazę galima rašyti tik įrašus, kuriuose visi laukai yra užpildyti. Už įraše pateiktos informacijos korektiškumą atsako tą įrašą sukūrės buhalterijos darbuotojas.

3.1.3. Paskolintų knygų duomenų bazę skirta įrašams apie skaitytojams paskolintas knygas saugoti. Įrašas aprašo konkretų paskolintos knygos egzempliorių, jis identifikuojamas paskolintos knygos inventoriniu numeriu. Bazę turi būti realizuota kaip kompiuterizuota duomenų bazė. Įrašomi į paskolintų knygų bazę ir šalinami iš jos automatiškai, registruojant knygų išdavimo skaitytojams ir jų gražinimo operacijas. Skaityti įrašus iš bazės gali visi bibliotekos darbuotojai. Įrašai pateikiami darbuotojo darbo vėtos kompiuterio ekrane. Rašant į bazę reikia patikrinti ar įrašas apie išduodamą knygą yra knygų bazėje, ar įrašas apie skaitytoją, kuriam, išduodama knyga, yra skaitytojų bazėje ir ar paskolintų knygų bazėje dar nėra įrašo, kad ta knyga jau paskolinta.

“

Analogiškai formuluojami ir kitų informacijos saugyklu reikalavimai.



Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Ką?” skamba šitaip:

“Funkciniai informacinės sistemos komponentai apdoroja nurodytose informacijos saugyklose saugomus informaciinius objektus, tenkinančius nurodytus vaizdavimo tikslumo ir darnos reikalavimus. Prieiga prie informaciinių objektų ribojama nurodytais apsaugos reikalavimais.”

2.6.5 Kas?

Atsakant į klausimą „Kas?“, iš vartotojo lygmenyje suformuluotų reikalavimų, nusakančių kokios IS paslaugų gavėjų grupės kokiomis jos teikiamomis paslaugomis naudosis, reikia išvesti IS komponentų interfeisų reikalavimus. Kitaip tariant, reikia pasakyti, kokius interfeisus privalo turėti kiekvienas IS komponentas ir kokie tie interfeisai turėtų būti, kad jais patogu būtų naudotis IS paslaugas per tuos interfeisus gaunantiems paslaugų gavėjams. Priminsime, kad paslaugų gavėjais gali būti ne tik

organizacijos darbuotojai, bet ir jos klientai, verslo partneriai, informacinės sistemos stebimi ar valdomi įrenginiai, kokios nors programų sistemos. Trumpumo dėlei, toliau šiame skyrelyje visus paslaugų gavėjus vadinsime vartotojais.

Visų pirma aptarkime IS komponento vartotojo interfeiso sampratą. Programuotojai, programų sistemų inžinieriai, o kartais netgi informacinių sistemų inžinieriai, linkę kalbėti tik apie programų sistemų vartotojo interfeisus. Tačiau visos dirbtinės sistemos, iškaitant nekompiuterizuotas sistemas, turi savo vartotojus, kurie vienaip ar kitaip tomis sistemomis naudojasi. Tai reiškia, kad visos dirbtinės sistemos, iškaitant nekompiuterizuotus IS komponentus, turi vartotojų interfeisus, kuriuos reikia specifikuoti ir projektuoti. Bendruoju atveju, *vartotojo interfeisas yra apibrėžiamas kaip vartotojo ir sistemos sąveikos būdas* (sąveikos protokolas), kuris yra specifikuojamas taisyklėmis, nustatanciomis, ką gi turi padaryti vartotojas, norėdamas pasinaudoti ta sistema. Interfeisai gali būti labai įvairūs. Kai kuriomis sistemomis galima pasinaudoti išreikštiniu būdu joms paliepiant ką nors padaryti. Tokių sistemų vartotojo interfeisai vadinami komandiniais. Kitų sistemų reikia paprašyti, kad jos suteiktų tam tikrą paslaugą. Dar kitoms sistemoms pakanka pasakyti savo siekiams tikslus, o jos pačios „susipranta“, kokią paslaugą vartotojui reikia. Bendruoju atveju, sistemomis galima naudotis ir daugeliu kitų būdų.

Pavyzdys

Nepatenkintas gautu pažymiu studentas gali teikti apeliaciją. Tam jis naudojasi tam tikra universiteto kaip verslo sistemos teikiama paslauga – apeliacijos nagrinėjimo paslauga. Paslauga studentas gali pasinaudoti tik per universiteto informacинę sistemą, tiksliau, jis turi pasinaudoti vienu iš IS vartotojo interfeisių. Per jį pateikiamas prašymas išnagrinėti apeliaciją ir gaunamai apeliacijos nagrinėjimo rezultatai. Interfeisą aprašančios taisyklės nustato prašymo išnagrinėti apeliaciją formą, kam ir iki kada tas prašymas turi būti pateiktas, atsakymo apie apeliacijos nagrinėjimo rezultatus formą ir kaip tas atsakymas bus pateiktas studentui. Kompiuterizavus šį universiteto IS komponentą, prašymo išnagrinėti apeliaciją formą studentas galėtų gauti ekrane, ją užpildęs, tarkime, išrašytį į specialiai tam skirtą duomenų bazę ir po kurio laiko gauti kompiuterio ekrane apeliacijos nagrinėjimo rezultatus, skaitydamas juos, tarkime, iš tos pačios duomenų bazės. Šiuo atveju interfeisą sudaro ne tik kompiuterio ekrane pateikiamos formos, bet ir jų pildymo bei rašymo į bazę (skaitymo iš bazės) taisyklės, nusakančios ką gi turi padaryti studentas (pasirinkti komandas iš meniu, pildyti dialogines formas, klyktelėdamas pele, spaudyti dialoginių formų mygtukus ir kt.), kad gautų norimus rezultatus. Tieki nekompiuterizuoto, tiek ir kompiuterizuoto komponento interfeisai gali būti nepatogūs vartotojui. Pavyzdžiui, nekompiuterizuoto komponento atveju gali būti reikalaujama rinkti kokius nors parašus, registruoti prašymą katedroje, po to studentų skyriuje ar dar kur nors, atlikti kitus prašymo pateikimą apsunkinančius veiksmus. Kompiuterizuoto interfeiso atveju gali būti reikalaujama patvirtinti kiekvieno užpildyto lauko turinį jį renkant pakartotinai arba atlikti kokius nors kitus bereikalingus veiksmus. Kadangi nekompiuterizuotų IS komponentų interfeisai yra mažiau efektyvūs, tai blogai suprojektuotas interfeisas gali pareikalauti daug didesnės gaišties, negu blogai suprojektuotas kompiuterizuoto komponento interfeisas. Todėl nekompiuterizuotų IS komponentų interfeisai turėtų būti projektuojami ypač kruopščiai. Deja, taip esti labai retai.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kas?” skamba šitaip:

„Kiekviena informacinės sistemos paslauga gali naudotis nurodytos tos paslaugos gavėjų grupės, išskaitant ir antrinius vartotojus. Visi paslaugos gavėjai gali naudotis paslauga tik nustatyta įgaliojimų ribose. Bet kuri sistemos paslauga turi būti projektuojama, atsižvelgiant į nurodytus jos naudojimo ypatumus, teikiant pirmenybę arba naudojimosi paslauga paprastumui, arba paslaugos gavėjo darbo našumui, arba kitiems nurodytiems kriterijams. Projektuojant organizacijos darbuotojams skirtas paslaugas, reikia remtis prielaida, kad tie darbuotojai tenkina nurodytus kvalifikacinius reikalavimus. Projektuojant klientams skirtas paslaugas, privalu vadovautis prielaida, kad paslaugos turi būti tokios, jog jomis gebėtų pasinaudoti bet kuris vidurinį išsilavinimą turintis asmuo, išskaitant ir neįgaliuosius.“

2.6.6 Kur?

Atsakant į klausimą „Kur?“, yra formuliuojami IS darbo vietų reikalavimai. Šie reikalavimai nusako, kokia techninė ir programinė įranga turi būti įrengta kiekvienoje darbo vietoje ir kokius ergonominius reikalavimus ta darbo vieta turi tenkinti.

Techninės įrangos reikalavimai privalo aprašyti ne tik kompiuterinę, bet ir kitą techninę įrangą (dokumentų kopijavimo aparatai, telefonai ir kt.), kuri turi būti įrengta toje darbo vietoje. Kompiuterinės įrangos reikalavimai turi būti pakankamai bendro pobūdžio, nes detalūs informacinių sistemų palaikančių programų sistemų reikalavimai kol kas dar nėra suformuluoti ir todėl kol kas dar nėra tiksliai žinoma, kokie turi būti darbo vietoje įrengiamos kompiuterinės įrangos parametrai ir kokia sisteminė programinė įranga turi veikti ten pastatytuose kompiuteriuose. Nereikia pamiršti, kad darbo vietas sąvoka apima ir informacinių sistemų informacijos saugyklas. Todėl, formuliuojant darbų vietų techninės įrangos reikalavimus, reikia suformuluoti ir duomenų bazių bei kitų informacinių serverių techninės įrangos reikalavimus. Taip pat turi būti suformuluoti ir darbo vietas siejančių kompiuterių tinklų ar kitokių komunikavimo tinklų reikalavimai.

Programinės įrangos reikalavimai patikslina vartotojo lygmenyje suformuluotus reikalavimus, t. y. aprašo kokios informacinių sistemų palaikančios programų sistemas turi būti prieinamos vartotojui dirbančiam toje darbo vietoje, kad jis galėtų pasinaudoti vartotojo reikalavimais numatytomis informacinių sistemų paslaugomis.

Sakoma, kad darbo vieta yra *ergonomiška*, jei jos charakteristikos yra suderintos su ten dirbančių asmenų psichofiziologinėmis charakteristikomis. Darbo vietas charakteristikos apima jos apšvietimo ypatumus, antiradiacinę saugą, baldų metmenis ir daugelį kitų dalykų. Ergomininiai reikalavimai taip pat aprašo, kaip kompiuteriai ir jų periferiniai įrenginiai turi būti išdėstyti patalpoje, kad jais būtų patogu pasinaudoti ten dirbantiems asmenims. Specifikuojant klientams aptarnauti skirtų terminalų ergonominius reikalavimus, nereikia pamiršti, kad tais terminalais naudosis ir neįgalieliai asmenys. Be abejo, ergominės darbo vietas charakteristikos, kaip ir darbo vietoje įrengiamos nekompiuterinės techninės įrangos charakteristikos, programų sistemų inžinierių nedomina. Su jais dirba kiti specialistai.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kur?” skamba šitaip:

„Informacinių sistemų palaikančiomis programų sistemomis galima pasinaudoti tik nurodytose darbo vietose, kuriose yra įrengta nurodyta techninė įranga. Darbo vietas turi būti susietos tarpusavyje komunikavimo tinklais, tenkinančiais suformuluotus reikalavimus. Informacinės sistemos teikiamų paslaugų vartotojų darbo vietas turi būti kompiuterinio ryšio priemonėmis susietos su nurodytais duomenų bazėmis ir kitais informacinių serveriais, kurie turi būti aprūpinti nurodyta technine įranga.“

2.6.7 Kada?

Atsakant į klausimą „Kada?“ yra tikslinami vartotojo reikalavimų lygmenyje suformuluoti našumo reikalavimai ir yra nustatoma per kiek laiko turi būti įvykdymas kiekviena iš informacinėje sistemoje numatyta informacijos apdorojimo užduočių.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kada?” skamba šitaip:

„Informacinių sistemų palaikančios programų sistemas bei kitis informacijos apdorojimo užduotys atliekantys informacines sistemos komponentai, įskaitant ir nekompiuterizuotus komponentus, privalo užtikrinti, kad vartotojams reikalinga informacija bus pateikta neviršiant nurodytų laiko ribojimų.“

2.6.8 Baigiamosios pastabos

Informacinės sistemos reikalavimai tos sistemos programinės įrangos reikalavimus aprašo tik labai bendru lygmeniu:

- iš kokių programų sistemų bus sudaryta ta įranga;
- kokia yra kiekvienos programų sistemos vizija ir kokias galimybes kiekviena iš tų sistemų turi įgyvendinti;
- kokios duomenų bazės ir kokie kiti informacinių serveriai yra reikalingi kiekvienos iš informacinių sistemų palaikančių programų sistemų veikimui užtikrinti;
- kokios vartotojų grupės naudos kiekviena iš informacinių sistemų palaikančių programų sistemų, kokie yra tų grupių įgaliojimai (teisės), kokiais kriterijais (patogumo, našumo) vadovaujantis turi būti projektuojami joms skirti programų sistemų interfeisai;
- kokios yra pačios bendriausios programų sistemoms veikti reikalingos programinės įrangos ir informacinių sistemų kompiuterių tinklų charakteristikos;
- kokie laiko ribojimai riboja programų sistemų vykdomą informacijos apdorojimo užduočių vykdymo laiką.

Šie reikalavimai turi būti detalizuoti, patikslinti ir papildyti formuluojant kiekvienos programų sistemos bei organizacijos kompiuterių tinklo reikalavimus.

2.7 Programinės įrangos reikalavimai

2.7.1 Programinės įrangos reikalavimų formulavimo ypatumai

Programinės įrangos reikalavimai yra išvedami iš informacinių sistemų reikalavimų, lokalizuojant juos tą įrangą sudarančiose programų sistemose ir nuleidžiant iš tų sistemų lygmenį.

2.7.2 Kodėl?

Atsakant į klausimą „Kodėl?“, yra formuluojami ekonominiai, politiniai ir teisiniai programų sistemos ribojimai bei tos sistemos kokybės vertinimo kriterijai.

Programų sistemos ekonominiai ribojimai riboja ilgalaikius programų sistemos eksploatavimo kaštus. Šiuos kaštus sudaro programų sistemos diegimo kaštai, jos aptarnavimo kaštai, jos priežiūros kaštai ir lėšos prarandamos dėl to, kad nėra galima pakartotinai panaudoti sistemos kituose projektuose.

Programų sistemos politiniai ribojimai nusako kaip organizacijos politika riboja programų sistemos panaudojimą. Kitaip tariant, jie nusako nuo kokių grėsmių turi būti apsaugota programų sistema.

Programų sistemos teisiniai ribojimai nusako kokiais galiojančiais teisiniais aktais yra ribojama programų sistema ir kokius išorinius standartus ji turi tenkinti.

Programų sistemos kokybės vertinimo kriterijai nusako sistemos kokybės charakteristikų (t. y. nefunkcinių reikalavimų) prioritetus. Pavyzdžiui, jie pasako, kas yra svarbiau – sistemos patikimumas ar jos našumas.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kodėl?” skamba šitaip:

„Programų sistema todėl turi tenkinti jai formuluojamus funkcinius ir nefunkcinius reikalavimus, kad tie reikalavimai išplaukia iš nurodytų tos sistemos ekonominį, politinį, teisinių ribojimų bei iš nurodytų tos sistemos kokybės vertinimo kriterijų.“

2.7.3 Kaip?

Atsakant į klausimą „Kaip?“ (kaip patenkinti programų sistemos ribojimus), yra formuluojami programų sistemos funkciniai (išskyrus apdorojamų duomenų apsaugos reikalavimus), saugos, patikumo, diegimo, aptarnavimo ir priežiūros reikalavimai. Taigi, yra formuluojamos šios programų sistemos funkcinių reikalavimų grupės:

- programų sistemos tinkamumo funkciniu požiūriu reikalavimai;
- programų sistemos sąveikos su kitomis programų sistemomis reikalavimai;
- programų sistemos atitikimo galiojantiems teisės aktams ir standartams reikalavimai;
- programų sistemos trasuojamumo (patikrinamumo) reikalavimai.

Programų sistemos tinkamumo reikalavimai aprašo kokias funkcijas turi turėti kuriama programų sistema. Kiekviena funkcija yra aprašoma, nurodant jos pradinius duomenis, gaunamus rezultatus ir tų rezultatų gavimo būdą.

Programų sistemos sąveikos su kitomis programų sistemomis reikalavimai aprašo:

- programų sistemas su kuriomis turi būti realizuota kuriamos programų sistemos sąveika;
- formatus duomenų, kuriais programų sistema turi keistis su kitomis programų sistemomis;

- grafinius ir valdančiuosius ženklus, kuriais programų sistema turi keistis su kitomis programų sistemomis;
- programų sistemos sąveikos su kitomis programų sistemomis interfeisus;
- programų sistemos sąveikos su kitomis programų sistemomis standartus;
- pastangas, kurių turi pakakti programų sistemos sąveikai su kitomis, iš anksto nežinomomis programų sistemomis realizuoti.

Programų sistemos atitikimo galiojantiems teisės aktams bei standartams reikalavimai aprašo:

- kokių duomenų vaizdavimo formatai ir kaip turi būti standartizuoti kuriamojoje programų sistemoje;
- kokių media (garsų, vaizdų ir kt.) saugojimo formatai ir kaip turi būti standartizuoti kuriamojoje programų sistemoje;
- kokie grafiniai ir valdantieji ženklai ir kaip turi būti standartizuoti kuriamojoje programų sistemoje;
- kokie interfeisai ir kaip turi būti standartizuoti kuriamojoje programų sistemoje;
- kurios kuriamos programų sistemos funkcijos ir kokius standartus privalo tenkinti.

Programų sistemos trasuojamumo (angl. *traceability*) reikalavimai riboja pastangas (žmogaus darbo valandomis), kurių reikia tam, kad pasirinktuose programos taškuose būtų galima patikrinti tarpinių skaičiavimų rezultatų teisingumą. Kiekybiškai programų sistemos trasuojamumo reikalavimai gali būti formuluojami, nurodant kokio maksimalaus laiko (procentais nuo viso užduočiai atlikti reikalingo laiko) gali prireikti žmogui trasavimui atlikti. Galima taip pat nurodyti kiek laiko (procentais nuo viso užduočiai atlikti reikalingo laiko) leidžiama sugaišti sistemai trasavimui atlikti.



31 pav. Programų sistemos patikimumo komponentai

Programų sistemos patikimumo (angl. *reliability*) reikalavmai aprašo kokį laiko tarpą programų sistema, jei ji yra tinkamai eksplotuojama, privalo išlikti korekтиška ir produktyvi. Programų sistemos patikimumo reikalavimai yra skirtomi į (31 pav.):

- išbaigtumo reikalavimus;
- atsparumo triktims reikalavimus;

- atkuriamumo reikalavimus;
- prieinamumo reikalavimus;
- pažeidžiamumo reikalavimus.

Programų sistemos išbaigtumo (angl. *maturity*) reikalavimai riboja programų sistemoje likusių klaidų sukeliamų trikčių dažnį. Kiekybiškai programų sistemos išbaigtumo reikalavimai gali būti formuluojami, nurodant:

- koks vidutinis laikas tarp dviejų trikčių yra dar priimtinis užsakovui;
- koks trumpiausias laiko tarpas tarp dviejų trikčių, įvykusiu per nurodytą laiko periodą, yra dar priimtinis užsakovui;
- koks yra priimtinis užbaigtoje programų sistemoje likusių klaidų (pagal vertinimus) ir viso programų sistemos kodo apimties (pvz., matuojant operatoriais) santykis;
- koks turi būti sistemą testuojant atliktą testą ir viso programų sistemos kodo apimties santykis.

Programų sistemos atsparumo triktims (angl. *fault tolerance*) reikalavimai apibūdina, kokiui laipsniu ta sistema privalo išlikti korekтиška ir produktyvi po trikčių ar įsilaužimų. Kiekybiškai programų sistemos atsparumo triktims reikalavimai gali būti formuluojami, nurodant:

- kokį maksimalų laiką per nurodytą laiko periodą leidžiama prarasti dėl trikčių ar įsilaužimų;
- laiką, per kurį turi būti galima nutraukti sistemos darbą, pastebėjus triktį ar įsilaužimą;
- įsilaužimo per nurodytą laiko periodą tikimybę;
- didžiausią leistiną per nurodytą laiko periodą įvykusiu sistemos korekтиškumo ar produktyvumo praradimų skaičiaus ir per tą periodą įvykusiu trikčių skaičiaus santykį;
- didžiausią leistiną per nurodytą laiko periodą sistemos išduotų pranešimų apie klaidas ir faktiškai įvykusiu klaidų skaičiaus santykį.

Programų sistemos atkuriamumo (angl. *recoverability*) reikalavimai apibūdina pastangas, reikalingas atkurti po trikties prarastą sistemos funkcionalumą ir/arba duomenis. Kiekybiškai programų sistemos atkuriamumo reikalavimai gali būti formuluojami, nurodant:

- vidutinį laiką, reikalingą prarastam programų sistemos funkcionalumui atkurti;
- didžiausią leistiną per nurodytą laiko periodą automatiškai atkurto prarasto sistemos funkcionalumo skaičiaus ir visų per tą laiką buvusių funkcionalumo praradimo atvejų santykį;
- maksimalų laiką, per kurį turi būti atkuriamas prarastas programų sistemos funkcionalumas;
- vidutinį laiką, per kurį turi būti rasta ir pašalinta triktį sukėlus programų sistemoje likusi kaida.

Programų sistemos prieinamumo (angl. *availability*) reikalavimai apibūdina tikimybę, kad programų sistema bus galima pasinaudoti visuomet, kai to prireiks. Kiekybiškai programų sistemos prieinamumo reikalavimai gali būti formuluojami, nurodant:

- mažiausią laiką, kurį per nurodytą laiko periodą sistema privalo išlikti funkcionali ir produktyvi;
- mažiausią leistiną laiko trukmės, kuomet sistema galima naudotis, ir laiko trukmės, kuomet sistema reikėtų pasinaudoti, santykį;
- mažiausią procentą laiko, kuriuo per nurodytą periodą turi būti galima pasinaudoti sistema.

Programų sistemos pažeidžiamumo (angl. *degradability*) reikalavimai apibūdina pastangos, reikalingos atkurti esmines sistemos funkcijas, jai praradus savo funkcionalumą. Kiekybiškai programų sistemos pažeidžiamumo reikalavimai gali būti formuluojami, nurodant per kokią ilgiausią laiko tarpą turi būti atkurtos esminės sistemos funkcijos, jai praradus savo funkcionalumą.

Programų sistemos aptarnavimo reikalavimais yra siekiama sumažinti tos sistemos aptarnavimo kaštus. Programų sistemos aptarnavimas yra suprantamas kaip pastangos, reikalingos tos sistemos korektiškumui ir produktyvumui palaikyti. Taigi, programų sistemos aptarnavimas apima nuolatinį sistemos stebėjimą, išaiškėjusių klaidų šalinimą ir prarasto sistemos funkcionalumo atkūrimą. Todėl programų sistemos patikimumo reikalavimai apima ir jos aptarnavimo reikalavimus, nes juose gali būti nurodyta per kiek laiko turi būti galima rasti ir pašalinti išryškėjusių programų sistemos klaidą, per kiek laiko turi būti galima atkurti prarastą programų sistemos funkcionalumą arba bent jau esmines jos funkcijas ir pan. Transformuojant tokius reikalavimus į programų sistemos projektinius ir realizacinius reikalavimus yra nusakomi ribojimai kuriamos programų sistemos struktūrai, jos komponentų kodo struktūrai bei reikalavimai, kokie klaidų paieškos bei prarasto funkcionalumo atkūrimo mechanizmai privalo būti realizuoti kuriamojoje programų sistemoje. Be abejo, visa tai supaprastina ir atpigina tos sistemos aptarnavimą.



32 pav. Programų sistemos perkeliavimo reikalavimai

Programų sistemos diegimo reikalavimais siekiama sumažinti tos sistemos diegimo kaštus. Programų sistemos diegimas susideda iš techninio tos sistemos parengimo darbui, duomenų ir kitų tai sistemai veikti reikalingų išteklių akumuliavimo ir sistemos vartotojų mokymo. Atsakant į klausimą „Kaip?“ yra aptariami tik sistemos techninio parengiamumo reikalavimai. Jie vadinami programų sistemos perkeliavimo (angl. *portability*) reikalavimais, nes tiek diegiant sistemą pirmą kartą, tiek ir keliant ją į naują kompiuterinę platformą, praktiskai tenka atlikti tuos pačius techninius darbus. Kokios priemonės turi būti numatytos sistemoje jai veikti reikalingiems duomenims sukaupti, yra aptariama atsakant į klausimą „Ką?“, o ką reikia padaryti jos būsimųjų vartotojų mokymui supaprastinti – atsakant į klausimą „Kas?“

Programų sistemos perkeliavimo reikalavimai apima (32 pav.):

- programų sistemos adaptuojamumo reikalavimus;
- programų sistemos instaluojamumo reikalavimus;
- programų sistemos atitikimo keliamumo standartams reikalavimus;
- programų sistemos pakeičiamumo kita sistema (demontavimo) reikalavimus.

Programų sistemos adaptuojamumo (angl. *adaptability*) reikalavimai nusako tikimybę, kad tą sistemą pavyks perkelti į kitą platformą, nedarant jokių jos perdarymų. Programų sistemos adaptuojamumo reikalavimai gali būti formuluojami, nurodant:

- maksimalias pastangos (žmogaus darbo valandomis), kurių turi pakakti pritaikyti sistemą naujai platformai;
- kompiuterines platformas, į kurias turi būti galima kelti sistemą;
- operacines sistemas, į kurias turi būti galima kelti sistemą;
- DBVS, į kurias turi būti galima perkelti sistemos duomenis;
- mažiausią leistiną santykį sistemos naudojimo dokumentacijos, kurios nereikia keisti perkėlus sistemą į kitą platformą, apimties (puslapiais) su visa sistemos naudojimo dokumentacijos apimtimi.

Programų sistemos instaluojamumo (angl. *installability*) reikalavimai riboja pastangas, reikalingas sistemos instalavimo darbams atliliki. Kiekybiškai programų sistemos instaluojamumo reikalavimai gali būti formuluojami, nurodant:

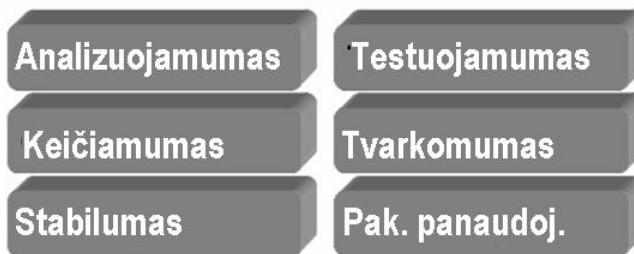
- žmogaus darbo valandas, kurių turi pakakti sistemai instaluoti;
- maksimalų procentą sistemos parametru (nuo bendro skaičiaus), kurių reikšmes leidžiama keisti instaluojant programų sistemą;
- maksimalų sistemos kodo procentą (nuo viso kodo apimties), leidžiamą keisti instaluojant programų sistemą;
- maksimalų procentą failų (nuo bendro sistemos failų skaičiaus), leidžiamų keisti instaluojant programų sistemą.

Programų sistemos atitikimo keliamumo standartams (angl. *conformance*) reikalavimai apibūdina reikalaujamą sistemos atitikimo perkeliavimo standartams ir susitarimams laipsnį. Atitikimo keliamumo standartams reikalavimai paprastai yra formuluojami, nurodant perkeliavimo standartus bei susitarimus, į kuriuos turi būti atsižvelgta kuriant sistemą.

Programų sistemos pakeičiamumo (angl. *replaceability*) reikalavimai riboja pastangas (žmogaus darbo valandomis), reikalingas anksčiau organizacijoje naudotą sistemą (kompiuterinę arba rankinę) pakeisti kuriamaja programų sistema. Kiekybiškai programų sistemos pakeičiamumo reikalavimai gali būti formuluojami, nurodant:

- maksimalų keičiamų sistemos funkcijų procentą (nuo viso funkcijų skaičiaus);
- maksimalų keičiamos kodo dalies procentą (nuo visos kodo apimties).

Keičiant kokį nors nekompiuterizuotą informacinių sistemos komponentą programų sistema, be abejo, tenka rašyti visą tos sistemos kodą (t. y. 100%). Todėl tokiais atvejais formuluooti antrajį reikalavimą jokios prasmės nėra.



33 pav. Programų sistemos prižiūrimumo reikalavimai

Programų sistemos prižiūrimumo (angl. *Maintainability*) reikalavimai riboja pastangas (žmogaus darbo valandomis), reikalingas sistemai perdaryti, norint ją pritaikyti pasikeitusioms aplinkos sąlygoms. Kitaip tariant, sistemos priežiūra yra suprantama kaip nuolatinis sistemos modernizavimas. Programų sistemos prižiūrimumo reikalavimai apima (33 pav.):

- programų sistemos analizuojamumo reikalavimus;
- programų sistemos keičiamumo reikalavimus;
- programų sistemos stabilumo reikalavimus;
- programų sistemos testuojamumo reikalavimus;
- programų sistemos tvarkomumo reikalavimus;
- programų sistemos pakartotino panaudojamumo reikalavimus.

Programų sistemos analizuojamumo (angl. *analysability*) reikalavimai riboja pastangas (žmogaus darbo valandomis), kurių reikia tam, kad būtų galima nustatyti sistemos defektų ar trikčių priežastis ir išsiaiškinti, ką sistemoje reikia keisti norint tas priežastis pašalinti. Kiekybiškai programų sistemos analizuojamumo reikalavimai gali būti formuluojami, nurodant:

- mažiausią leistiną santykį skaičiaus per nurodytą periodą įvykusiu tokių trikčių, kurių priežastis turi pavykti nustatyti teisingai, su visu per tą periodą įvykusiu trikčių skaičiumi (angl. *Fault position recognition ratio*);
- vidutinį laiką, kurio turi pakakti sistemos triktį iššaukusioms klaidoms rasti (angl. *mean failure analysis time*).

Programų sistemos keičiamumo (angl. *changeability*) reikalavimai riboja pastangas (žmogaus darbo valandomis), kurių reikia sistemoje esančiai klaidai ištaisyti arba padaryti pakeitimus, pritaikant programų sistemą jos pasikeitusiui aplinkai. Kiekybiškai programų sistemos keičiamumo reikalavimai gali būti formuluojami, nurodant:

- reikalaujamą vidutinio pakeitimui padaryti reikalingo laiko ir pakeitimo apimties santykį (angl. *modification effort per unit volume*);
- reikalaujamą vidutinį laiką, reikalingą kokiam nors sistemos defektui pašalinti (angl. *correction effort per defect*);

- reikalaujamą vidutinį laiką, reikalingą programoje esančiai klaidai ištaisyti (angl. *mean fault correction time*);
- kiek vidutiniškai gali praeiti laiko nuo trikties įvykimo momento iki momento, kada vartotojas vėl gali naudotis sistema (angl. *mean failure treatment time*);
- kiek programuotojo darbo valandų (vidutiniškai) turi pakakti programoje esančiai klaidai pašalinti (angl. *mean fault correction work time*);
- kiek laiko vidutiniškai gali būti gaištama (įskaitant išsiaiškinimo laiką) vienai programos pataisymų kodo eilutei parašyti (angl. *mean revise work time per changed line of source code*);
- mažiausią leistiną santykį priežiūros inžinieriams skirtos dokumentacijos (puslapiais) su sistemos pradinio kodo apimtimi (kodo eilutėmis) (angl. *fulfilment degree of maintenance document*).

Programų sistemos stabilumo (angl. *stability*) reikalavimai nusako tikimybę, kad sistemoje padaryti pakeitimai duos kokį nors netikėtą šalutinį efektą. Kiekybiškai programų sistemos stabilumo reikalavimai gali būti formuluojami, nurodant mažiausią leistiną vidutinio pakeitimą darant padarytų klaidų ir pakeitimo apimties santykį.

Programų sistemos testuojamumo (angl. *testability*) reikalavimai riboja pastangas (žmogaus darbo valandomis), kurių reikia sistemos korektiškumui (pradžioje ar padarius pakeitimus) patikrinti. Kiekybiškai programų sistemos testuojamumo reikalavimai gali būti formuluojami, nurodant:

- maksimalias leistinas pastangas (žmogaus darbo valandomis), kurių turi pakakti nurodytos kodo apimties korektiškumui (naudojant nurodytą testų skaičių) patikrinti (angl. *test effort per unit volume*);
- kiek vidutiniškai laiko turi pakakti vartotojui įsitikinti, kad klaida buvo pataisыта teisingai (angl. *mean user's work time to verify the fault correction*);
- kiek vidutiniškai laiko turi pakakti programuotojui patikrinti, ar teisingai buvo pataisыта klaida (angl. *mean work time to test the fault correction*).

Programų sistemos tvarkomumo (angl. *manageability*) reikalavimai riboja pastangas (žmogaus darbo valandomis), kurių reikia padaryti sistemą veikiančia arba kurių reikia jos veikimui atkurti. Kiekybiškai programų sistemos tvarkomumo reikalavimai gali būti formuluojami, nurodant didžiausią leistiną santykį žmogaus darbo valandų (įskaitant priežiūros darbus), gaištamų per nurodytą laiko periodą padaryti sistemą veikiančią ar jos veikimui atkurti, su valandomis, kuriomis per tą patį periodą vartotojas gali naudotis sistema (angl. *control effort ratio*).

Programų sistemos pakartotino panaudojamumo (angl. *reusability*) reikalavimai nusako kokiu laipsniu sistema ar jos dalimis turi būti galima pakartotinai pasinaudoti kituose projektuose. Kiekybiškai programų sistemos pakartotino panaudojamumo reikalavimai gali būti formuluojami, nurodant mažiausią leistiną santykį pakartotinai panaudojamų sistemas dalių su visų jos dalių skaičiumi (angl. *ratio reusable parts*).

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kaip?” skamba šitaip:

„Programų sistema įgyvendina reikalaujamas galimybes ir tenkina nustatytais ekonominiaus, politinius bei teisinius ribojimus įgyvendindama nurodytą funkcionaluą, užtikrindama nurodytą patikimumą ir nereikalaudama daugiau pastangų savo diegimui, aptarnavimui bei priežiūrai negu leidžiamą atitinkamais reikalavimais.“

2.7.4 Ką?

Atsakant į klausimą „Ką?“, yra formuluojami programų sistemos apdorojamų duomenų reikalavimai, tų duomenų apsaugos reikalavimai ir nurodoma, kokios priemonės turi būti numatytos sistemoje jai veikti reikalingiemis duomenims sukaupti jos diegimo metu.

Programų sistemų inžinierius dirba su skaitmenizuotais informaciniiais objektais (angl. *digital objects*) arba, kitaip tariant, su informaciniais objektais pavaizduotais kompiuteryje. Kadangi tai dar nėra projektinio lygmens reikalavimai, tai tikslus informacinių objektų vaizdavimo kompiuteryje būdas dar nėra aiškus, t. y. dar nėra žinoma, ar jie bus vaizduojami reliacinės duomenų bazės lentelėmis, ar objektais objektinės paradigmos prasme, ar dar kaip nors kitaip. Priminsime, kad skaimeniuojami ne tik informaciniai verslo objektų aprašai, bet ir vaizdai, garsai bei kita informacija. Tikslus jų vaizdavimo būdas taip pat dar nėra žinomas, pavyzdžiu, nėra žinoma, ar vaizdai bus saugomi .jpg, .gif ar dar kokiu nors formatu. Tai projektavimo meto sprendimai. Tačiau, formuluojant programų sistemos reikalavimus, jau galima kalbėti apie vaizduojamų objektų turinį ir bendruosius informacinių objektų vaizdavimo reikalavimus, visų pirma, apie informacinių objektų vaizdavimo tikslumą. Tai ir yra aprašoma duomenų reikalavimais. Priminsime, kad kalba eina tik apie sistemos informacijos saugyklose saugomą informaciją. Sistemai pateikiama duomenų ir jos generuojamų duomenų reikalavimai yra formuluojami atsakant į klausimą „Kaip?“.

Programų sistemos apsaugos (angl. *security*) reikalavimai nusako, kokių laipsniu sistema turi būti apsaugota nuo galimybų tyčia ar netycia ja pasinaudoti neteisėtai. Teisėtu programų sistemos naudojimu vadintamas tos sistemos naudojimas pagal jos paskirtį, atliekamas tam oficialius įgaliojimus turinčio asmens ar proceso (įskaitant kitas programų sistemas). Programų sistemos apsaugos reikalavimai aprašo:

- grėsmes, nuo kurių turi būti apsaugota sistema;
- vartotojų ir procesų registravimo procedūras;
- vartotojų ir procesų skirstymo į klasses ir teisių priskyrimo toms klasėms taisykles;
- vartotojų ir procesų autorizavimo taisykles;
- duomenų ir sistemos funkcijų klasifikavimo pagal slaptumo kategorijas taisykles ir tų klasių apsaugos lygmenis.

Kiek lėšų prireiks tam, kad būtų sukaupti programų sistemai veikti reikalingi duomenis ir kiti informacijos ištakliai, priklauso nuo to, ar kuriamoji sistema keičia anksčiau naudotas rankines informacijos apdorojimo procedūras, ar anksčiau naudotą kitą programų sistemą, demontojamą diegiant naujai sukurtą sistemą. Antruoju atveju reikiami informaciniai ištakliai jau yra sukaupti ir juos tik reikia perkelti į

naujają sistemą. Tam reikia numatyti tam tikrą vienkartinio panaudojimo kuriamosios programų sistemos funkcionalumą. Pirmuoju atveju informacinius išteklius dar reikia sukaupti ir tai gali būti labai didelis darbas. Pavyzdžiu, kompiuterizujant kokią nors biblioteką gali tekti digitalizuoti visą jos milijoninių apimčių katalogą. Šiam darbui palengvinti taip pat reikia numatyti atitinkamą vienkartinio panaudojimo pagalbinį kuriamos programų sistemos funkcionalumą. Abiem atvejais turi būti numatytos priemonės sukauptiems informaciniams ištekliams peržiūrėti, analizuoti bei vertinti. Aišku, kai kuriais atvejais programų sistema gali naudotis vien tik išoriniai informacinių ištekliai. Kadangi tokiais atvejais diegiant sistemą jokių informacinių išteklių kaupti nereikia, tai tam nereikia numatyti ir jokio papildomo funkcionalumo. Kitaip tariant, tokiais atvejais informacinių išteklių kaupimo reikalavimai nėra formuluojami.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Ką?“ skamba šitaip:

„Programų sistema privalo nurodytose informacijos saugyklose saugoti nurodytus digitalizuotus informacijos objektus. Tie objektai turi būti apsaugoti nuo nurodytų tyčinių ir netycinių grėsmių jais neteisėtai pasinaudoti. Digitalizuotiemis informacijos objektams sistemos informacijos saugyklose sukaupti sistemoje turi būti numatytas nurodytas papildomas funkcionalumas.“

2.7.5 Kas?

Atsakant į klausimą „Kas?“ yra formuluojami programų sistemos vartotojo interfeisių ribojimai. Vartotojo interfeisių ribojimai riboja programų sistemos ir jos vartotojų sąveikos būdus (komandų sintakę, dialogines formas, funkcinius klavišus, pelēs naudojimą ir pan.). Programų sistemos vartotojo interfeisių reikalavimai nieko nekalba apie sistemos galimybes. Jie tik aprašo, kaip tomis galimybėmis galima pasinaudoti. Pažeidus interfeiso reikalavimus, sistema išlieka korektiška, bet jos sąveika su vartotojais tampa nekorektiška. Nors sistema ir vykdo visas funkciniais reikalavimais numatytas funkcijas, tačiau tomis funkcijomis nebegalima pasinaudoti pilnoje apimtyje. Taigi vartotojo interfeisių reikalavimai aprašo, kiek vartotojo interfeisių turi turėti kuriamoji programų sistema ir kaip kiekvienas iš tų interfeisių turi atrodyti. Kiekvienas vartojo interfeisas turi būti aprašomas tokiomis reikalavimų grupėmis:

- vartotojo užduočių formulavimo reikalavimai;
- interfeiso panaudojamumo reikalavimai;
- interfeiso ergonominių reikalavimai.

Programų sistemos vartotojo užduočių formulavimo kalbos reikalavimai aprašo, kokia turi būti užduočių formulavimo kalba (UFK) ir kaip ta kalba yra formuluojamos užduotys. Užduočių formulavimo kalba yra suprantama labai plačiai, kaip priemonių, įskaitant kliktelėjimus pele, meniu, dialogo langus ir kt., pasinaudodamas kuriomis vartotojas formuluoja užduotis kompiuteriui visuma. Visos tos priemonės yra traktuojamos kaip užduočių formulavimo kalbos konstrukcijos. Formuluojant vartotojo užduočių formulavimo kalbos reikalavimus, reikia aprašyti:

- kalbos konstrukcijų abstrakcijos lygmenį (kalbos semantinę galią);

- kaip ta kalba yra nurodoma kompiuteriui ką jis turi atlikti (kalbos procedūriškumo laipsnį);
- užduočiai formuluoti vartojamų dalykinės srities terminų žodyną (metaforą, kuria yra grindžiama kalba);
- kokius užduoties naudojamiems duomenims filtruoti skirtus filtrus galima ta kalba aprašyti (kalbos selektyvinę gebą);
- kokius užduoties aspektus galima specifikuoti ta kalba (kalbos raiškos galią), papildomai nenaudojant kitokių kalbinių priemonių (pvz., papildomai pridedamų Perl ar kitokių skriptų);
- kalbos sintaksės ypatumus (operatorių kalba, meniu kalba, formų kalba ar pan.);
- užduočių pateikties kompiuteriui protokolą.

Programų sistemos vartotojo interfeiso naudojimo paprastumo reikalavimai apima:

- interfeiso vidinės darnos reikalavimus (komandų formato standartas, klavišų naudojimo nepriklausomybė nuo konteksto, pranešimuose vartojamų terminų darna, manipuliavimo ekrano operacijų standartas ir pan.);
- interfeiso išorinės darnos reikalavimus (išorinius standartus, pavyzdžiui, Microsoft langų standartą, kuriuos turi tenkinti interfeisas);
- interfeiso akivaizdumo ir prasmingumo reikalavimus.
- interfeiso patogumo vartotojui (angl. *easy-to-use*) reikalavimai.



34 pav. Programų sistemos panaudojamumo reikalavimai

Programų sistemos panaudojamumo reikalavimai riboja pastangas (žmogaus darbo valandomis), kurių reikia vartotojams pasinaudoti sistema. Panaudojamumo reikalavimai apima (34 pav.):

- programų sistemos suprantamumo reikalavimus;
- programų sistemos išmokstamumo reikalavimus;
- programų sistemos operabilumo reikalavimus;
- programų sistemos būsenos vizualizavimo reikalavimus;
- programų sistemos individualizuojamumo reikalavimus;

- programų sistemos patrauklumo reikalavimus;
- programų sistemos aiškumo reikalavimus;
- programų sistemos informatyvumo reikalavimus;
- programų sistemos patogumo vartotojui reikalavimus.

Programų sistemos suprantamumo (angl. *understandability*) reikalavimai riboja pastangas (žmogaus darbo valandomis), kurių vartotojui reikia sistemos koncepcijai ir jos panaudojimo būdu perprasti. Kiekybiškai programų sistemos suprantamumo reikalavimus galima suformuluoti, nurodant:

- reikalaujamą vartotojų duodamo sistemos informacinio aprūpinimo (naudojimo instrukcijos, help failai, užduočių formulavimo kalba ir jos pateiktis vartotojo interfeise ir kt.) suprantamumo įvertį (angl. *rated understandability*);
- reikalaujamą produkto (dokumentai, pranešimai, piktogramos ir kt.) skaitomumo įvertį (angl. *readability score*), užrašomą panaudojant Dale-Chall formulę (ši formulė skaitomumą vertina pagal sakinių ilgį ir skaitytojui nežinomų žodžių skaičių);
- reikalaujamą programų sistemos koncepcinio aiškumo (angl. *concept clearness*) įvertį (kokis mažiausias procentas sistemos funkcijų turi būti aprašomas vartotojui išprastomis sąvokomis bei terminais);
- kokiam mažiausiam programų sistemos funkcijų procentui jų veikimas turi būti aiškinamas demonstraciniais pavyzdžiais;
- reikalaujamą programų sistemos panaudojimo aiškumo (angl. *usage clearness*) įvertį (kokiam mažiausiam programų sistemos funkcijų procentui jų panaudojimas sistemoje turi būti kaip nors (modeliais, demo, aprašymais ar kt.) paaiškintas);
- kokiam mažiausiam programų sistemos funkcijų procentui dokumentacijoje turi būti pateikti jų pradinių duomenų ir rezultatų aprašai;
- kokiam mažiausiam programų sistemos funkcijų parametrų procentui dokumentacijoje turi būti nurodyta, ar jie yra pastovūs, ar pakeičiami vykdant tas funkcijas.

Programų sistemos išmokstamumo (angl. *learnability*) reikalavimai riboja pastangas, matuojanas žmogaus darbo valandomis, kurių vartotojui reikia išmokti naudotis sistema (užduočių formulavimas ir pateiktis, įvesties duomenys, rezultatų interpretavimas ir pan.). Kiekybiškai programų sistemos išmokstamumo reikalavimus galima suformuluoti, nurodant:

- vidutinį laiką, kurio turi pakakti vartotojui išmokti naudotis programų sistemą;
- reikalaujamą vartotojo dokumentacijos naudingumo įvertį (kokis vidutinis pavyzdžių, paveikslėlių bei kitos iliustracinių medžiagos vienetų skaičius turi tekti vienai programų sistemos kodo eilutei);

- reikalaujamą vartotojo dokumentacijos prieinamumo įvertį (mokomosios medžiagos vienetų skaičių vienai programų sistemos funkcijai);
- mažiausią leistiną savarankiškam skaitymui skirtos įvadinės medžiagos vienetų skaičiaus santykį su sistemos funkcijų skaičiumi;
- mažiausią leistiną ruošiamos ir teoriškai reikalingos mokomosios medžiagos apimčių santykį;
- mažiausią leistiną ruošiamą ir teoriškai reikalingą help aprašų santykis;
- vidutinę laiką, kurio turi pakakti vartotojui išmokti pasinaudoti programų sistema vieno tipo užduotims vykdyti.

Programų sistemos operabilumo (angl. *operability*) reikalavimai riboja vartotojo pastangas, matuojamas žmogaus darbo valandomis, reikalingas užduotims formuliuoti ir vykdyti. Kiekybiškai programų sistemos operabilumo reikalavimus galima suformuluoti, nurodant:

- mažiausią leistiną programų sistemos operabilumo įvertį, nurodomą lyginant jį su kokia nors etalonine, gerai žinoma sistema;
- maksimalų leistiną sumarinį laiką, reikalingą pradžioje parengti sistemą naudojimui (instaliavimas, pritaikymas organizacijai ir pan.);
- maksimalų leistiną rankinių operacijų, atliekamų instaliujant sistemą, skaičių;
- minimalų leistiną sistemos instaliavimą palaikančių funkcijų skaičiaus santykį su visų sistemai instaliuoti reikalingų operacijų skaičiumi;
- taškų, kuriuose būtų galima pertraukti sistemos instaliavimo procesą, o po to jį vėl atnaujinti, skaičių;
- kiek ir kokų kompiuterinių išteklių, išskaitant techninę ir sisteminę programinę įrangą, turi pakakti sistemai instaliuoti;
- maksimalų leistiną žingsnių, reikalingų sistemos teisingumui patikrinti, skaičių;
- minimalų programų sistemos komandų, turinčių nutyrimus parametrus, skaičiaus santykį su visų sistemos komandų skaičiumi;
- minimalų komandų, turinčių standartizuotą (suvienodintą) formatą, procentą;
- minimalų standartizuotų terminų, vartojamų programų sistemos komandose, jos spausdinamuose pranešimuose ir kt., procentą;
- minimalų programų sistemos pranešimų, kuriuose turi būti aiškiai nurodyta pranešimo priežastis ir vartotojui pasakyta, ką jis turi padaryti, gavęs tą pranešimą, procentą;
- minimalų programų sistemos funkcijų, naudojant kurias, vartotojas turi turėti galimybę pasirinkti jo patirtį atitinkantį darbo režimą, procentą;
- minimalų ekraninių komandų, kurias privalu suvienodinti, procentą;

- minimalų standartizuotų ekraninių duomenų įvesties/išvesties laukų procentą;
- maksimalų leistiną veiksmų, atliekamų pradedant darbą su sistema, procentą;
- maksimalų leistiną užduočiai pakartoti reikalingų veiksmų skaičiaus santykį su pirmam tos užduoties vykdymui reikalingų veiksmų (to paties seanso metu) skaičiumi;
- vidutinį laiką tarp dviejų operatoriaus daromų klaidų;
- maksimalų leistiną laiką, reikalingą darbui su sistema užbaigti;
- minimalų procentą komandų/duomenų, kurių galima atsisakyti (angl. *cancel*) padarius klaidą;
- kokius dėmesio akcentus (spalvos, garsas, blykstelėjimai ir kt.) ir kokiai atvejais reikia naudoti;
- minimalų, maksimalų ir vidutinį sistemos reakcijos į vartotojo veiksmus laiką (angl. *response time for user*);
- minimalų, maksimalų ir vidutinį laiką, per kurį ekrane, to paprašius, turi keistis laukų turinys (angl. *display time*).

Programų sistemos būsenos vizualizavimo (angl. *explicitness*) reikalavimai aprašo sistemos gebėjimą informuoti apie tai, kas su ja vyksta (atliekamų veiksmų progreso indikacija ir pan.). Kiekybiškai programų sistemos būsenos vizualizavimo reikalavimus galima suformuluoti, nurodant:

- vidutinių laiko, kurį vartotojų leistina palikti nežinioje, trukmę (angl. *insecure time*);
- minimalų procentą veiksmų, į kuriuos programų sistema privalo reaguoti tuo pat, pranešdama apie savo būseną (angl. *status report ratio*);
- kiek daugiausiai laiko vartotojai gali būti paliekami nežinioje (angl. *insecure time*);
- reikalaujamų realizuoti progreso indikacijos atvejų sąrašas (angl. *status report list*).

Programų sistemos individualizuojamumo (angl. *customisability*) reikalavimai aprašo galimybes prisitaikyti sistemą savo individualiems poreikiams. Kiekybiškai programų sistemos individualizuojamumo reikalavimus galima suformuluoti, nurodant:

- minimalų leistiną konfigūruojamo programų sistemos funkcionalumo procentą (angl. *configurability ratio*);
- vidutinį laiką, reikalingą programų sistemių perkonfigūruoti (angl. *configurability effort*).

Tačiau dažniausiai yra nurodoma ne kokią funkcionalumo dalį reikia realizuoti kaip konfigūruojamą, o konkrečiai koki funkcionalumą reikia realizuoti kaip konfigūruojamą.

Programų sistemos patrauklumo (angl. *attractivity*) reikalavimai aprašo, kokiui laipsniu sistemos teikiamos paslaugos, jos elgsena bei jos naudojami informacijos vizualizavimo bei pateikties būdai turi tenkinti išreikštiniu būdu nesuformuluotus vartotojo pageidavimus, preferencijas bei nuostatas. Vienintelis būdas kiekybiškai suformuluoti programų sistemos patrauklumo reikalavimus yra nurodyti, koks sistemos vartotojų procentas jos patrauklumą turėtų vertinti teigiamai.

Programų sistemos aiškumo (angl. *clarity*) reikalavimai aprašo, kokiui laipsniu vartotojui turi būti akivaizdu, ką ta sistema moka daryti. Kiekybiškai programų sistemos aiškumo reikalavimus galima suformuluoti, nurodant:

- minimalų procentą programos sistemos galimybę, kurios turėtų savaime paaiškėti vartotojui neskaitant dokumentacijos, help failų ar kitos medžiagos ir vien tik padirbėjus su ta sistema nurodytą laiką;
- procentą programų sistemos galimybę pritaikytą vidutiniam (ne pradedančiajam) vartotojui.

Programų sistemos informatyvumo (angl. *helpfulness*) reikalavimai aprašo kokiu laipsniu vartotojams turi būti prieinama informacinė medžiaga apie tą sistemą. Kiekybiškai programų sistemos aiškumo reikalavimus galima suformuluoti, nurodant:

- apytikrią ekrano dalį (procentais), skiriamą vartotojo informacinei medžiagai, išskaitant ekrane spausdinamus pranešimus, talpinti (angl. *ratio of expounding text*);
- minimalų leistiną vartotojui prieinamos (išskaitant pranešimus) informacinės medžiagos apimties (žodžiais)santykį su visos programų sistemos kodo apimtimi (kodo eilutėmis) (angl. *normalised ratio of expounding text*).

Programų sistemos patogumo vartotojui (angl. *user-friendliness*) reikalavimai aprašo reikalaujamą vartotojų pasitenkinimo sistema laipsnį. Kiekybiškai programų sistemos patogumo vartotojui reikalavimus suformuluoti neįmanoma, tačiau galima pateikti kokybinę vertinimo skalę ir nurodyti, koks turėtų būti sistemos vertinimas pagal tą skalę. Šitaip suformuluotų reikalavimų įgyvendinimo laipsnį vertina arba nepriklausomi ekspertai, arba skaičiuojama koks procentas vartotojų praėjus tam tikram laikui po sistemos diegimo mano, kad sistema palengvino jų darbą (angl. *rated user-friendliness*).

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kas?“ skamba šitaip:

„Programų sistema privalo turėti nurodytus vartotojo interfeisus, leisti per tuos interfeisus formuluoti užduotis nurodytomis užduočių formulavimo kalbomis, pritaikytomis ten dirbančių vartotojų operacinius poreikius, išsilavinimą ir psichologinius ypatumus, ir užtikrinti nurodytą savo panaudojamumo laipsnį.“

2.7.6 Kur?

Atsakant į klausimą „Kur?“, yra formuluojami reikalavimai, nusakantys kokiose kompiuterinėse platformose (apimant techninę ir sisteminę programinę įrangą) privalo kuriamoji programų sistema veikti ir kokių minimalių techninių resursų turi pakakti jos veikimui užtikrinti.

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą „Kur?” skamba šitaip:

„*Programų sistema privalo veikti nurodytose kompiuterinėse platformose ir jos veikimui turi pakakti nurodytų techninės įrangos išteklių*“.

2.7.7 Kada?

Atsakant į klausimą „Kada?“, yra formuluojami programų sistemos našumo reikalavimai. Programų sistemos našumas (angl. *efficiency*) yra suprantamas kaip programų sistemos veikimo greičio ir prie nurodytų sąlygų jos naudojamų resursų santykis.

Našumas pagal laiką

Našumas pagal resursus

35 pav. Programų sistemos našumo reikalavimai

Programų sistemos našumo reikalavimai yra skirtomi į (35 pav.):

- našumo pagal laiką reikalavimus;
- našumo pagal resursus reikalavimus.

Programų sistemos našumo pagal laiką (angl. *time behaviour*) reikalavimai aprašo tos sistemos reakcijos laiką, skaičiavimų trukmę ir jos pralaidumą (gebėjimą per tam tikrą laiką apdoroti tam tikrą įvykių skaičių). Kiekybiškai programų sistemos našumo pagal laiką reikalavimus galima suformuluoti, nurodant:

- maksimalų laiką, per kurį sistema turi atlikti paketinio pobūdžio skaičiavimus (angl. *batch turnaround time*);
- kokias maksimalias duomenų apimtis privalo gebeti apdoroti programų sistema, atlikdama paketinio pobūdžio skaičiavimus (angl. *batch capacity*);
- kokią didžiausią nuosekliai apdorojamų duomenų struktūrų (pvz. sąskaitų faktūrų) kiekį privalo gebeti apdoroti programų sistema;
- vidutinį ir maksimalų kiekvieno tipo užduoties vykdymo laiką (angl. *processing time*) prie nurodytų duomenų apimčių;
- kiek kiekvieno tipo užduočių su nurodytomis duomenų apimtimis privalo gebeti apdoroti sistema per nurodytą laiko tarpą (angl. *processing capacity*);
- vidutinį kiekvieno tipo vidinės sistemos užduoties su nurodytomis duomenų apimtimis vykdymo laiką (angl. *average internal transaction time*);
- maksimalų kiekvieno tipo vidinės sistemos užduoties su nurodytomis duomenų apimtimis vykdymo laiką (angl. *maximum internal transaction time*);
- kiekvieno tipo užduoties bendrą skaičiavimų laiką, išskaitant užduoties formulavimą, pradinių duomenų pateiktį ir rezultatų pateiktį (angl. *turnaround time*);
- maksimalų laiką bet kurio tipo užduočiai, praeinantį nuo užduoties formulavimo pabaigos iki rezultatų pateikties pradžios (angl. *response time*);
- maksimalų laiką, praeinantį nuo programos vykdymo pradžios iki jos vykdymo pabaigos (angl. *CPU elapsed time*);

- kiek procesoriaus darbo laiko gali sunaudoti programa nuo jos vykdymo pradžios iki vykdymo pabaigos (angl. *CPU execution time*);
- kiek laiko gali sugaišti programa mainams tarp išorinės ir operatyviosios atminčių (angl. *I/O processing time*);
- laiką, kurį programų sistema gali sugaišti laukimui (pvz., spausdintuvo, diskų ar pan.) (angl. *waiting time*);
- laiką, kuris gali būti gaištamas duomenims perduoti kompiuterių tinklu (angl. *network processing time*);
- laiką, kurį užduoties vykdymo metu skaičiavimams gali sunaudoti kliento kompiuteris (angl. *terminal processing time*);
- kiek nurodyto tipo įvykių (pvz., kreipinių iš skirtinės klientų) privalo gebeti apdoroti programų sistema per nurodytą laiką (angl. *throughput*);
- kiek transakcijų privalo gebeti apdoroti programų sistema per nurodytą laiką (angl. *number of processed transactions*).

Programų sistemos našumo pagal resursus (angl. *resource behaviour*) reikalavimai nustato, kiek resursų ir kiek laiko gali sunaudoti programų sistema vykdydama savo funkcijas. Kiekybiškai programų sistemos našumo pagal resursus reikalavimus galima suformuluoti, nurodant:

- kiek ženklų per nurodytą laiką turi gebeti programų sistema perduoti per kiekvieną iš savo interfeisų, vykdydama nurodytas užduotis, (angl. *communication occupancy*);
- kiek operatyviosios atminties gali naudoti programų sistema nurodytoms duomenų apimtimis apdoroti (angl. *internal memory occupancy*);
- kiek išorinės atminties turi pakakti programų sistemai nurodytoms duomenų apimtimis apdoroti (angl. *external memory occupancy*);
- kiek procesoriaus darbo laiko gali sunaudoti programų sistema (angl. *processor occupancy*);
- kiek realiosios atminties gali užimti programų sistema (angl. *real memory occupancy*);
- kiek virtualiosios atminties gali užimti programų sistema (angl. *virtual memory occupancy*);
- kiek atminties gali naudoti programų sistema savo failams talpinti (angl. *file occupancy*);
- kokias duomenų apimtis gali programų sistema perdavinėti kompiuterių tinklu (angl. *network occupancy*);
- kiek laiko per atitinkamą laiko vienetą programų sistema gali naudotis procesoriumi (angl. *CPU utilisation*);
- kiek atminties per nurodytą laiko vienetą gali naudoti programų sistema (angl. *main memory utilisation*);
- kiek laiko per nurodytą laiko vienetą programų sistema gali naudotis įvesties/išvesties kanalais (angl. *I/O channel utilisation*);
- kiek laiko per nurodytą laiko vienetą programų sistema gali dirbti su failais (angl. *file utilisation*);
- kiek laiko per nurodytą laiko vienetą programų sistema gali naudotis kompiuterių tinklais (angl. *network utilisation*);
- kiek laiko per nurodytą laiko vienetą programų sistema gali naudotis įvesties/išvesties įrenginiais (angl. *I/O device utilisation*);

- kiek laiko per nurodytą laiko vienetą programų sistema gali naudotis kliento kompiuteriu (angl. *terminal utilisation*).

Trumpai reziumuojant, galima sakyti, kad atsakymas į klausimą “Kada?” skamba šitaip:

„Programų sistema privalo vykdyti savo užduotis per nurodytą laiką ir vykdyma tas užduotis naudoti ne daugiau kompiuterinių išteklių negu yra nurodyta“

2.7.8 Projektavimo reikalavimai

Programų sistemos projektavimo reikalavimai (tiksliau, projektavimo lygmens reikalavimai) apima:

- programų sistemos architektūros eskizinio lygmens reikalavimus;
- programų sistemos komponentų funkcinius, saugos, robastiškumo, patikimumo, diegimo, aptarnavimo ir priežiūros reikalavimus;
- loginio lygmens duomenų bazių ir kitų duomenų saugyklu reikalavimus;
- programų sistemos komponentų interfeisių reikalavimus;
- programų sistemos komponentų išdėstymo kompiuterių tinkle reikalavimus;
- programų sistemos komponentų našumo reikalavimus.

Programų sistemos projektavimo reikalavimai yra gaunami lokalizuojant programų sistemos reikalavimus tos sistemas komponentuose ir nuleidžiant lokalizuotus reikalavimus į programų sistemos komponentų lygmenį.

2.7.9 Realizavimo reikalavimai

Programų sistemos realizavimo reikalavimai (tiksliau, realizavimo lygmens reikalavimai) apima:

- programų sistemos detaliros architektūros reikalavimus;
- programų sistemos komponentų realizavimo ir testavimo reikalavimus;
- duomenų bazių ir kitų duomenų saugyklu fizinio lygmens reikalavimus;
- programų sistemos komponentų vidinių dalių (procedūrų, klasių ir kt.) interfeisių reikalavimus;
- skirtinguose tinklo mazguose veikiančių programų sistemos komponentų sąveikos reikalavimus;
- programų sistemos komponentų realizuojamų algoritmų efektyvumo reikalavimus.

Programų sistemos realizavimo reikalavimai yra gaunami tikslinant ir papildant tos sistemos projektinius reikalavimus.

2.7.10 Baigiamosios pastabos

Programų sistemos reikalavimai yra išvedami iš tos sistemos palaikomos informacinės sistemos reikalavimų. Šitaip gaunami bendrieji programų sistemos reikalavimai. Lokalizuojant bendruosius programų sistemos reikalavimus tos sistemos komponentuose ir nuleidžiant juos žemyn į komponentų lygmenį yra išvedami kiekvieno programų sistemos komponento reikalavimai arba, kitaip tariant, projektiniai programų sistemos reikalavimai. Tikslinant ir papildant programų sistemos reikalavimus yra gaunami tos sistemos realizacinių reikalavimų.

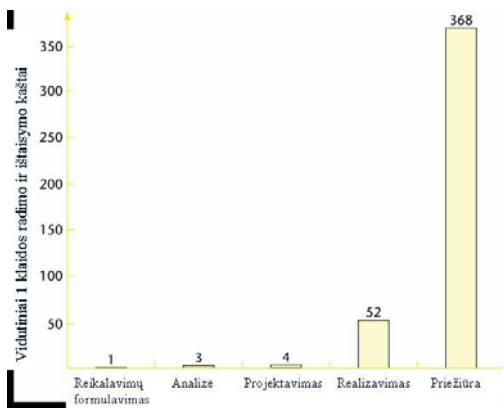
Programų sistemos reikalavimai turi būti, kiek tai įmanoma padaryti, suformuluoti kuo aiškiau, be dviprasmybių ir kiekybiškai. Būtina vengti negriežtų ir nepatikrinamų reikalavimų, kurių samprata yra subjektyvi, pavyzdžiui, „programinė įranga turi būti patikima“, „sistemos interfeisai turi būti patogūs vartotojams“ ir pan. Tai ypač svarbu nefunkciniams reikalavimams.

2.8 Reikalavimų svarba projekto sėkmėi

Reikalavimų reikšmė yra labai didelė. Apie tai mes jau šiek tiek kalbėjome pačioje šios knygos pradžioje. Apie reikalavimų formulavimo proceso sudėtingumą ir reikalavimų svarbą projekto sėkmėi vienas praktikas rašė šitaip:

„Reikalavimai yra didelis dalykas. Aš juos rašau visą gyvenimą. I reikalavimus aš sudeu viską apie tai, ką turėtų daryti programų sistema: kokias verslo taisykles ji turi tenkinti, ko turi daryti sistema, vartotojui paspaudus tą ar kitą klavišą, kokias užduotis vartotojas norės vykdyti ir taip toliau, ir taip toliau. Čia tikrai yra apie ką pagalvoti. Kartais ko nors pritrūksta. Kartais nieko netrūksta, bet vykdytojai klaudingai supranta mano mintį. Aš susidūriau su daugybe situacijų, kurios yra visiškai akivaizdžios verslo žmonėms, bet praranda savo akivaizdumą arba dėl jų perpasakojimo kitaip terminais, arba dėl to, kad nors ir nesunku įsivaizduoti tokias situacijas realizuojančias juodąsias dėžes, labai sunku realizuoti pačias tokias juodąsias dėžes. Ir visa tai vyksta nepaisant to, kad mes visi dirbame kartu jau daugelį metų, sėdimi tame pačiame pastate ir daugelis iš mūsų jau ištisus dešimtmečius dirba tokį darbą. Rasti ir pataisyti klaidą sistemoje testavimo metu, o juo labiau po to, kai ji pradėta eksploatuoti, kainuoja dešimt kartų daugiau, negu ją rasti ir pataisyti reikalavimuose ar projektinėje sistemos specifikacijoje. Blogai valdomame projekte kliaidoms surasti ir pataisyti jūs galite sugaišti daugiau laiko, negu jo buvo sugaišta sistemai sukurti.“ [9]

Tai tikra tiesa. Kiek kainuoja rasti ir pašalinti klaidą reikalavimų formulavimo metu ir kiek kainuoja tai padaryti baigus kurti sistemą ir pradėjus ją eksploatuoti, parodyta 36 paveikslėlyje.



36 pav. Klaidos radimo ir šalinimo kaštai

Visa tai nėra nauja. Apie reikalavimų svarbą projekto sėkmėi kalbama jau daugiau kaip dvidešimt metų. Apie tai prirašyta kalnai knygų ir straipsnių. Daugelis studentų kartu studijavo reikalavimų inžineriją ir laikė atitinkamus egzaminus. Padėtis, be abejo, gerėja, tačiau tai vyksta labai pamažu. Karl Wiegers [117] siūlo tuo įsitikinti patiemis, paėmus bet kurią programinę įrangą kuriančią bendrovę ir pabandžius atsakyti, kokiu mastu jos esamą būklę aprašo šie teiginiai:

- Nei produkto vizija, nei jo galimybės išreikštiniu būdu niekuomet nėra aprašomi.
- Užsakovai visuomet yra per daug užsiémę, kad galėtų skirti laiko padėti analitikui suformuluoti reikalavimus.
- Vartotojų vardu kalba tarpininkai, dažniausiai, užsakovo padalinių vadovai, kurie niekuomet iki galo nežino vartotojų tikrųjų poreikių.
- Reikalavimai saugomi programinį produktą kuriančių „ekspertų“ galvose ir niekuomet nėra oficialiai dokumentuojami.
- Užsakovai teigia, kad visi reikalavimai yra vienodai svarbūs ir todėl jie negali jų prioritetizuoti.
- Reikalavimų dviprasmybės ir nepakankamas išsamumas išaiškėja tik pradėjus rašyti programas.
- Analitikas su užsakovais svarsto vartotojo interfeiso ypatumus, o ne tai, ką vartotojai darys, naudodamiesi sukurta programinę įrangą.
- Užsakovai pasirašo reikalavimus jų netgi iki galo neperskaitę, o po to pradeda juos nuolat kaitalioti.
- Užsakovams kaitaliojant reikalavimus, darbų apimtis auga, bet niekas ir negalvoja peržiūrėti projekto terminus ar mažinti kuriamo produkto funkcionalumą.
- Aptarti reikalavimų pokyčiai yra pamirštami ir po kurio laiko nei vykdytojai, nei užsakovai, nebežino, kokia gi reikalavimų versija yra paskutinė.
- Užsakovai reikalauja tam tikro funkcionalumo, vykdytojai jį įgyvendina, bet nei vienam vartotojui niekuomet jo neprisireikia.
- Visi reikalavimai yra įgyvendinti, bet užsakovas nepatenkintas tuo, ką jis gavo.

Kai kurie reikalavimai aprašo tokias sistemos savybes, kurias pagimdo jos komponentų sąveikos būdas ir kurios nėra įgyvendintos nei viename konkrečiame sistemos komponente. Tokios savybės vadinamos *architektūrinėmis* (angl. *emergent properties*), nes jos priklauso nuo pasirinktos sistemos architektūros.

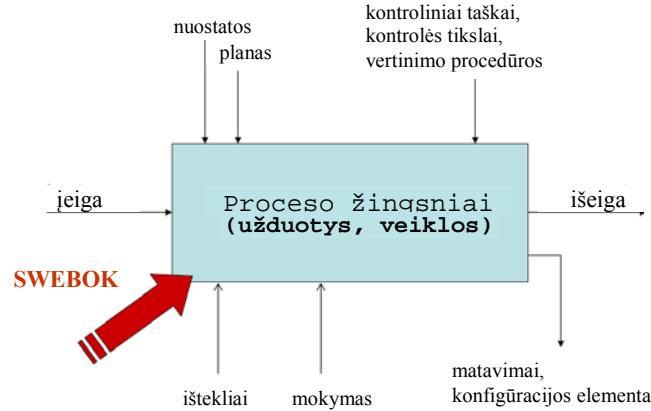
3 Reikalavimų inžinerijos procesas

3.1 Reikalavimų inžinerijos proceso samprata

Įsivaizduokite, kad jūs dirbate mažoje ar vidutinio dydžio bendrovėje, kuriančioje programinę įrangą. Kiekviena grupė ar netgi kiekvienas darbuotojas dirba taip, kaip jis išmano, jokios taisyklės nereglamentuoja nei to, kaip reikia aiškintis ir formuliuoti reikalavimus, nei to, kaip projektuoti programas, nei to, kaip jas įgyvendinti. Visi darbuotojai yra „praktikai“, puikūs specialistai ir, savaime aišku, su panieka žiūri į visokius paistalus, kurių pripaistytą programų sistemų inžinerijos vadoveliuose ir kurių pilna universitetuose dėstomuose kursuose. Tai niekam nereikalingos teorijos, o jie, kaip jau minėjome, yra „praktikai“, gerai išmano savo darbą, dirba po penkis ar daugiau metų, sėkmingai vykdo projektus ir puikiausiai apsieina be jokių teorijų. Tarkime, kad taip ir yra iš tiesų. Tarkime, kad bendrovė dirba sėkmingai, gauna vis daugiau užsakymų, pradeda plėsti savo veiklą ir priiminėti naujus darbuotojus. Pamažu ji auga. Vieną dieną staiga prasideda nesėkmės. Laiku nebebaigiami projektai, jiems vykdyti nebepakanka sutartyse numatytu pinigų, užsakovai ir vartotojai yra nepatenkinti sukurtų programų sistemų funkcionalumu, patogumu, patikimumu ar darbo greičiu. Vadovybė pradeda aiškintis nesėkmėmis priežastis ir suvokia, kad reikia griežčiau reglamentuoti ir kontroliuoti darbo procesus. Prie esamo darbuotojų skaičiaus toliau senais metodais dirbtį nebeįmanoma. Kaip dar prieš porą šimtų metų rašė žinomas vokiečių filosofas Hēgelis, kiekybė visuomet peraugia į kokybę. Tikriausiai firmos vadovybei ką nors apie tai teko girdėti universitete, bet, aišku, jie seniai tą pamiršo, o ir tada vargu ar labai susidomėjo tuo, ką paisto koks nors filosofas. Tačiau, jei bendrovės dar neištiko bankrotas, joje pradedama įvedinėti tvarką. Greičiausiai tai nepadeda ir vieną dieną kas nors prabyla apie tai, kad, pasirodo, bendrovė ne pirmoji susiduria su panašiomis problemomis. Problemos gerai žinomas, jos ir jų sprendimo būdai seniai jau išnagrinėti ir aprašyti vadoveliuose. Pasirodo, kad apie tai netgi buvo kalbama universitete išklausytuose kursuose. Bet kas gi tada turėjo laiko domėtis visokiais ten procesais, gyvavimo ciklo modeliais ir kitais panašiais niekais, o juo labiau sėdėti paskaitose? Šiaip ar taip bendrovėje pradedama diegti ISO 9000 [ST18], gebėjimų brandos modelis [87] ar kita kokybės valdymo sistema <Trm25>. Tenka pradėti domėtis programų sistemų gyvavimo ciklo modeliais, programų sistemų kūrimo technologiniai procesais, o tuo pačiu ir reikalavimų inžinerijos procesais. Bendrovėje prasideda pertvarka, gyvenimas keičiasi iš esmės. „Praktikai“ į visas tas naujoves žvelgia su panieka, joms priešinasi, o dažniausiai jau ir nebegali kitaip dirbtį. Jie arba pereina į kitas mažas bendroves, dirbančias amatiniais metodais (tačiau ten jau yra savi „praktikai“), arba, kaip dažniausiai ir atsitinka, apskritai „lieka už borto“.

Bendruoju atveju, reikalavimų inžinerijos procesas, kaip ir bet kuris kitas technologinis procesas, yra sudėtingas daugiaaspektis reiškinys (37 pav.). Jis grindžiamas vienomis ar kitomis nuostatomis, kuriomis remiantis yra apibrėžtas reikalavimų aiškinimosi, formulavimo, analizės, specifikavimo ir tvarkymo veiklų turinys, jų vykdymo tvarka (proceso žingsniai) ir užduotys, kurias reikia atliliki, atliekant vieną ar kitą veiklą. 37 pav. parodyti ne visi proceso elementai. Ten neparodyta, kad procesas numato ne tik konkrečias užduotis, bet ir jų vykdymo būdus (procedūras) ir priemonės, išskaitant instrumentines priemones, skirtas toms procedūroms palaikyti. Reikalavimų inžinerijos procesus galima nagrinėti įvairiaisiais aspektais. Kaip parodyta 37 pav., IEEE ir ACM rekomendacijos SWEBOK [111], kuriomis buvo vadovautasi rašant šią knygą, rekomenduoja nagrinėti tik funkcinius

proceso aspektus (t. y. veiklas ir užduotis), paliekant nuošalyje operacinių aspektą (t. y. įeigą, išeigą, procedūras bei jas palaikančias instrumentines priemones) ir organizacinį vadybinį aspektą (t. y. proceso skaidymą į žingsnius, kontrolinius taškus, vykdytojus, vadybos procedūras). Rekomenduojama palikti nuošalyje ir proceso brandos klausimus. Kita vertus, JAV programų sistemų inžinerijos sukurtas gebėjimų brandos modelis [15] ir vėliau parengtas patobulintas ir išplėstas jo variantas [1] (šiuos modelius aptarsime 3.8 poskyryje) į pirmą vietą iškelia proceso brandos klausimus ir jo operacinių bei organizacinį vadybinį aspektus.



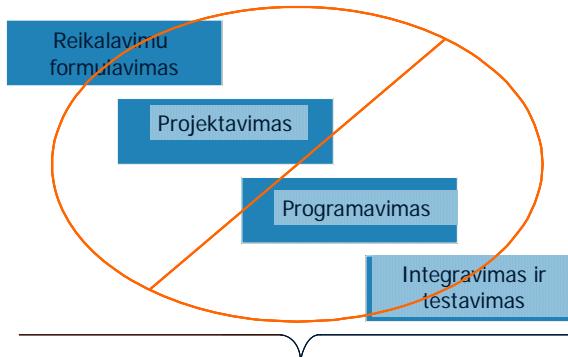
37 pav. Proceso modelis

Be to yra daug skirtinį reikalavimų inžinerijos procesų, tiksliau, tokio procesų modelių. Modelis – tai apibendrintas procesas, kuris, diegiant jį konkrečioje organizacijoje, yra pritaikomas tos organizacijos poreikiams. Visi to pačio modelio pagrindu sukurti procesai yra labai panašūs ir skiriasi tik kokiomis nors detaliemis. Todėl svarbu yra pasirinkti tinkamą reikalavimų inžinerijos proceso modelį. Tačiau, kadangi reikalavimų inžinerijos procesas visuomet yra naudojamas kartu su projektavimo, programavimo ir kitais programų sistemos kūrimo technologiniais procesais, o šie, savo ruožtu, tampriai susiję su naudojamu programų sistemos gyvavimo ciklo modeliu ir su organizacijoje veikiančia kokybės valdymo sistema, tai, savaime suprantama, renkantis reikalavimų inžinerijos proceso modelį, būtina atsižvelgti į tai, kokioje aplinkoje jis bus naudojamas. Apie tai mes dar kalbėsime 3.6 ir 3.7 poskyriuose.

Bandydami suderinti visus šiuos dalykus, reikalavimų inžinerijos proceso nagrinėjimą pradėsime nuo konkretių tokio modelių pavyzdžių. Po to šiame skyriuje aptarsime proceso organizacinių aspektų ir jo kokybės bei brandos klausimus. Funkcinių ir operacinių aspektus aptarsime 4, 5, 6, 7 ir 8 skyriuose.

3.2 Reikalavimų specifikacijos traktuotės

Iki 1993 metų buvo manoma, kad reikalavimų specifikacija tėra tik priedas prie užsakovo ir vykdytojo sandorio, nustatantis, kokią programų sistemą vykdytojas įsipareigoja pateikti užsakovui. Remiantis šiuo požiūriu ir buvo suformuluota reikalavimo, kaip formalaus vykdytojo įsipareigojimo užsakovui samprata. Šitaip suprantant reikalavimus, pakanka juos išsiaiškinti, suformuluoti ir dokumentuoti. Tuo ir baigtusi darbas su reikalavimais. Po to jais būtų tik naudojamas.



38 pav. Tipinės „krioklio“ modelio stadijos

Šitaip darbą su reikalavimais traktuoją, pavyzdžiui, klasikinis programų sistemos gyvavimo ciklo modelis (38 pav.), vadinas „krioklio“ modelis. Beje, dauguma reikalavimų inžinerijos vadovelių remiasi išreikštiniu būdu nesuformuluota prielaida, jog sistema yra kuriamā naudojamas būtent „krioklio“ modeli. Jeigu taip ir nėra daroma, vis vien remiamasi prielaida, kad gyvavimo ciklo modelis numato specialią reikalavimų inžinerijos stadiją [8]. Taip yra todėl, kad tada yra akivaizdu, kaip programų sistemos projektavimo metu efektyviai pasinaudoti reikalavimų inžinerijos rezultatais.

Modernesniuose gyvavimo ciklo modeliuose, tarkime, evoliuciniame gyvavimo ciklo modelyje, reikalavimo inžinerijos ir projektavimo veikų interfeisai nėra tokie aiškūs arba, kitaip tariant, kaip derinti reikalavimų inžineriją su kitomis programų sistemos kūrimo veiklomis nėra akivaizdu. Tačiau, klasikinis gyvavimo ciklo modelis turi daugelį gerai žinomų trūkumų ir vadovautis šia metodika kuriant programų sistemas galima tik tam tikrais atvejais.

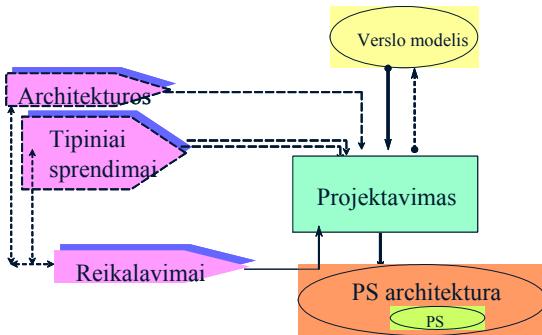
Grįžtant prie reikalavimų specifikacijos sampratos, reikia pasakyti, kad, dirbant pagal bet kurį programų sistemos gyvavimo ciklo modelį, traktuoti reikalavimų specifikaciją kaip sudėtinę sandorio dalį nėra tikslingo, nes dėl to kyla keletas sunkiai išsprendžiamų problemų. Visų pirma, realiuose projektuose reikalavimų nepavyksta “įšaldyti”. Dėl įvairių objektyvių ir subjektyvių priežasčių projekto eigoje jie nuolat kinta. Todėl tenka keisti reikalavimų specifikaciją. Traktuojant ją kaip neatskiriamą sandorio dalį, keičiant specifikaciją tektų keisti ir patį sandorį, dėl ko kiltų nepageidautinų teisinių ir techninių problemų. Antra, ir užsakovą, ir vykdytoją varžo finansiniai ir laiko ribojimai. Neretai užsakovui svarbiau yra laiku gauti sistemą su ne visomis reikalautomis funkcijomis, negu pavėluotai gauti visus pradinius reikalavimus tenkinančią sistemą. Kitaip tariant, gana dažnai produkto ir proceso reikalavimai pradeda konfliktuoti ir tokius konfliktus tenka spręsti keičiant produkto reikalavimus. Dar daugiau problemų kyla kuriant programų sistemas ne konkretiems užsakovams, o norint jas parduoti rinkoje. Tokiais atvejais programų sistemos sėkmę lemia ne jos funkcinės galimybės ir netgi ne jos patikimumas ar našumas, o visai kiti veiksnių. Pavyzdžiui, svarbiau yra tinkamu laiku pradėti pardavinėti produktą rinkoje, negu pradėti pardavinėti geresnį produktą, kuomet visi jau yra įsigiję analogiškus konkurentų pardavinėjamus produktus. Dėl visų šių priežasčių reikalavimų specifikaciją buvo nustota traktuoti kaip sudėtinę sandorio dalį, suskirstant specifikacijoje reikalavimus į grupes pagal jų svarbą ir numatant sandoryje, kad turi būti realizuota tiek reikalavimų, kiek tai galima padaryti numatytais terminais už numatytais pinigus. Tai pakeitė darbą su reikalavimais, nes dirbant šitokiu būdu nebepakanka vieną kartą suformuluoti reikalavimus, juos aprobuoti ir po to užsiimti tik jų įgyvendinimu. Reikalavimus tenka nuolat peržiūrėti,

atsižvelgiant į realią projekto eiga. Be to, reikalavimai ir projektavimo metu priimami sprendimai dažnai taip pat esti glaudžiai susiję ir gali konfliktuoti vien su kita. Ypač dažnai tai nutinka kuriant sistemas, kurių didžioji dalis arba netgi jos visos ištisai yra surenkamos iš gatavų komponentų. Iš pradinių sistemos reikalavimų išplaukiantys kurio nors komponento reikalavimai ir reikalavimai, kuriuos tenkina turimas komponentas, dažniausiai nesutampa ir todėl tenka arba pritaikyti komponentą prie sistemos reikalavimų, arba keisti sistemos reikalavimus, atsižvelgiant į turimo komponento ypatumus. Trumpai tariant, šiandien darbas su reikalavimais vyksta visą programų sistemos kūrimo laiką ir reikalavimų inžinerija susideda ne tik iš reikalavimų aiškinimosi, formulavimo ir dokumentavimo, bet ir iš daugelio kitų veikų. Todėl, kaip rašo Jim Van Buren ir David A. Cook, šiandien "*suprasti tai, kaip skirtinges reikalavimų inžinerijos veiklos yra susietos tarpusavyje ir kaip jos padeda viena kitai, yra nemažiau svarbu, negu suprasti konkretių veiklų technines detales.*" [114].

Kaip tarpusavyje siejamos reikalavimų inžinerijos veiklos projektuose, kuriuose programų sistemos yra kuriamos pagal konkretių užsakovų užsakymus, šiandien jau yra žinoma pakankamai daug. Anot žinomo reikalavimų inžinerijos specialisto Anthony Finkelstein [32], didžioji dauguma darbų reikalavimų inžinerijos srityje, išskaitant ir daugumą vadovelių, yra skirti būtent tokio pobūdžio projektams, nors juose apie tai paprastai ir nėra kalbama. Taip atsitiko todėl, kad dauguma tokų publikacijų autorių, ypač pirmaisiais kompiuterių eros dešimtmečiais, buvo vienaip ar kitaip susiję su gynybos ar karo pramonės struktūromis, o programų sistemos šitoms struktūroms paprastai yra kuriamos pagal užsakomuosius projektus. Pirmają programų sistemų inžinerijos konferenciją taip pat organizavo NATO. Tačiau iš tiesų užsakomieji projektai sudaro tik tam tikrą visų projektų dalį. Daugybė programų sistemų yra kuriamos vykdant ir kitokio pobūdžio projektus. Programų sistemos yra kuriamos parduoti rinkoje, pagal vidinius projektus pačių organizacijų vidiniams poreikiams tenkinti, adaptuojant rinkoje parduodamas sistemas konkretios organizacijos poreikiams, kooperuojantis kelioms organizacijoms kartu kuriančioms kokią nors didelę programų sistemą ir kt. Skirtingo tipo projektuose su reikalavimais yra dirbama skirtingai, skirtingi yra ir reikalavimų inžinerijos procesai. Tačiau, kaip ir dauguma kitų autorių, šiame vadovelyje daugiausia dėmesio skirsime procesams, pritaikytiems naudoti užsakomuosiuose projektuose, nes kitų tipų projektuose naudojami procesai kol kas mokslinėje literatūroje yra mažai išnagrinėti.

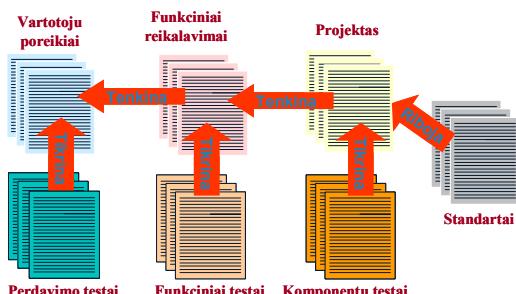
3.3 Reikalavimų inžinerijos proceso sąsajos su kitais programų sistemų inžinerijos procesais

Reikalavimų inžinerijos procesas yra tamprai susipynęs su kita programų sistemos kūrimo procesais [23], [106]. Ankstesniuose skyriuose mes jau trumpai aptarėme kaip susipina reikalavimų formulavimo ir projektavimo procesai. Matėme, kad nežinant iš kokių komponentų bus sudaryta kuriamoji IS arba, kitaip tariant, nepradėjus tos sistemos projektuoti, negalima suformuluoti jos programinės įrangos reikalavimų. Panašiai yra ir žemesniuose abstrakcijos lygmenyse. Pavyzdžiu, nežinant iš kokių komponentų bus sudaryta kuriamoji programų sistema, tai yra, nepradėjus tos sistemos projektuoti, negalima suformuluoti jos komponentų reikalavimų.



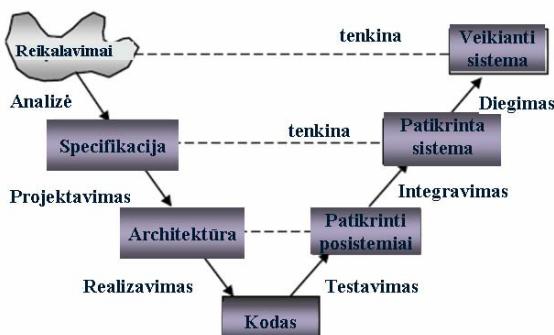
39 pav. Reikalavimų panaudojimas projektuojant programų sistemą.

Kita vertus, neturint programų sistemos reikalavimų, negalima tos sistemos suprojektuoti (39 pav.). Be reikalavimų taip pat yra neįmanoma suprojektuoti testų ir patikrinti, ar surinktoji programų sistema tenkina operacinius vartotojų poreikius (40 pav.).



40 pav. Reikalavimų panaudojimas programų sistemos testams projektuoti ir tai sistemių testuoti.

Taigi, su reikalavimais yra dirbama visose programų sistemos kūrimo stadijose (41 pav.). Remiantis vartotojo reikalavimais yra projektuojami testai, kurie, perduodant sistemą užsakovui ir prieš pradedant ją diegti to užsakovo organizacijoje, yra naudojami patikrinti sistemos tinkamumą vartotojams. Kita vertus, programų sistemos reikalavimų specifikacija irgi yra gaunama iš vartotojo reikalavimų ir, be kita ko, yra naudojama projektuojant funkcinius ir nefunkcinius testus, naudojamus patikrinti sistemos savybėms baigus integruoti tos sistemos komponentus į vieną visumą ir prieš pradedant baigiamuosius bandymus, atliekamus perduodant sistemą užsakovui. Reikalavimų specifikacija yra naudojama suprojektuoti sistemos architektūrą, nustatyti iš kokių komponentų ji turi būti sudaryta ir suformuluoti tų komponentų reikalavimus. Komponentų reikalavimai yra naudojami projektuojant testus, kurie, savo ruožtu, yra naudojami tų komponentų kodui testuoti.



41 pav. Reikalavimų vaidmuo skirtingose programų sistemos kūrimo stadijose.

3.4 Reikalavimų inžinerijos proceso modeliai

3.4.1 Reikalavimų inžinerijos proceso modelio samprata

Reikalavimų inžinerijos proceso modelis aprašo kaip reikalavimų inžinerijos veiklos yra susiję tarpusavyje ir kaip jas konfigūruoti, atsižvelgiant į skirtingus projektų tipus ir į tų projektų ribojimus. Nagrinėjant reikalavimų inžinerijos proceso modelį, siekiama geriau suvokti patį procesą. Anksčiau buvo manoma, kad reikalavimų inžinerija apima tik reikalavimų *išsiaiškinimą*, *formulavimą* ir *dokumentavimą* arba, kaip dažnai yra sakoma, reikalavimų *specifikavimą*. Vėliau šis veikų rinkinys buvo išplėstas, pridedant reikalavimų *analizę* ir *verifikavimą* bei *vertinimą*. Dar vėliau – pridedant reikalavimų *modeliavimą* ir *tvarkymą*. Šiuolaikinė reikalavimų inžinerijos samprata (žr. [63], [95], [106], [107]) papildomai apima reikalavimų *klasifikavimą pagal prioritetus*, reikalavimų *anotavimą*, reikalavimų, iškaitant ir neišsamius bei konfliktuojančiuos reikalavimus, *igyvendinamumo analizę*, *reikalavimų ir projektavimo sprendimų tarpusavio priklausomybės klausimus*, bei darbo su reikalavimais, kuriant programų sistemas ne konkrečiam užsakovui, o tikslu parduoti jas rinkoje, metodikas. Taigi, reikalavimų inžinerijos proceso modeliai paprastai apima ne tik tiesiogines darbo su reikalavimais veikas, bet ir gretimas, tokias kaip, pavyzdžiui, *marketingas* ar reikalavimų *igyvendinamumo analizę*¹¹. [63]; [95]; [106]; [107].

Kol reikalavimų specifikacija buvo traktuojama kaip neatskiriamā sandorio tarp programų sistemų kuriančio vykdytojo ir tos sistemos užsakovo dalis, nebuvo būtinybės klasifikuoti reikalavimus pagal jų prioritetus arba, kitaip tariant, pagal jų svarbą užsakovui. Vykdytojas paprasčiausiai privalėjo įgyvendinti visus reikalavimų specifikaciją numatytais reikalavimais. Atsisakius šio požiūrio ir iškeliant į pirmą vietą projekto terminus arba jo kainą, kai kurių reikalavimų tenka atsisakyti. To negalima padaryti sudarant sandorį, nes tikrosios darbų apimtys ir jų tikroji kaina neretai paaiškėja tik projekto eigoje. Tada tenka spręsti, kurių reikalavimų galima atsisakyti, o tam reikia juos suklasifikuoti pagal prioritetus.

Kuriant programų sistemas ne konkrečiam užsakovui, o tikslu parduoti jas rinkoje, reikalavimų formulavimui didžiulę įtaką daro rinkos analizės rezultatai. Programų sistemos patraukumas potencialiems pirkėjams, o tai reiškia ir jos komercinę sėkmę, dažnai priklauso ne nuo tos sistemos funkcionalumo ir netgi ne nuo jos patikimumo ar našumo, o visiškai nuo kitų veiksniių, pavyzdžiui, nuo tuo momentu rinkoje vyraujančios mados. Kitaip tariant, sistemos paklausą gali lemti ne jos duodama nauda, o visuotinas susižavėjimas kokia nors technologija ar pirkėjo noras įsigyti prestižinį gaminį. Todėl reikalavimų formulavimas šiuo atveju yra labai sudėtingas, jis tampriai susipina su marketingo klausimais. Be to, šiuo atveju reikalavimų formulavimas tampa ir labai atsakinga užduotimi, nes jis didžiaja dalimi gali nulemti bendrovės komercinę sėkmę ar nesėkmę ar netgi visišką jos žlugimą.

Taigi, šiandien reikalavimų inžinerijos procesas yra traktuojamas ne kaip veikla, vykdoma tik pradinėje PS gyvavimo ciklo stadijoje, bet kaip veikla, iniciuota to ciklo pradžioje ir besitęstanti visą projekto vykdymo laiką. Programų sistemos reikalavimai yra traktuojami kaip lygiaverčiai kuriamos sistemos konfigūracijos elementai ir kartu su kitais sistemos elementais dalyvauja visuose konfigūracijos valdymo procesuose bei yra keičiami, siekiant atsižvelgti į besikeičiančius užsakovo ir

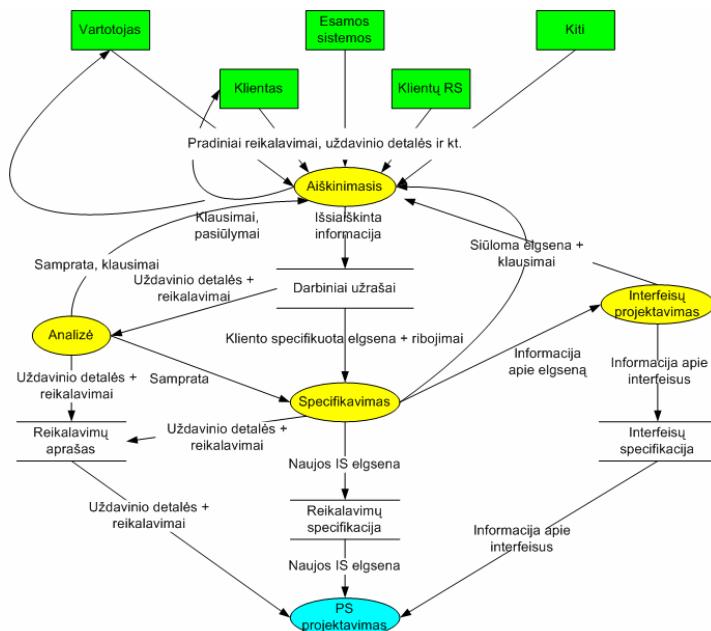
¹¹ Angliškas terminas „*feasibility study*“ į lietuvių kalbą dažnai yra verčiamas ne kaip „*reikalavimų igyvendinamumo analizę*“ bet kaip „*galimybų studija*“.

paties PS kūrimo projekto poreikius. Pats reikalavimų inžinerijos procesas arba, tiksliau, tą procesą sudarančios veiklos (reikalavimų formulavimas, analizė, specifikavimas ir kt.), taip pat gali būti konfigūruojamas, pavyzdžiui, tam, kad tenkinti konkretaus projekto ribojimus, arba tam, kad pritaikyti tą procesą skirtingu tipu projektams.

Trumpai aptarsime keletą populiariausiu reikalavimų inžinerijos proceso modelių.

3.4.2 I.K. Bray reikalavimų inžinerijos proceso modelis

Tai vienas iš geriausiai apgalvotų RI procesų modelių. Jis buvo pasiūlytas darbe [8]. I.K. Bray siūlo vaizduoti modelį duomenų srautų diagrama (42 pav.). Diagramą sudaro strėlėmis sujungtos trijų tipų viršūnės. Stačiakampiai vaizduoja reikalavimų šaltinį, ovalai – reikalavimų inžinerijos proceso veiklas, dviem lygiagrečiom atkarpom – informacija, sukaupta vykdant reikalavimų inžinerijos procesą (klasikinėse duomenų srautų diagramose šitaip žymimos duomenų saugyklos). Strėlės vaizduoja informacijos srautus.



42 pav. I.K. Bray reikalavimų inžinerijos proceso modelis (paimta iš [8]).

Kaip parodyta 42 paveikslėlyje, I.K. Bray modelis numato reikalavimų aiškinimąsi, reikalavimų analizę, reikalavimų specifikavimą ir vartotojo interfeisių projektavimą. Iš tiesų dar yra numatytas reikalavimų vertinimas, tačiau ši veikla paveikslėlyje neparodyta, nes ji vyksta kartu su kitomis veiklomis ir nėra siejama su jokia savarankiška RI proceso stadija. Trumpai aptarsime kiekvieną iš šių veiklų.

Reikalavimų aiškinimosi¹² paskirtis yra surinkti informaciją apie kompiuterizuojamą verslo sistemą. Norint surinkti tokią informaciją, reikia atsakyti į tris klausimus: kokia informacija yra reikalinga, iš kur galima ją gauti ir kokius metodus reikia naudoti tai informacijai rinkti. I pirmajį klausimą iš karto atsakyti yra neįmanoma, nes paprastai pačioje pradžioje tiek apie verslo sistemą, tiek ir apie problemas, su kuriomis ji susiduria, yra žinoma labai nedaug. Tiktai surinkus ir išanalizavus pradinę informaciją apie verslo sistemą, pradeda aiškėti, kokios dar

¹² Anglų kalboje šiai veiklai įvardinti yra vartojami keli skirtini terminai: *elicitation, requirements gathering, requirements capture* ir *requirements acquisition*. I.K. Bray vartoja terminą *elicitation*.

informacijos reikia. Pradinė informacija gi yra renkama vadovaujantis mūsų jau aptartomis bendrosiomis teorinėmis nuostatomis, sakančiomis, kad reikia susipažinti su verslo misija ir surinkti informaciją, kurios prieiks apibrėžti matams, skirtiems misijos įgyvendinimo sėkmeli matuoti, išoriniams verslo sistemos aspektams analizuoti, verslo problemoms bei grėsmėms aiškintis ir vartotojų operaciniams poreikiams nustatyti. Taigi, renkamos informacijos pobūdis priklauso nuo konkrečios verslo sistemos ypatumų. Šaltiniai iš kurių bus renkama informacija taip pat nuo tos sistemos ypatumų. Bendruoju atveju tai būsimieji sistemos vartotojai, verslo sistemos klientai ir kitos kuriamaja sistema suinteresuotos šalys. Taip pat yra būtina išsiaiškinti, kokiomis programų sistemomis jau naudojasi analizuoamoji verslo sistema ir kokie yra kuriamosios sistemas sąveikos su tomis sistemomis reikalavimai. Be abejo yra būtina susipažinti ir su verslą reglamentuojančiais normatyviniais dokumentais bei kitais su analizuojama verslo sistema susijusiais dokumentiniais šaltiniais. Renkant informaciją iš vartotojų ir kitų kuriamaja programų sistema suinteresuotų šalių, negalima pamiršti, kad kiekviena iš suinteresuotų šalių turi tam tikrus interesus ir tai daro poveikį jų pateikiamai informacijai. Todėl ši informacija niekuomet nėra visiškai objektyvi.

Renkamos informacijos pobūdis ir reikalavimų šaltiniai apsprendžia kokius informacijos rinkimo metodus tikslinga naudoti. Proceso modelis leidžia naudoti visus reikalavimų aiškinimosi metodus, aptariamus poskyryje 4.1. Galutinis reikalavimų aiškinimosi stadijos rezultatas yra darbiniai užrašai (angl. *elicitation notes*). Šie užrašai nėra koks nors oficialus dokumentas, nors jų forma dažniausiai ir yra vienaip ar kitaip formalizuota. Pavyzdžiu, tai gali būti interviu protokolai, apklausos anketos ar kokie nors kiti nustatytos formos dokumentai. Tačiau dalis medžiagos gali būti sukaupta taip pat ir garso ar vaizdo įrašų pavidalu. Todėl mūsų vartojamas terminas „darbiniai užrašai“ nėra labai tikslus. Visa surinkta medžiaga turi būti organizuota taip, kad iš jų būtų galima daryti nuorodas kituose dokumentuose.

„Darbiniai užrašai“ yra pradiniai duomenys kitai I.K. Bray modelio veiklai – reikalavimų analizei (42 pav.). Terminas *reikalavimų analizė* nėra tikslus. Jis susiklostė istoriškai ir prastai nusako, kas iš tiesų yra daroma. Tikslesnis būtų terminas *verslo sistemos analizė*, nes yra ne tiek analizuojami kuriamos sistemas reikalavimai, kiek nagrinėjami verslo sistemos ypatumai, aiškinamasi kaip ji veikia ir bandoma atskleisti ir aprašyti su kokiomis problemomis susiduria verslas ir kokios grėsmės jam gresia.

Kai kurie autoriai, pavyzdžiu, Benjamin Kovitz [64], reikalavimų analizę skaido į dvi dalis. Pirmoji apima informacijos rinkimą ir verslo sistemos bei jos problemų aiškinimąsi, antroji – išsiaiškintos informacijos perteikimą inžineriniam personalui. Yra pabrėžiama, kad renkant informaciją ir bendraujant su užsakovais vyksta ne tik informacijos rinkimo, bet ir mokymosi procesas, nes be to būtų neįmanoma suprasti kaip vyksta analizuojamasis verslas. Toks požiūris išryškina glaudžius analizės ir aiškinimosi veiklų sąryšius, bet, įjungdamas aiškinimąsi į analizę, nebepripažista aiškinimosi kaip savarankiškos veiklos arba, tariant I.K. Bray žodžiais, ignoruoja aiškinimosi kaip savarankiškos veiklos tapastį [8]. Tačiau, kitų autorų nuomone, šitaip elgtis yra netikslingo, nes aiškinimosi stadijoje iškyla savos specifinės problemas ir toms problemoms spręsti yra naudojami specialiai tam skirti metodai. I.K. Bray irgi traktuoja reikalavimų analizę ir aiškinimąsi kaip dvi savarankiškas veiklas. Tačiau, jo nuomone, Benjamin Kovitz požiūris vis tik yra tuo svarbus, kad jis išryškina bene svarbiausiąjį analizės aspektą – mokymąsi. Kitaip tariant, Benjamin Kovitz vienas iš pirmųjų atkreipė dėmesį į tai, kad aiškinantis reikalavimus reikėtų naudoti taip pat ir mokymosi metodus. Mūsų nuomone, ne

mažiau svarbu yra suprasti taip pat ir tai, kad atliekant analizę yra vykdomi dalykinės srities tyrimai ir kad nepaisant to fakto, jog tie tyrimai nėra mokslinio pobūdžio, juos vykdant galima ir reikia naudoti tradicinius tyrimų metodus. Visa tai reiškia, kad reikalavimų aiškinimosi negalima traktuoti kaip mechaninio informacijos rinkimo ir fiksavimo proceso ir kad, ruošiant analitikus, juos be kita ko reikėtų supažindinti ir su mokymosi bei tyrimų metodais. Taigi, mokslinėje literatūroje kai kurį autorių pateikti siūlymai kompiuterizuoti reikalavimų aiškinimosi procesą, pateikiant užsakovams specialias dialogines formas, užpildžius kurias reikalavimai automatiškai būtų sukaupiami atitinkamose duomenų bazėse ir analizuojami bei apdorojami programiškai, iš tiesų yra nerealūs. Asmeninis analitiko bendravimas su užsakovais, be kurio neįmanomas gilus verslo sistemos bei jos problemų supratimas, yra būtini reikalavimų aiškinimosi proceso elementai. Kadangi tai intelektuali veikla, ją kompiuterizuoti, bent jau artimiausioje ateityje, yra neįmanoma. Grįžtant prie reikalavimų analizės, reikia pasakyti, kad nors analizės mechanizmai dar ir nėra iki galo suvokti, svarbiausiais iš jų yra kompiuterizuojamos verslo sistemos konceptualizavimas ir jos ontologijos sudarymas. Aišku, vien šito nepakanka, nes yra siekiama ne tik suvokti esamą verslo sistemą, bet ir pasiūlyti kaip ją reikia tobulinti arba, kitaip tariant, suformuluoti patobulintos verslo sistemos reikalavimus. Kadangi verslo sistemą norima tobulinti diegiant į ją kuriamąją programą sistemą, tai patobulintos verslo sistemos reikalavimai turi būti formuluojami tos programų sistemos generuojamų verslo efektų terminais. Paprastai tai yra daroma kuriant atitinkamus modelius, kurie, viena vertus, turi būti pakankamai visapusis, nes juose turi būti visa reikalinga informacija apie kuriamos programų sistemos generuojamus verslo efektus, ir, kita vertus, tiek abstraktūs, kad tą informaciją būtų įmanoma aprépti ir suprasti. Svarbu nepamiršti, kad analizės metu yra nagrinėjama ir modeliuojama verslo sistema, o ne kuriamoji programų sistema. Nepaisant analizės, specifikavimo ir netgi projektavimo veiklų persidengimų ir iteracijų, analizė vis tik yra atliekama prieš pradedant specifikuoti kuriamos programų sistemos elgseną. Modeliuojant verslo sistemą visų pirma siekiama perprasti analizuojamo verslo pobūdį ir išsiaiškinti problemą, su kuriomis susiduria verslas, priežastis. Pagrindiniai analizės rezultatai yra koncepcinis verslo sistemos modelis ir trumpas kuriamos programų sistemos reikalavimų apibūdinimas (angl. *statement of the requirements*), aprašantis kokius verslo efektus turi generuoti ta sistema, kad būtų išspręstos analizės metu išryškintos verslo problemos. Kitaip tariant, tai pačio aukščiausio abstrakcijos lygmens reikalavimai, formuluojami verslo terminais. Kalbėdami apie reikalavimų rūšis, mes aprašinėjome efektus tikslų medžiu ir jų detalizuojančiais reikalavimais. Poskyryje 2.4, kalbėdami apie reikalavimų rūšis, pageidaujamus verslo efektus mes aprašinėjome tikslų medžiu ir tą medžių detalizuojančiais reikalavimais. Analizės rezultatus aprašantį dokumentą I.K. Bray vadina *reikalavimų aprašu* (angl. *requirement document*) ir skiria jį nuo *reikalavimų specifikacijos* (angl. *specification document*), dokumento, aprašančio kuriamos programų sistemos elgseną. Ši terminija atspindi asmeninį I.K. Bray požiūrį. Kituose reikalavimų inžinerijos modeliuose šie dokumentai gali būti vadinami ir traktuojami kitaip.

Trečioji I.K. Bray modeliu numatyta veikla yra reikalavimų specifikavimas (42 pav.). I.K. Bray modelyje reikalavimų specifikavimas suprantamas kaip pageidaujamus verslo efektus generuojančios sistemos elgsenos konstravimas ir aprašymas. Tai labiau kūrybinė negu techninė veikla, nes specifikuojamos programų sistemos dar nėra ir jos elgseną reikia „išrasti“. Užsakovas, bent jau iš dalies, gali suformuluoti verslo problemas, galbūt gali pateikti pradinius kuriamos programų sistemos reikalavimus, suformuluotus verslo terminais, bet tik išskirtiniais atvejais jis

diktuoja kokia turi būti tuos reikalavimus tenkinančios sistemos elgsena. Kita vertus, užsakovas turi aprobuoti programų sistemų inžinieriaus siūlomą elgseną. Be to, reikia turėti omenyje, kad tuos pačius verslo efektus gali generuoti skirtingos elgsenos, t. y. galima parengti kelias skirtingas programų sistemos reikalavimų specifikacijas. Todėl tam, kad galima būtų parinkti vieną iš jų, reikia suformuluoti labiausiai tinkamos elgsenos parinkimo kriterijus. I.K. Bray traktuoja reikalavimų specifikavimą kaip projektavimo proceso dalį. Tačiau tai yra išorinis, o ne vidinis sistemos projektavimas.

Anglų kalboje programų sistemos reikalavimų specifikacijai apibūdinti vartojama daug skirtinguų terminų: *requirements specification*, *system requirements specification*, *requirements definition*, *functional requirements definition* ir kt. I.K. Bray vartoja terminą *specification document*. Tai vienas iš svarbiausių dokumentų, nes jis aprašo, ką gi vykdytojai ketina pateikti užsakovui. Panašūs dokumentai turi būti rengiami ir kiekvienam iš tos sistemos komponentų.

Ketvirtoji I.K. Bray modeliu numatyta veikla yra interfeisių projektavimas (42 pav.). Nors vartotojų interfeisių projektavimas yra sudėtinė sistemos išorinio, o ne vidinio projektavimo dalis, iš tiesų specifikuojant programų sistemą interfeisių projektavimas yra tik tai pradedamas. Interfeisus reikia suprojektuoti daug detaliau, negu tai yra padaroma formuluojuojant jų reikalavimus (angliškai sakoma „*look and feel*“ lygmenyje). Be to, detalus interfeisių projektavimas yra specialaus pobūdžio veikla, reikalaujanti specialių žinių, išskaitant ir mokėjimą naudotis tam tikromis technologijomis. Yra ir kitų priežasčių, kodėl I.K. Bray modelyje interfeisių projektavimo stadijoje interfeisai nėra suprojektuojami iki galo. Detali vartotojo interfeisių specifikacija būtų didelės apimties dokumentas. Be to, programų sistema turi ne tik vartotojo, bet ir kitus išorinius interfeisus, pavyzdžiui, sąveikai su kitomis programų sistemomis arba sąveikai su įrenginiais. Tai dar padidina interfeisių specifikacijos apimtis. Taigi, įjungus visą tai į programų sistemos reikalavimų specifikaciją, interfeisių specifikacija sudarytų didesnį reikalavimų specifikacijos dalį, užgožtų kitus klausimus ir nustumtų juos į antrą planą.

Paskutinė I.K. Bray modeliu numatyta veikla yra reikalavimų vertinimas (angl. *validation*). 42 paveikslėlyje ji neparodyta, nes, kaip jau minėjome, reikalavimų vertinimas nėra savarankiška stadija. Ši veikla „išsibarsto“ po kitas stadijas. Taip yra todėl, kad reikalavimų vertinimas atliekamas tikslu rasti ir pašalinti klaidas, o klaidos gali būti ir būna daromos bet kurioje reikalavimų inžinerijos proceso stadijoje. Jos daromos dėl nesusikalbėjimo, dėl dviprasmybių skaitomuose dokumentuose arba paprasčiausiai dėl nepakankamo atidumo. Be abejo yra metodų, padedančių apsisaugoti nuo klaidų, tačiau visai jų išvengti yra neįmanoma. Kadangi reikalavimų inžinerijos metu padarytas klaidas vėliau ištaisyti yra labai sunku ir brangu, tai savaime aišku, jog reikia pasistengti jas rasti ir ištaisyti kuo anksčiau.

Atliekant reikalavimų vertinimą, visų pirmą ieškoma klaidų, susijusių su kuriamos programų sistemos funkcionalumo aprašymu, arba, kitaip tariant, yra siekiama užtikrinti, kad kuriamoji programų sistema turėtų tokį funkcionalumą, kurio iš tiesų reikia vartotojams. Vertinant reikalavimus yra vadovaujamasi šitokia samprotavimų schema: „Jei verslo sistema funkcionuoja taip, kaip aprašyta jos modelyje, ir jei reikalavimai suformuluoti teisingai ir jei kuriamoji programų sistema veikia taip, kaip tą numato reikalavimų specifikacija ir jei specifikavimas atliktas korektiškai, tai galima teigti, kad kuriame tai, ko iš tiesų reikia vartotojams“. Ši schema padeda nustatyti potencialius klaidų šaltinius ir sufleruoja, kad vertinant reikalavimus reikia atsakyti į tokius klausimus:

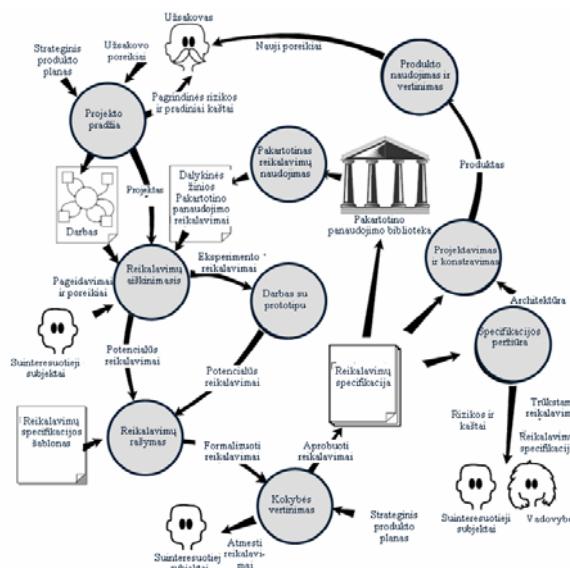
- Ar verslo sistemos modelis iš tiesų atspindi realias verslo sistemos savybes?
- Ar reikalavimai aprašo tikrai tuos efektus, kurių iš tiesų reikia užsakovui?
- Ar išorinis projektavimas atliktas teisingai, ar pasiūlyta sistemos elgsenas tikrai generuos tuos efektus, kurie numatyti reikalavimuose?
- Ar reikalavimų specifikacija tikrai aprašo tą elgseną, kuri buvo pasiūlyta išorinio projektavimo metu?

Šie klausimai parodo, jog reikalavimų vertinimo veikla gali išsibarstyti ne tik po atskiras I.K. Bray pasiūlyto reikalavimų inžinerijos modelio stadijas, bet netgi ir po atskirus tų stadijų etapus. Pavyzdžiu, paėmus interviu, asmeniui iš kurio buvo imtas interviu galima pateikti savais žodžiais atpasakotas jo mintis ir šitaip pasitikrinti, ar jis buvo teisingai suprastas. Po to, jau kitame tos pačios stadijos etape, gali būti atlikta formaliai rengiamų dokumentų juodraščių peržiūra.

Baigiant aptarti I.K. Bray pasiūlytą reikalavimų inžinerijos modelį, reikia pasakyti, kad netgi pats I.K. Bray pripažista, jog šis modelis yra idealizuotas ir apima ne visus realaus reikalavimų inžinerijos proceso aspektus. Be to galimos įvairios šio modelio variacijos arba, kitaip tariant, naudojant šį procesą realiamje projekte, jį reikia sukonkretinti, atsižvelgiant į konkretias to projekto aplinkybes. Taip pat reikia turėti omenyje, kad I.K. Bray vis dar daro prielaidą, kad reikalavimų specifikacija yra neatskiama sandorio dalis. Todėl šis proceso modelis prastai dera su reikalavimų formulavimo proceso schema, kurią mes turėjome omenyje, nagrinėdami reikalavimų rūšis 2 skyriuje. Norint dirbtini pagal šią schemą, reikalingas kitoks reikalavimų inžinerijos proceso modelis.

3.4.3 Reikalavimų inžinerijos proceso modelis „Volere“

Tai bene vienintelis reikalavimų inžinerijos proceso modelis turintis tikrinį pavadinimą.



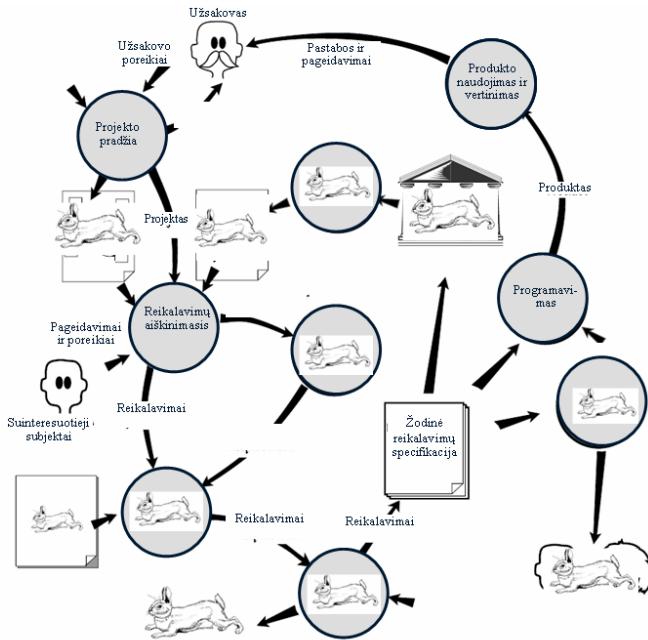
43 pav. Reikalavimų inžinerijos proceso modelis „Volere“ [95].

43 paveikslėlyje pateiktas supaprastintas „Volere“ proceso modelis. Ši modelių sukūrė Suzanne ir James Robertson’ai. Tariant jų žodžiais, „...tai reikalavimų aiškinimosi ir specifikavimo procesas, grindžiamas principais, kurie tinkamai veikia bet kuriai programų sistemai kuriamai veikia bet kurio pobūdžio projekte.“ [95]. Autoriai teigia, kad procesas gime kaip jų praktinės patirties išdava ir kad jie visų pirma siekė

sukurti analitikams skirtą praktinį darbo įrankį, padedantį dirbtį produktyviau ir tiksliau, o ne vien tik tam tikrą darbo metodiką. Suzanne ir James Robertson’ai sako: „...mes asmeniškai stebėjome šimtus kompanijų, kurios sėkmingai pritaikė šį procesą savo korporacinei kultūrai ir savo organizaciniams ypatumams, ir mes žinome apie tūkstančius kitų, kurios padarė tą patį.“ [95]. Todėl jie yra visiškai įsitikinę praktinę „Volere“ proceso modelio nauda. Šis modelis gali būti naudojamas kartu su įvairiais programų sistemų gyvavimo ciklo modeliais, išskaitant „krioklio“ modelį, ekstremalujį programavimą, RUP¹³ ir evoliucinį gyvavimo ciklo modelį. Panašiai kaip ir I.K. Bray, autoriai jų pasiūlytą modelį vaizduoja stilizuota duomenų srautų diagrama. Diagramoje parodytos modeliu numatytos veiklos ir tomis veiklomis kuriami rezultatai.

Be kita ko, „Volere“ proceso modelis buvo kuriamas planuojant panaudoti jį agiliuosiuose programų sistemų kūrimo procesuose. Programų sistemos kūrimo procesas vadinamas agiliu, jei nėra reikalaujama, kad, nepriklausomai nuo kuriamo produkto pobūdžio, būtinai būtų vykdomos visos tuo procesu numatytos veiklos. Tai reiškia, jog „Volere“ proceso modelis yra sukonstruotas taip, kad galima atesti tas veiklas arba tas atskirų veiklų dalis, kurios nėra būtinos kuriant konkretų produktą arba vykdant konkretų projektą. Be abejo, ne visi projektai yra vienodai agilūs. Didelis projektu suinteresuotų šalių skaičius, dokumentacijos poreikis, po skirtinges vietas išsibarstęs vykdytojų kolektyvas ir kiti veiksnių neišvengiamai diktuoja projekto agilumo ribas. Pagal agilumą Robertson’ai visus projektus skirsto į „triušio dydžio projektus“ (angl. *rabbit project*), „arklio dydžio projektus“ (angl. *horse project*) ir „dramblilio dydžio projektus“. „Triausio dydžio projektai“ yra didžiausio agilumo projektai. Dažniausiai tai maži projektai, kuriuose visos jais suinteresuotos šalys turi galimybę glaudžiai tarpusavyje bendradarbiauti. Tokiuose projektuose rašytinės reikalavimų specifikacijos paprastai nėra rengiamos. Todėl, naudojant tokiuose projektuose „Volere“ proceso modelį, rašytinė reikalavimų specifikacija yra keičiamā žodine specifikacija ir visos modeliu numatytos veiklos, išskyrus projekto pradžią, reikalavimų aiškinimąsi, konstravimą ir produkto naudojimą bei vertinimą, yra ignoruojamos (44 pav.).

¹³ Rational Unified Process.



44 pav. „Volere“ proceso modelio adaptavimas „triušio dydžio projektams“ (paimta iš [95]).

„Triausio dydžio projektuose“ dažniausiai yra dirbama pagal ekstremalaus programavimo (angl. *XP*) metodiką. Tokie projektai yra iteratyvūs ir juose reikalavimai yra kaupiami porcijomis, paprastai, vienai verslo užduočiai. Realizavus sukauptą reikalavimų porciją, kaupiama kita porcija ir t.t.

„Arklio dydžio projektai“ yra vidutinio agilumo projektai. Tokiuose projektuose privalu naudoti daugumą „Volere“ proceso modeliu numatytyų veiklų, tačiau modelis duoda galimybę atsižvelgti į tai, kad tokie projektai taip pat yra iteratyvūs. Juose, kaip ir „triausio dydžio projektuose“, reikalavimai yra kaupiami porcijomis. Sukaupus eilinę reikalavimų porciją, yra kuriamas einamasis sistemos prieaugis, po to kaupiama nauja reikalavimų porcija ir t.t. Tokia strategija reikalauja, kad prieš kuriant pirmajį (pradinį) sistemas prieaugi būtų suprojektuota visos sistemos architektūra. Be kitų pranašumų ji turi dar ir tą, kad ir analitikai, ir projektuotojai visą laiką yra užimti.

„Arklio dydžio projektuose“ ypač svarbios yra reikalavimų aiškinimosi, reikalavimų rašymo ir kokybės vertinimo veiklos. Jei tos veiklos yra atliekamos kruopščiai ir iteratyviai, projekto sėkmė yra beveik garantuota.

Mažiausio agilumo projektus Robertson'ai vadina „dramblio dydžio projektais“. Tai dideli projektai, kuriuose dalyvauja daug vykdytojų, išsibarsčiusių po skirtinges darbo vietas ir daug suinteresuotujų šalių. Tokiuose projektuose būtinė visiškas reikalavimų formalizavimas ir Volere proceso modelis turi būti naudojamas pilnumoje.

Volere proceso modelis nenumato RI proceso pabaigos. Atidavus užsakovui visą arba dalinį produktą ir vartotojams pradėjus su juo dirbtį, prasideda produkto evoliucija. Pradėję dirbtį su produkту, žmonės supranta, kaip dar būtų galima jį panaudoti. Taip gimsta nauji poreikiai bei reikalavimai ir prasideda naujas RI proceso ciklas.

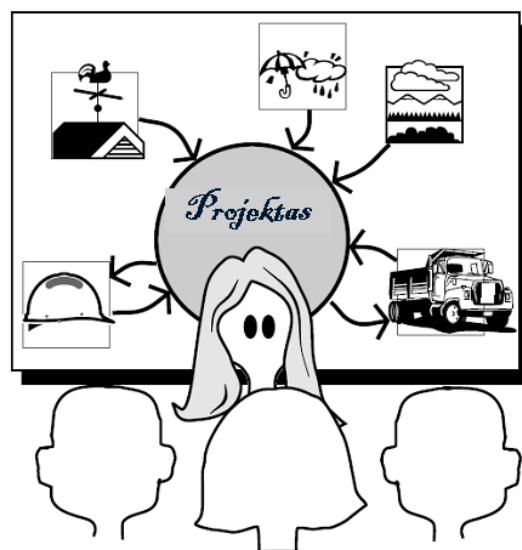
Kadangi produktas vis vien evoliucionuoja pats savaime, tikslinga visą darbo metodiką organizuoti evoliucinio gyvavimo ciklo pagrindu, t. y. kurti minimalų funkcionalumą turinčius pirmuosius produkto prieaugius ir vėliau plėsti produktą naujais iš anksto suplanuotais prieaugiais. Volere proceso modelis visų pirma yra pritaikytas būtent tokiam darbo stiliui.

Volere proceso modelių pateikėme taip, tarsi ji būtų galima naudoti tik kuriant sistemas „nuo nulio“. Iš tiesų taip nėra. Jis gerai tinktaip pat ir sistemų modernizavimo projektams, projektams, kuriuose sistemos renkamos iš gatavų komponentų, bei projektams, kuriuose reikalavimai negali būti „jšaldyti“. Tačiau kol kas į šiuos klausimus nesigilinsime ir pereisime prie Volere proceso modelio veiklų turinio aptarimo.

Projekto pradžia. Prieš pradedant aiškintis reikalavimus, reikia pasistengti suvienodinti svarbiausiuju suinteresuotujių šalių (užsakovas, pagrindiniai vartotojai, verslo konsultantai, analitikas, sistemą kuriantis inžinerinis personalas ir kt.) požiūrius į tai, kokie gi yra svarbiausieji pradedamo projekto tikslai. Efektyviausias instrumentas tą padaryti yra projektų tikslų aptarimas svarbiausiuju suinteresuotujių šalių susitikime (angl. *blastoff meeting*).

Kadangi proceso modelyje parodyta, jog tarp veiklos „Projekto pradžia“ pradinių duomenų yra strateginis produkto planas, tai reiškia, kad verslo lygmens reikalavimai jau yra suformuluoti. Kitaip tariant, Volere proceso modelyje daroma prielaida, jog pradedant dirbtį pagal šį modelį jau yra aprašyti verslo sistemos misija ir vizija, sukonstruotas verslo tikslų medis, nustatyti jo ribojimai, suformuluoti verslo objektų reikalavimai, identifikuotos sistemos paslaugų gavėjų grupės, suplanuotos jų darbo vietas ir suformuluoti verslo sistemos našumo reikalavimai. Kaip formuluoti verslo lygmens reikalavimus, Volere proceso modelis nenagrinėja. Daroma prielaida, kad tai ne RI proceso objektas.

Suinteresuotujių šalių susitikimui pirmininkauja sisteminis analitikas. Rekomenduojama svarstyti pradeti nuo verslo problemų ir siūlomo jų sprendimo būdo (t. y. tikslų medžio) aptarimo. Taigi pirmiausiai turėtų būti susitarta dėl projekto ribų bei apimties ir tuo pačiu išsiaiškinama, koki verslo sistemos fragmentą reikės aiškintis analitikams. Po to turėtų būti aptartas projekto kontekstas, t. y. tai, kaip projektas siejamas su jį supančiu pasauliu. Tam reikia nubraižyti konteksto diagramą (45 pav.) ir galutinai susitarti, kokia ji turi būti.



45 pav. Konteksto diagrammos pavyzdys kelių valymo informacinei sistemei (paimta iš [95]).

Išsamiau konteksto modeliavimo klausimus aptarsime 5.3 poskyryje.

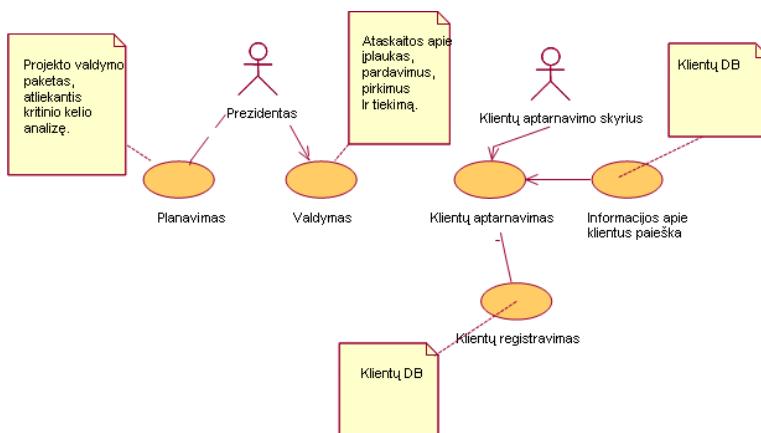
Susitarus dėl projekto ribų ir jo konteksto diagrammos, pagal tą diagramą reikia patikslinti projektu suinteresuotujių šalių sąrašą arba, kitaip tariant, potencialių vartotojo lygmens reikalavimų šaltinių sąrašą.

Suinteresuotujų šalių susitikimo metu taip pat reikia galutinai nuspręsti, ar verta pradėti projektą ir susitarti dėl projekto sėkmės matū. Patartina atlirkti preliminarų projekto kaštų ir jo īgyvendinamumo galimybių (t. y. projekto svarbiausių rizikos veiksnių) vertinimą.

Dažnai atsitinka, kad susitikimo dalyviai esti susižavėję būsimuoju projektu bei nauda, kurią duos kuriamas produktas. Reikia pasistengti šito išvengti ir labai kritiškai įvertinti projekto atsiperkamumą ir jo rizikos veiksnius. Karti daugelio projekčių patirtis rodo, jog daug geriau yra nutraukti projektą jo nepradėjus, negu tai padaryti po kelių mėnesių ar netgi kelių metų darbo, išleidus daug pinigų ir sunaudojuš daug kitų ištaklių.

Patartina vėliau, baigus aiškintis kuriamos sistemos reikalavimus, suintersuotųjų šalių atstovams susirinkti dar kartą, įvertinti tuos reikalavimus ir galbūt pakoreguoti projekto tikslus. Ypač tai svarbu, jei pirmojo susitikimo metu paaiškėja, jog galutiniams sprendimams prijimti nepakanka informacijos.

Reikalavimų aiškinimas. Jei suinteresuotujų šalių susitikime nusprendžiamą pradeti vykdyti projektą, analitikai gali pradeti aiškintis kuriamos sistemos reikalavimus. Tačiau prieš pradedant tai daryti, konteksto diagramą patartina pertvarkyti į užduočių diagramą (46 pav.). Diagramoje kaip užduotys vaizduojamos verslo transakcijos, apdorojančios atitinkamus verslo ivykius.



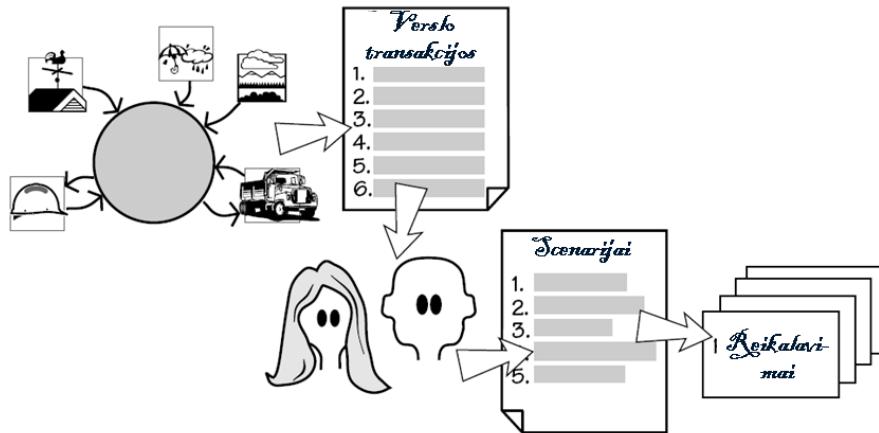
46 pav. Užduočiu diagrama, aprašanti kuriamos sistemos panaudojimą.

Kiekvieną užduočių diagramoje parodytą užduotį galima paveсти aiškintis skirtiniems analitikams. Jie gali dirbti beveik nepriklausomai vienas nuo kito. Tačiau vis vien turi būti paskirtas vyriausias analitikas, kuris privalo susumuoti analitikų gautus rezultatus ir juos apibendrinti.

Bakalauro pakopos studentams skirtuose programų sistemų inžinerijos kursuose yra aptariami interviu, apklausos ir kiti analitikų naudojami reikalavimų aiškinimosi metodai. Yra ir kitų, modernesnių metodų: pameistrystė (angl. *apprenticing*), scenarijai (angl. *scenario*), **užduočių aptarimai** (angl. *use case workshops*) ir kt.

Naudojant tameistrystės metodą, analitikas pats mokosi vykdyti verslo užduotį ir tuo pat metu aiškinasi, kokių informacinių, skaičiuojamujų, komunikavimo ar kitokij paslaugų reikia tai užduočiai vykdyti.

Aiškindamiesi reikalavimus, analitikai kartu su sistema suinteresuotomis šalimis privalo nustatyti projekto gylį, t. y. išsiaiškinti kokių mastų kiekviena iš užduočių turi būti kompiuterizuota ir kokią naudą iš to gaus kiekviena iš šalių. Informacija apie tai pateikiama komentarų forma užduočių diagramose.



47 pav. Reikalavimų formulavimas, dirbant pagal Volere proceso modelį (paimta iš [95]).

Reikalavimų aiškinimasis baigiamas suformuluojant potencialius kuriamos sistemos reikalavimus (47 pav.). Kaip parodyta paveikslėlyje, iš konteksto diagramos yra gaunamas verslo transakcijų sąrašas (vaizduojamas užduočių diagrama), kiekvienai užduočiai aprašomas jos vykdymo scenarijus ir, remiantis tais scenarijais, formuluojami kuriamos sistemos funkciniai reikalavimai. Nefunkcininiai reikalavimai yra formuluojami kartu su funkciniais reikalavimais. Kad būtų galima suformuluoti šiuos reikalavimus, aprašinėdamas užduoties vykdymo scenarijų analitikas privalo konsultuotis su panaudojamumo, operabilumo, apsaugos ir kitais ekspertais.

Volere proceso modelis numato, kad analitikai aiškinasi tik vartotojo lygmens reikalavimus. Robertson'ai juos vadina potencialiaisiai reikalavimais (angl. *potential requirements*). Kita vertus, jie rašo:

„Bene sunkiausia reikalavimų aiškinimosi dalis yra projekto esmės išsiaiškinimas. Dauguma suinteresuotųjų šalių šneka apie jų įsivaizduojamą problemų sprendimą. Tačiau esmė slypi ne sprendimuose, o verslo motyvuose turėti tokį produktą.“ [96]

Taigi, šitoje vietoje Robertson'ai yra nenuoseklūs, nes, nepaisant išreikštiniu būdu nesuformuluotos prielaidos, kad verslo lygmens reikalavimai jau yra suformuluoti, jie vėl grįžta prie tų reikalavimų aiškinimosi problemų.

Darbas su maketu. Kartais pasitaiko reikalavimų, kurie yra arba neteisingai suformuluoti, arba kurių vartotojai nesugeba paaiškinti, arba kurių analitikas nesugeba suvokti. Gali būti ir taip, kad kuriamas produktas yra toks inovatyvus, kad niekas nesugeba jam suformuluoti reikalavimų. Visais tokiais atvejais rekomenduojama reikalavimus aiškintis konstruojant sistemos mąketą. Mąketas gali būti veikiantis arba ne, pavyzdžiui, sukonstruotas popieriuje ar ekrane. Išsamiau apie mąketavimą kalbėsime 5.5 poskyryje.

Užduočių vykdymo scenarijai taip pat yra savo išskirtinis rūšis. Scenarijus – tai istorija, žingsnis po žingsnio pasakojanti, kaip yra vykdoma verslo užduotis. Scenarijai yra vienas iš būdų, padedantis kalbėtis su sistema suinteresuotomis šalimis ir aiškintis jų reikalavimus.

Reikalavimų rašymas. Viena iš didžiausių programų sistemų kūrimo problemų yra neteisingai suprasti reikalavimai. Siekiant išspręsti šią problemą, reikalavimai yra išrašomi tokia forma, kad juos būtų galima testuoti arba, kitaip tariant, tikrinti, ar jie teisingai suprasti. Be to, visos sistema suinteresuotosios šalys turi perskaityti reikalavimus ir jiems pritarti.

Kadangi kalbama apie vartotojo lygmens reikalavimus, jie turi būti formuluojami verslo terminais, t. y. sistema suinteresuotoms šalims suprantama kalba,

nes, kaip ką tik minėjome, jos turi reikalavimus vertinti ir aprobuoti. Kita vertus reikalavimai turi būti suprantami ir techniniam personalui, nes jam reikės toliau su jais dirbtį.

Siekiant taip suformuluoti reikalavimus, kad būtų galima patikrinti jų įgyvendinimą, prie kiekvieno reikalavimo pridedamas atitikimo vertinimo kriterijus (angl. *fit criterion*). Atitikimo kriterijus – tai kiekybinis matas arba kokybinė skalė, kuriu pagrindu testuotojai konstruoja testus patikrinti, ar reikalavimas yra įgyvendintas.

Dirbdamas pagal Volere proceso modelį, analitikas turi du įrankius, padedančius jam rašyti reikalavimus. Pirmasis – reikalavimų specifikacijos šablonas, nustatantis iš kokių skyrių ir poskyrių turi būti sudaryta reikalavimų specifikacija, antrasis – reikalavimo šablonas (angl. *snow card*), padedantis nustatyti, ar reikalavimas turi visas reikalingas dalis. Abu juos panagrinėsime 6 skyriuje.

Nors Volere proceso modelyje reikalavimų rašymas parodytas kaip savarankiška veikla, tai nereiškia, kad tai yra savarankiška proceso stadija. Iš tiesų reikalavimų aiškinimasis, darbas su maketu, reikalavimų rašymas ir reikalavimų kokybės vertinimas vyksta tuo pat metu. Kai kurie reikalavimo šablono laukai yra pildomi specifikacijos peržiūros metu.

Tam tikra funkcinių reikalavimų rašymo alternatyva yra reikalavimų modeliavimas. Tačiau ši alternatyva nėra rekomenduotina, nes dažniausiai modeliai yra sunkiai suprantami nespecialistams, o be to juos vis vien reikia papildyti rašytiniais nefunkciniiais reikalavimais.

Pagrindinis reikalavimų rašymo tikslas yra ne rezultatas (nors jis taip pat yra labai reikalingas), bet pats rašymo procesas. Reikalavimo ir jo atitikimo kriterijaus išrašymas padeda analitikui geriau suprasti tą reikalavimą ir pastebeti savo klaidas bei pražiopojimus. Bandymai suformuluoti iki galo nesuprastą reikalavimą generuoja beprasmybes, kurios iš karto pastebimos vertinant reikalavimų kokybę.

Kokybės vertinimas. Kalbėdami apie reikalavimų kokybės vertinimą, Robertson'ai vartoja terminą „*quality gateway*“.¹⁰ Apskritai, kiekvieną reikalavimą, prieš jį įtraukiant į reikalavimų specifikaciją, reikia įvertinti. Reikalavimų vertinimas taip organizuojamas, kad tik vienas asmuo, testuotojas arba vyriausias analitikas (jei dirba analitikų grupė), turi teisę leisti įtraukti reikalavimą į specifikaciją. Dirbdami kartu, analitikas ir testuotojas vertina kiekvieno reikalavimo išsamumą, relevantiškumą, testuojamumą, vienareikšmiškumą, trasuojamumą ir kitas savybes. Reikia patikrinti ir tai, ar visiems reikalavimams yra suformuluoti atitikimo kriterijai.

Viena iš svarbiausių kokybės vertinimo užduočių yra užtikrinti, kad reikalavimų specifikacijoje neatsirastų reikalavimų, kuriuos nežinia kas ten įrašė ir kurių reikalingumo niekas nebegali paaiškinti.

Pakartotinas reikalavimų naudojimas. Labai retai kada produktai esti visiškai unikalūs. Todėl prieš pradedant aiškinantis naujo produkto reikalavimus yra prasminga peržiūrėti anksčiau vykdytų projektų reikalavimų specifikacijas ir pažiūrėti, ar kuo nors potencialiai galima pasinaudoti naujame projekte. Kartais galima be jokių pakeitimų pakartotinai panaudoti ištisas reikalavimų grupes, kartais reikalavimus reikia tik nežymiai modifikuoti. Be abejo, tam, kad būtų galima spręsti, ar galima kokius nors reikalavimus panaudoti pakartotinai, reikia jau šį tą žinoti apie kuriamąjį produktą. Ar organizacijoje vyksta kiti projektai, skirti kompiuterizuoti panašias problemines sritis? Ar produkto vartotojai jau dirba su kokiais nors kitaip produktais ir ar panaudojamumo, interfeisių ir ergonominių reikalavimai išliks panašūs? Ar projekto ribojimai bus panašūs į kitų projektų ribojimus? Ar susitarimai dėl vardų ir apibrėžtys gali būti imami iš kitų projektų? Ar kitų projektų prielaidos ir

kiti reikšmingi faktai išlieka? Ar galima pasinaudoti kitų projektų konteksto diagramomis? Visą tą informaciją galima surinkti pradinio susitikimo su svarbiausiomis suinteresuotomis šalimis metu.

Specifikacijos peržiūra. Vertinant kokybę, reikalavimai yra vertinami po vieną. Tačiau, baigus formuoti reikalavimų specifikaciją, reikia patikrinti reikalavimų tarpusavio darną. Taip pat reikia patikrinti specifikacijos išsamumą, t. y. įsitikinti, ar specifikacijoje netrūksta kokių nors reikalavimų. Atliekant specifikacijos peržiūrą, taip pat yra patikslinami projekto kaštai ir atliekama galutinė reikalavimų įgyvendinamumo analizė. Specifikacija čia suprantama kaip kokybės kontrolę praėjusių reikalavimų rinkinys, o ne kaip dokumentas. Specifikacijos peržiūrai naudojami inspektavimo ir kiti panašūs metodai.

Inspektavimas, tiksliau Fagan'o inspektavimo metodas, yra formalizuotas dokumentų kokybės vertinimo metodas. Inspektuoja specialistų (inspektorų) grupė, kuriai išdalinama inspektuojama medžiaga. Kiekvienas grupės narys yra atsakingas už tam tikrą vertinimo aspektą ir turi klausimyną, padedantį jam rasti klaidas. Per 2-3 dienas inspektorai atsako į savo klausimynų klausimus ir aptaria atsakymus bendrame posėdyje.

Reikalavimų kokybės vertinimo ir specifikacijos peržiūros veiklos yra derinamos viena su kita. Vertinant kokybę, analizuojami individualūs reikalavimai ir žiūrima, ar jie korekiškai suformuluoti, nedviprasmiški, susiję su sprendžiamu problema, testuojami, trasuojami ir ar nėra „auksiniai“ (angl. *gold plating*), t. y. ar yra įgyvendinami protingais kaštais. Peržiūrint specifikaciją, vertinama reikalavimų visuma. Taigi, peržiūrint specifikaciją, yra tikrinama:

- ar specifikacija yra išsami, t. y. ar joje netrūksta kokių nors reikalavimų;
- ar funkciniai reikalavimai suformuluoti visoms verslo transakcijoms;
- ar nėra vienos su kitu nesuderinamų (konfliktuojančių) reikalavimų.

Peržiūrint specifikaciją taip pat yra atliekamas reikalavimų prioretizavimas, t. y. reikalavimai yra suskirstomi į grupes pagal jų svarbą užsakovui. Prioritetų reikia dėl daugelio priežasčių. Be kita ko, jie padeda programuotojams suprasti, kiek darbo tikslingo įdėti vienam ar kitam reikalavimui realizuoti ir kiek skubiai tai reikia daryti. Nebūtina laukti, kol bus sukaupti visi reikalavimai ir tikta tada pradėti juos skirtysti pagal jų svarbą. Idealiu atveju, tai turėtų būti nenutrūkstama, nuolat vykstanti veikla. Pavyzdžiui, gali būti peržiūrimi su konkretičia verslo transakcija siejami reikalavimai, po to – su kita ir t.t. Gali būti ir kitaip, pavyzdžiui, gali būti peržiūrimi su viena sistemos vykdoma užduotimi susieti reikalavimai.

Specifikacijos peržiūra yra iteratyvi ir kitu požiūriu. Radus klaidas ir jas ištasisius, specifikacija vėl yra peržiūrima, nes reikia įsitikinti, ar taisant senas klaidas nebuvo padaryta naujų klaidų. Ciklas baigiasi, kuomet peržiūrint specifikaciją nebesurandama nei vienos klaidos. Patariama registruoti pašalinamus reikalavimus, kad jie atsitiktinai vėl nebūtų įtraukti į specifikaciją ir kad būtų galima analizuoti, kokie reikalavimai ir kodėl buvo pašalinti.

Specifikacijos peržiūra yra svarbi dar vienu požiūriu. Radus specifikacijoje klaidas, galima aiškintis jų atsiradimo priežastis ir taip tobulinti RI procesą, kad ateityje būtų apsaugota nuo panašaus pobūdžio klaidų. Jei specifikacijoje yra randama labai daug klaidų arba jei peržiūros metu paaiškėja, kad kaštai ir rizikos veiksnių persveria naudą, kurios yra tikimasi, tai projektą rekomenduojama nutraukti.

Tikrinimas, ar funkciniai reikalavimai yra suformuluoti visoms verslo transakcijoms, pradedamas verslo įvykių sąrašo sudarymu. Po to verslo įvykis susiejamas su verslo transakcija ir sprendžiama apie tos transakcijos

kompiuterizavimo gylį, t. y. ką turi daryti tą transakciją kompiuterizuojanti programų sistemos užduotis.

Reikalavimus galima grupperoti pagal sistemos vykdomas užduotis. Tokia strategija yra pakankamai efektyvi, jei pavyksta nustatyti visus verslo įvykius. Trūkstant verslo įvykių, savaime aišku, trūksta ir atitinkamų reikalavimų. Bet kaip sužinoti, ar visi verslo įvykiai buvo nustatyti?

3.5 Reikalavimų inžinerijos proceso dalyviai

Reikalavimų inžinerijos procesas yra tarpdisciplininio pobūdžio procesas. Reikalavimus formuluojantis asmuo ar asmenys veikia kaip tarpininkai tarp verslo struktūrų, kurioms yra kuriama programų sistema, ir tą sistemą kuriančio inžinerinio personalo. Jiems tenka derėtis ir su vienais, ir su kitais. Be to, programų sistema suinteresuoti yra ne tik užsakovai, bet ir daugelis kitų subjektų. Tokiais subjektais gali būti ne tik paskiri asmenys ar jų grupės, bet ir įvairios organizacinės struktūros, įskaitant ištisas organizacijas. Skirtingų programų sistema suinteresuotų subjektų interesai dažnai skiriasi. Jų suformuluoti reikalavimai gali konfliktuoti ar netgi būti prieštaragingi. Todėl visiškai patenkinti visų programų sistema suinteresuotų šalių reikalavimus dažnai yra tiesiog neįmanoma. Tokiais atvejais tenka ieškoti tam tikro kompromiso. Toks kompromisas turi būti priimtinas visoms svarbiausioms sistema suinteresuotoms šalims. Kita vertus, kompromisas turi tenkinti finansinius, teisinius, politinius, techninius ir kitus projekto ribojimus. Kaip pabrėžia daugelis autorų [23]; [63]; [95]; [107]; [120], norint pasiekti tokį kompromisą, būtina nustatyti visas programų sistema suinteresuotas šalis, išsiaiškinti jų interesus ir perprasti jų formuluojamų reikalavimų esmę.

Programų sistema suinteresuotų šalių (subjektų) sąrašas priklauso nuo konkrečios sistemos pobūdžio, tačiau, kaip pažymi J. Goguen ir C. Linde [36], tame sąraše visuomet esti būsimieji programų sistemos vartotojai. Tai vienas iš svarbiausiuju programų sistema suinteresuotų šalių. Tačiau ši grupė taip pat nėra vienalytė. Kai kurie dalykinės srities specialistai patys naudojasi programų sistema, kiti naudojasi operatorių paslaugomis. Be to, netgi ir dalykinės srities specialistai versle užima skirtinges pozicijas, dirba skirtinguose kompiuterizuojamos organizacijos padaliniuose ir, savaime aišku, turi savus, neretai prieštaragingus interesus. Kitos svarbios suinteresuotosios šalys yra sistemos užsakovai (t. y. tie kas moka pinigus arba tie, kas atstovauja rinkai), marketingo specialistai, valstybinės valdžios ir valdymo institucijos, programų sistemų inžinieriai ir programų sistemų aptarnausiančios tarnybos (t. y. tie, kas ją administruos ir ekspluoatos). Marketingo specialistai ypač svarbūs tampa tuomet, kuomet nėra konkretaus užsakovo. Iš tiesų, galima sakyti, kad jie veikia kaip užsakovai. Valstybinės valdžios ir valdymo institucijos reguliuoja bankų, draudimo bendrovų ir daugelio kitų organizacijų darbą. Taigi, programų sistema turi tenkinti šių institucijų nustatytus reikalavimui, o neretai ir keistis duomenimis su tą institucijų ekspluoojamomis programų sistemomis. Programų sistemų inžinieriai visų pirma yra suinteresuoti minimizuoti ir supaprastinti savo darbą, kuo daugiau panaudoti iš ankstesnių projektų sukaupta patirtimi ir artefaktais (projektine dokumentacija, komponentais ir kt.). Programų sistemų aptarnaujančios tarnybos nori, kad sistemą būtų kuo paprasčiau administruoti ir kad ji automatizuotų kuo daugiau jai tvarkyti reikalingų procedūrų. Nei viena iš šių šalių taipogi nėra vienalytė ir kiekvienos iš jų pateiktai reikalavimai gali būti prieštaragingi ir konfliktuoti su kitų grupių pateiktais reikalavimais.. Taigi, kaip matome, kompromisų paieška čia toli gražu nėra paprasta. Yra žinomi atvejai, kuomet brangūs ir labai

svarbūs projektai žlugo vien todėl, kad tokio kompromiso taip ir nepavyko rasti, nors jo buvo ieškoma net keletą metų. Pavyzdžiui, 1994 metais JAV Federalinė aviacijos administracija nutraukė dešimt metų tėstą oro erdvės kontrolės sistemos tobulinimo projektą, nes sistema suinteresuotosios šalys per tą dešimtmetį taip ir nesugebėjo rasti kompromiso ir susitarti dėl sistemos reikalavimų [3]. Buvo parašyta kelių šimtų puslapių reikalavimų specifikacija, tam išleista daugiau kaip milijonas JAV dolerių, bet visa ta specifikacija buvo bevertė, nes Vašingtone reziduojanti administracija ir regioniniai oro erdvės kontrolės centralai taip ir nesusitarė, ar sistema turi būti centralizuota, ar išskirstyta.

Taigi, kuriant šiuolaikines programų sistemas, ypač formuluojant jų reikalavimus, susiduria daugelio skirtingu šalių interesai. Sąrašas apima:

- užsakovus, finansuojančius projektą arba perkančius gatavą sistemą savo organizacijos verslo poreikiams tenkinti;
- naudotojus, tiesiogiai ar netiesiogiai dirbančius su sistema (vartotojai traktuojami kaip užsakovų klasės poklasis);
- analitikus, formuluojančius reikalavimus ir perteikiančius juos sistemą kuriančiam inžineriniam personalui;
- produktą projektuojančius, kuriančius ir prižiūrinčius inžinierius;
- testuotojus, tikrinančius, ar sukurtas toks produktas, kokio buvo reikalauta;
- informacijos inžinierius, rašančius informacinės pagalbos tekstus, vartotojo vadovus ir jiems mokyti skirtą mokomąją medžiagą;
- projekto vadovybę, planuojančią projektą ir sekančią jo vykdymo eiga;
- teisininkus, atsakingus už produkto suderinamumą su galiojančių teisės aktų reikalavimais;
- gamybininkus, gaminančius produktą, į kurį yra įmontuojama kuriama programų sistema;
- pardavimo, marketingo, konsultavimo ir kitų su produkту ir užsakovais dirbančių tarnybų darbuotojus.

3.6 Reikalavimų inžinerijos proceso palaikymas ir vadyba

Reikalavimų inžinerijos procesas, kaip, beje, ir bet kuris kitas programų sistemų inžinerijos procesas, bus nesėkmingesnis, jei jis nebus aprūpintas reikiamais resursais ir nebus reikiamu būdu atliekama jo vadyba. Kaip pabrėžia Suzanne ir James Robertson [95], Ian Sommerville ir Pere Sawyer [107] bei Ralf Young [120], net ir geriausias reikalavimų inžinerijos proceso modelis neduos jokios naudos, jei:

- žmonės nebus apmokyti, kaip dirbti pagal proceso reikalavimus;
- modeliu numatytos veiklos nebus kompiuterizuotos, t. y. palaikomas atitinkamų instrumentų;
- nebus numatyta lėšos procesui diegti, palaikyti ir tobulinti.

Vienas iš svarbiausių proceso palaikymo elementų yra reikalavimų tvarkymas. Kaip jau minėjome, ši veikla persmelkia visas kitas bet kurio RI proceso modelio veiklas. Todėl ji turi būti harmonizuota su visomis kitomis atliekamomis veiklomis ir tas harmonizavimas kiekviename proceso modelyje atliekamas kitaip, atsižvelgiant į

konkrečius to modelio ypatumus. Tačiau, nepriklausomai nuo to, koks konkretus proceso modelis yra naudojamas, tvarkant reikalavimus visuomet turi būti atliekamas reikalavimų trasavimas ir turi būti vykdoma nuolatinė reikalavimų pokyčių kontrolė.

Kiekvienam reikalavimui turi būti sukonstruotos tokios trasos, kuriomis, judant atgal, būtų galima pasiekti to reikalavimo pirminius šaltinus, ir, judant pirmyn, – projektavimo sprendimus, nustatančius, kaip tą reikalavimą reikia įgyvendinti, tuos sprendimus įgyvendinantį kodą ir to kodo teisingumui tikrinti skirtus testus. Kaip konstruoti tokias trasas ir kaip su jomis dirbt, išsamiai aptarsime 8 skyriuje. Ten pat aptarsime ir pokyčių kontrolės problemas bei jų sprendimo būdus. Reikalavimų tvarkymo detales aptarsime 8 poskyryje.

Svarbu yra suderinti reikalavimų tvarkymą ne tik su visomis naudojamo reikalavimų inžinerijos proceso veiklomis, bet ir su projektavimu bei kitomis programų sistemos kūrimo proceso veiklomis, išorinėmis reikalavimų inžinerijos proceso atžvilgiu. Kadangi pradiniai programų sistemos reikalavimai beveik visuomet yra formuluojami lietuvių, anglų ar kokia nors kita natūraliajā kalba, o, kuriant programų sistemą, jų pagrindu kuriami įvairūs daugiau ar mažiau formalizuoti modeliai, rašomas programų kodas, kuriami testai bei kiti inžineriniai artifaktai, tai, pereinant nuo natūraliosios kalbos prie formaliųjų ar formalizuotų kalbų, reikalavimus yra labai nesunku iškreipti. Kitaip tariant, nepaisant to fakto, jog reikalavimai buvo suformuluoti remiantis verslo vizija ir detalizuojami 2 skyriuje aptartu būdu, pereinant nuo reikalavimų inžinerijos prie projektavimo ir kitų programų sistemų inžinerijos veiklų vėl iškyla realus pavojus atitrūkti nuo verslo siekiams tikslų ir tikrujų verslo poreikių. Be abejo, šis atotrūkis anksčiau ar vėliau išaiškės, tačiau gali būti prarasti lėšos ir laikas, projektas gali užsitęsti, gali būti sulaukta užsakovo nepasitenkinimo ir sugadinta programų sistemą kuriančios bendrovės reputacija. Todėl natūralu, kad dėl to, kaip išvengti tokio atotrūkio rūpinasi ne tik analitikai, projektuotojai, programuotojai bei kitas techninis personalas, bet ir vidutinės ir netgi aukščiausios grandžių vadovai. Juolab, jog greta techninio pobūdžio problemų, tokį kaip, pavyzdžiui, reikalavimų neišsamumas ar dviprasmiškumas, atotrūkis gali atsirasti ir dėl prastos vadybos. Svarbiausios vadybinės problemas čia yra tokios:

- Vadovybė turi užtikrinti reikalavimus formuluojančios bei specifikuojančios grupės (t. y. analitikų) ir inžinerinio personalo greitą ir patikimą bendravimą. Iškilus bet kokiems neaiškumams, projektuotojai, programuotojai, testuotojai bei naudotojo dokumentaciją rašantys specialistai turi turėti galimybes skubiai tuos neaiškumus išsiaiškinti ir būti tikri, kad gauti išaiškinimai yra patikimi ir galutiniai. Taigi, formuojant analitikų grupę turi būti nustatyta, kas kokiais klausimais konsultuos inžinerinį personalą ir numatyta atsakomybė, už ne laiku suteiktas ar nekokybiskas konsultacijas. Kadangi analitikai kartais ir patys gali nežinoti atsakymo, tai turi būti susitarta su užsakovu ar, kuriant parduoti rinkoje skirtas sistemas, su atitinkamais konsultantais dėl skubių konsultacijų gavimo būdų.
- Vadovybė turi suprasti, kad reikalavimų keitimas projekto eigoje yra neišvengiamas ir numatyti reikalavimų keitimo bei pokyčių pasekmių vertinimo procedūras. Tik tokiu būdu galima užtikrinti projekto stabilumą arba, vaizdžiai tariant, būti įsitikinusiem, jog reikalavimų kaitaliojimas „neišmuš“ projekto iš vėžių“. Vadovybė turi turėti galimybes greitai ir patikimai įvertinti, kiek laiko ir lėšų prireiks vienam ar kitam reikalavimo pokyčiui įgyvendinti. Tam visų pirma reikia turėti išsamias ir patikimas reikalavimų trasas, kas yra įmanoma

tik tuomet, kuomet projekte yra numatytos trasų konstravimo bei palaikymo veiklos ir tam yra skirti visi reikiami ištekliai.

- Vadovybė turi iš anksto apgalvoti kaip dažnai ir kokia forma jai turi būti teikiama informacija apie projekto eigą. Kitaip jai gali tekti gaišti per daug laiko tam, kad įsitikinti, jog vienas ar kitas reikalavimas jau yra įgyvendintas ir tai padaryta tinkamu būdu, jog to reikalavimo įgyvendinamumas patikrintas ir jog tikrai galima pasitikėti testavimo rezultatais. Informacija apie projekto eigą turi būti taip pateikta, kad ji būtų lengvai aprēpiama ir suvokama ne tik specialistams, bet ir užsakovo atstovams bei kitoms suinteresuotosioms šalims.
- Vadovybė turi numatyti būdus, kaip visus projekto dalyvius supažindinti su visų lygmenų atsakymais į klausimą „Kodėl?“. Jei inžinerinis personalas nesuvokia, kaip vienas ar kitas reikalavimas siejasi su aukštesniųjų lygmenų reikalavimais, įskaitant verslo tikslus, sistemos funkcionalumo atotrūkio nuo realių verslo poreikių išvengti yra labai sunku.
- Dideliuose projektuose, kuriuose lygiagrečiai dirba kelios vykdytojų grupės, kuriančios, tarkime, skirtingus kuriamos sistemos posistemius, vadovybė turi apgalvoti ir sukurti tų grupių sinchronizavimo mechanizmus. Jei kokio nors reikalavimo nepavyksta lokalizuoti konkrečiame posistemyje, jį realizuodamos vykdytojų grupės turi glaudžiai bendradarbiauti. Tačiau posistemių sudėtingumas dažnai esti skirtingas ir jie yra kuriami skirtingu greičiu. Todėl nuo to laiko, kai reikalavimas yra įgyvendinamas viename posistemyje, iki to laiko, kuomet su tuo reikalavimu pradedama dirbtiniame posistemyje, gali praeiti savaitės ar netgi mėnesiai. Todėl grupių sinchronizavimo mechanizmai čia yra labai svarbūs, netgi kritiniai.

Tačiau grįžkime prie reikalavimų inžinerijos proceso palaikymo ir vadybos problemų. Svarbiausia čia yra du dalykai – ką, išskyrus instrumentines priemones, reikia parengti reikalavimų inžinerijos procesui palaikyti ir kaip įvertinti ir eliminuoti reikalavimų inžinerijos rizikos veiksnius.

Reikalavimų inžinerijos proceso palaikymas turi apimti visas veiklas, įskaitant reikalavimų aiškinimąsi, analizę, specifikavimą, vertinimą ir tvarkymą. Dirbant pagal bet kurį proceso modelį, turi būti parengti:

- bendrujų proceso principų, kuriais reikia vadovautis, atliekant bet kurią proceso numatytyų veiklą, aprašas;
- proceso procedūrų aprašas (detalios instrukcijos kaip vykdyti kiekvieną iš proceso numatytyų užduočių);
- proceso vykdymo kontrolės veiksmų aprašas (kontroliniai taškai, kas kokius patikrinimus atlieka ir kt.);
- reikalavimo aprašo, reikalavimų specifikacijos ir kitų rengiamų dokumentų šablonai;
- reikalavimų duomenų bazės struktūra;
- klausimynai, kuriais turi būti naudojamas atliekant reikalavimų specifikacijos bei kitų rengiamų dokumentų peržiūras ar inspektavimą;
- klausimynas, kurio turi būti naudojamas vertinant reikalavimų pokyčių pasekmes;
- nurodymai, kaip turi būti aprašomos reikalavimų trasos (pavyzdžiui, reikalavimų trasavimo matricos formato aprašas);

- kontrolinis sąrašas, kuriame išrašoma kokie dokumentai, maketai, prototipai ir galbūt kokie nors kiti artifaktai turi būti sukurti atliekant reikalavimų inžinerijos darbus ir kokie įvykiai (peržiūros, dokumentų aprobavimas ir kt.) turi įvykti tų darbų eigoje;
- planas, kaip įgyvendinti kontroliniame sąraše numatytais tikslus).

Be to, turi būti sudarytas reikalavimų pokyčių aprobavimo bei jų pasekmių vertinimo komitetas (PAK).

Bet kuris reikalavimų inžinerijos procesas paprastai numato tokias procedūras:

- reikalavimų aiškinimosi ir specifikavimo procedūras (kaip nustatyti suinteresuotąsias šalis ir reikalavimų šaltinius, kaip aiškintis reikalavimus, kokią tarnybinę informaciją pateikti su kiekvienu reikalavimu, kaip analizuoti ir vertinti reikalavimus);
- reikalavimų lokalizavimo procedūras;
- reikalavimų proretizavimo procedūras;
- peržiūros ir inspektavimo procedūras;
- reikalavimų pokyčių valdymo procedūras;
- reikalavimo statuso sekimo procedūras (kokias būsenas gali turėti reikalavimas, kada jos keičiasi, kaip tai atspindima atskaitose ir pan.);
- PAK darbo procedūras.

Reikalavimų inžinerija baigiasi nesėkme, jei:

- gaunami prastos kokybės reikalavimai;
- reikalavimų inžinerijos darbams atliliki buvo išleista per daug lėšų;
- reikalavimų inžinerijos darbai užsiėsė per ilgai (jiems turėtų būti sugaišta ne daugiau nei 15-20 procentų viso projektui skirto laiko);
- reikalavimų inžinerijos darbams sunaudota per daug kitų ištaklių;
- užsakovas liko nepatenkintas gautais rezultatai.

Svarbiausieji rizikos veiksniai yra šie:

- skirtingos suinteresuotosios šalys turi skirtingas kuriamos sistemos vizijas;
- analitikai neteisingai suvokė užsakovo poreikius;
- nepavyko dalykinės srities specialistus įtraukti į reikalavimo aiškinimosi ir vertinimo procesus reikiamu laipsniu;
- nepavyko apibrėžti projekto ribų arba jos nuolat kito;
- nuolat kito per didelis reikalavimų skaičius;
- dėl kultūrių skirtumų ar kitų priežasčių analitikams nepavyko susikalbėti su dalykinės srities specialistais ar kitomis projektu suinteresuotomis šalimis;
- pasikeitus vykdytojų organizacijos vadovybei, buvo pradėtas keisti reikalavimų inžinerijos procesas;
- darbų eigoje atsirado naujos projektu suinteresuotos šalys, pavyzdžiu, papildomi projekto finansavimo šaltiniai;
- nepakankama vykdytojų grupės sutelktis;
- vykdytojų kaita;
- darbo su subvykdytojais nesėkmės;
- techniniai nesklandumai (nepavyksta laiku įdiegti ar įsisavinti procese naudojamų instrumentų ar procedūrų);
- žinių apie dalykinę sritį trūkumas.

Visus šiuos rizikos veiksnius būtina išanalizuoti. Reikia pasistengti įvertinti jų tikimybes ir pasekmes, kurias jie gali sukelti. Reikia taip pat numatyti, kokių veiksmų bus imtasi, pasireiškus vienam ar kitam rizikos veiksniui. Aišku, yra neįmanoma iš anksto įvertinti visų rizikos veiksnį tikimybes ir juo labiau apsaugoti nuo visų galimų neigiamų pasekmių. Tačiau gerai parengtas rizikos valdymo planas vis vien gerokai padidina reikalavimų inžinerijos darbų sėkmės tikimybę. Planą reikia nuolat peržiūrėti, atnaujinti ir, savaime aišku, juo reikia naudotis valdant projektą.

3.7 Reikalavimų inžinerijos proceso kokybė

Programų sistemos kūrimo proceso kokybė gali būti vertinama pagal daugelį parametrų. Tačiau svarbiausia yra tai, kiek kainavo sukurti programų sistemą, kiek laiko tai truko ir kokių mastu programų sistema suinteresuotos šalys, visų pirma užsakovas, liko ja patenkintos. Aišku, visa tai priklauso ne tik nuo reikalavimų inžinerijos proceso, bet ir nuo kitų programų sistemų inžinerijos procesų, naudotų kuriant sistemą. Tačiau reikalavimų inžinerijos proceso vaidmuo čia yra esminis [107]. Pasirinkus tinkamą reikalavimų inžinerijos procesą, galima ženkliai sumažinti viso projekto kainą ir pagerinti kuriamos programų sistemos kokybę. Pasirinkus blogą procesą, projektas apskritai gali baigtis nesėkmę. Taip gali įvykti dėl to, kad naudojant blogą procesą, nėra jokių garantijų, jog į galutinę reikalavimų specifikaciją įtraukti reikalavimai tikrai tenkins kompiuterizuojamo verslo poreikius, bus tinkamai dokumentuoti ir įgyvendinami, arba kad, netgi ir parengus kokybišką specifikaciją, ji pamažu neišsigims, keičiant reikalavimus projekto eigoje. Taigi, svarbu suvokti, kas tai yra reikalavimų inžinerijos proceso kokybė, ir išmokti tokius procesus tobulinti [63]; [107]; [120]. Be kita ko, yra svarbu, kad programų sistemų ir programų inžinerijos proceso kokybės standartai apimtų taip pat ir reikalavimų inžinerijos procesus.

Apskritai, bet kuris reikalavimų inžinerijos procesas yra labai sudėtingas reiškinys ir suvokti, kame slypi jo trūkumai ir kaip jie pasireiškia, o juolab išmokti matuoti tokį procesų kokybę bei vertinti jų tinkamumą konkretiems procesams toli gražu nėra paprasta. Todėl paprastai reikalavimų inžinerijos proceso kokybė yra vertinama vertinant to proceso generuoojamas išeigos (reikalavimų specifikacijos, sukurtų maketų ir kt.) kokybę, t. y., vertinant kokių mastu reikalavimų inžinerijos proceso sukurta išeiga tenkina kitų programų sistemai kurti naudojamą procesą, projektavimo, programavimo, testavimo, diegimo ir tolimesnio sistemos tobulinimo, poreikius ir gali būti panaudota tų procesų sėkmei užtikrinti. Reikia pažymėti, kad nors toks reikalavimų inžinerijos proceso kokybės vertinimo būdas yra gana paplitęs, jis yra per daug supaprastintas, nes, kaip pamatysime šiek tiek vėliau, reikia atsižvelgti ir į kitas proceso savybes, pavyzdžiui, jo naudojimo paprastumą.

Įvertinti reikalavimų inžinerijos proceso tinkamumą konkrečiam projektui yra dar sudėtingiau. Bene išsamiausiai šį klausimą yra išnagrinėjęs žymus programų sistemų inžinerijos specialistas Donald Firesmith [32]. Aptarsime reikalavimų inžinerijos procesų tinkamumo ir kokybės aspektus remdamiesi būtent šiame darbe paskelbtomis idėjomis.

Procesas gali būti specifinis, tinkamas tik kokio nors vieno tipo projektams, arba pritaikomas skirtiniems projektų tipams. Juo universalesnis yra procesas, tuo jis yra sudėtingesnis. Kitaip tariant, universalesni procesai turi daugiau komponentų, o ir patys tie komponentai yra universalizuoti, apibendrinti ir, pritaikant tuos komponentus konkrečiam projekto tipui, juos reikia „perkirpti“ (angl. *tailore*). Todėl, renkantis

universalius procesus, labai svarbu išsiaiškinti, kokių komponentų „perkirpimo“ mechanizmus jie turi ir kiek paprasta tais mechanizmais pasinaudoti.

Renkantis konkretiam projektui geriausiai tinkanti reikalavimų inžinerijos procesą, reikia atsižvelgti į šiuos projekto ypatumus:

- projekto dydį ir sudėtingumą;
- naudojamą sutarties modelį;
- standartus, pagal kuriuos yra dirbama projekte;
- naudojamą kokybės valdymo sistemą;
- geografinį projekto išskirstymą;
- kuriamos programų sistemos svarbą jos palaikomam verslui;
- reikalavimų kintamumą;
- programų sistema suinteresuotų šalių aktyvaus dalyvavimo projekte laipsni;
- pageidaujamą pasirenkamo proceso pobūdį;
- korporacinę kultūrą;
- instrumentinį projekto palaikymą;
- reikalavimų kokybės vertinimo būdus;
- projekto biudžetą.

Projekto dydį ir sudėtingumą galima vertinti pagal jam skirto biudžeto dydį, tame dalyvaujančių žmonių skaičių ir planuojamą projekto trukmę. Juo didesnis ir sudėtingesnis yra projektas, taip pat tuo didesnė ir sudėtingesnė yra kuriamoji programų sistema. Todėl dideliems ir sudėtingiems projektams reikia labiau formalizuotų ir sudėtingesnių reikalavimų inžinerijos procesų. Mažiems ir paprastiems projektams paprastai pakanka paprastų ir mažai formalizuotų procesų.

Jei projekte dalyvauja subvykdymo arba jeigu programų sistemą partnerių teisėmis kuria kelios organizacijos, iškyla darbų pasidalijimo problema ir juridinės atsakomybės klausimai. Todėl tokiuose projektuose reikalavimai turi būti labiau formalizuoti ir geriau parengti bei tvarkomi. Reiškia, tokiuose projektuose taip pat reikia labiau formalizuoto ir sudėtingesnio reikalavimų inžinerijos proceso. Panašiai yra ir tais atvejais, kuomet reikalavimų specifikacija yra laikoma neatskiriama sutarties dalimi arba, kitaip tariant, kuomet ji įgyja formalų juridinį statusą ir užsakovas priima programų sistemą tikrindamas, ar visi specifikacijoje surašyti reikalavimai tikrai yra įgyvendinti.

Jei yra reikalaujama, kad projekte būtų vadovaujamas kokiais nors tarptautiniai arba nacionaliniai sistemos kūrimo būdo, sistemos saugos, sistemos patikimumo ar kitais standartais pavyzdžiui, ISO/IEC 12207 [ST15], IEEE STD 830-1998 [ST3], FIPS PUB 140-2 [ST33] ar MIL-STD-882D [ST34], tai gali prireikti, kad reikalavimams saugoti skirtame repozitoriuje su reikalavimais būtų susieti tam tikri atributai ir kad tų atributų reikšmės tenkintų tam tikus reikalavimus. Panašiai yra ir tais atvejais, kuomet vykdytojo organizacijoje yra įdiegta kokia nors kokybės valdymo sistema, pavyzdžiui, kuomet ta organizacija dirba pagal SEI CMM [87] ar

ISO Spice [ST18] reikalavimus. Reikalaujama repozitorijaus struktūra riboja reikalavimų inžinerijos proceso parinkimą.

Jei projekto dalyviai yra išsibarstę po skirtinges vietas ir tarpusavyje daugiausia bendrauja raštu, t. y. keisdamiesi dokumentais bei žinutėmis, tai kuriamos programų sistemos reikalavimai turi būti gerai dokumentuoti ir bent iš dalies formalizuoti, nes kitaip nepavyks išvengti dviprasmybių ir kitokių nesusipratimų. Pageidautina, kad šiuo atveju būtų pasirinktas tokis reikalavimų inžinerijos procesas, kuriamė, be kita ko, būtų numatytais repozitorijus, prieinamas visiems geografiškai išsibarsčiusiems projekto dalyviams, išskaitant užsakovus ir kitas kuriamas programų sistema suinteresuotas šalis.

Reikalavimų inžinerijos proceso pasirinkimas priklauso ir nuo kuriamos programų sistemos kritišumo laipsnio arba, kitaip tariant, nuo to, kokias pasekmes gali sukelti sukurtos programų sistemos potencialios trikty. Jei kuriamoji programų sistema turės valdyti, tarkime, branduolinį reaktorių ar keleivinių lėktuvų skrydžius, jos trikty gali baigtis daugelio žmonių žūtimi. Rimtų problemų, nors ir nesibaigiančių kieno nors žūtimi, gali kilti neteisingai veikiant, tarkime, bankų, mokesčių inspekcijų muitinių ar kokių nors kitų svarbių institucijų veiklą palaikančiomis programų sistemoms. Todėl, tuo kritiškesnė yra kuriamoji programų sistema, tuo kruopščiau ir išsamiau turi būti formuluojami tos sistemos reikalavimai ir tuo labiau formalizuoti turi būti tie reikalavimai. Dažnai gali prireikti netgi reikalavimų maketavimo ir jų tikrinimo panaudojant išmetamus maketus <Trm19>. Taigi, projektams, kuriuose yra kuriamos kritinės sistemos, yra reikalingi sudetingi ir maksimaliai formalizuoti reikalavimų inžinerijos procesai.

Reikalavimų inžinerijos procesas prilauso ir nuo to, kaip dažnai projekto eigoje yra keičiami reikalavimai. Juo dažniau yra kaitaliojami reikalavimai, tuo svarbiau yra, kad procesas numatyta priemones greitai ir paprastai tuos reikalavimus keisti ar papildyti. Tokias galimybes turintys procesai yra vadinami agiliaisiais (angl. *agile*). Beje, klaudinga būtų manyti, nors tokia nuomonė ir yra gana plačiai paplitusi, jog agilieji procesai visuomet yra beveik neformalizuoti ir paviršutiniški.

Reikalavimų inžinerijos procesas priklauso ir nuo to, kiek yra šalių, suinteresuotų kuriamas programų sistema, ir kaip aktyviai tos šalys dalyvauja aiškinantis, formuluojant, analizuojant ir vertinant reikalavimus. Savaime aišku procesas turi būti toks, kad jis būtų paprastas ir suprantamas visoms šalims, aktyviai dalyvaujančioms reikalavimų inžinerijos veiklose. Užsakovo atstovai, būsimieji sistemos naudotojai ir kitos sistema suinteresuotos šalis turi suprasti, koks vaidmuo kiekvienai iš jų numatytais procese, ko iš jų tikimasi, ir susidoroti su joms pavestomis užduotimis. Visoms suinteresuotoms šalims turi būti suprantami bei aiškūs taip pat ir visi rengiami dokumentai, kuriami maketai ir visa kita, kas yra sukuriama vykdant procesu numatytais reikalavimų inžinerijos veiklas.

Dar vienas svarbus veiksnys, į kurį būtina atsižvelgti renkantis reikalavimų inžinerijos procesą, yra jo autonomiškumo laipsnis. Tai priklauso nuo, kaip yra organizuotas darbas vykdytojų organizacijoje, t. y. nuo to, ar ten yra įdiegtas koks nors visa apimantis procesas, reglamentuojantis visas vykdomas veiklas, išskaitant projekto vadybą, sistemos kūrimą, sukurtos sistemos diegimą, aptarnavimą bei priežiūrą ir, pagaliau, jos demontavimą, keičiant ją nauja sistema, ar yra reglamentuojamos tik reikalavimų inžinerijos ir tiesiogiai su ja siejamos kitos programų sistemų inžinerijos veiklos. Aišku, jei reikalavimų inžinerijos procesas yra įkomponuotas į kokį nors visa apimanti procesą, jis negali turėti atskiro repozitorijaus

ir privalo naudotis bendru su kitais procesais repozitorijumi. Taigi, ir reikalavimų inžinerijos procedūros bei rengiami dokumentai taip pat turi tenkinti tam tikrus reikalavimus. Jei organizacijoje joks universalus procesas nėra įdiegtas, tai reikalavimų inžinerijos procesas gali naudotis vidiniu repozitorijumi ir pakanka to, kad būtų aiškiai apibrėžti to proceso sąveikos su kitais procesais interfeisai. Universalaus proceso naudojimas užtikrina geresnę skirtingų procesų sąveiką ir sumažina darbą, kurio prireikia dubliuojant tą pačią informaciją skirtingų procesų repozitorijuose arba kitaip organizuotose informacijos saugyklose. Kitą vertus, universalų procesų palaikymas yra sudėtingas, tuo turi užsiiminėti specialiai tam išskirti žmonės ir todėl jis yra pakankamai brangus. Todėl tokią prabangą sau leisti paprastai gali tik pakankamai didelės organizacijos.

Reikalavimų inžinerijos procesas negali konfliktuoti su vadinamąjį organizacijos korporacine kultūra, t. y. su tuo, kaip yra įpratę dirbtį ten dirbantys žmonės, koks yra jų mentalitetas, išsilavinimas ir profesinė kvalifikacija. Darbuotojai turi būti pasirengę priimti procesą, suprasti, kokią naudą jis duoda, o ne traktuoti jį kaip kažką primestą iš viršaus ir tiktai kliudantį dirbtį. Aišku, diegiant reikalavimų inžinerijos procesą, beveik visuomet tenka keisti organizacijos korporacinę kultūrą. Tam rengiami teoriniai ir praktiniai mokymai, darbuotojams tenka pakartotinai išklausyti kai kuriuos universitetinius kursus, sertifikuotis, dalį darbuotojų, visų pirma, nusipelniusius „praktikus“, tenka pakeisti naujais darbuotojais. Tai skausmingi, tačiau neišvengiami pertvarkymai. Kita vertus, būtina gerai suprasti, kad ne bet kokius pertvarkymus galima padaryti. Beveik garantuotai nepavyks įdiegti proceso, pritaikyto, tarkime, Japonijos arba JAV sąlygomis, sukurto, atsižvelgiant į tų šalių mentaliteto ypatumus. Nesėkmė greičiausiai ištiks ir tuomet, kuomet jokio proceso neturėjusi organizacija bandys diegti sudėtingą ir labai formalizuotą procesą.

Jei projekte naudojami kokie nors programų kodo generatoriai ar kitos priemonės, galbūt netgi kokia nors CASE sistema <Trm28>, tai reikalavimų inžinerijos procesas, visų pirma to proceso instrumentinis palaikymas, turi būti suderintas su jau naudojamomis programinėmis priemonėmis. Pavyzdžiui, vargu ar būtų protinga diegti procesą, dirbantį, tarkime, su RequisitPro repozitorijumi, jei programų sistemos yra projektuojamos, tarkime, su MagicDraw sistema.

Vertinant reikalavimų kokybę yra naudojami vieni ar kiti kokybės matai, pavyzdžiui, gali būti matuojami reikalavimų išsamumas, neprieštaragingumas ar kintamumas. Kokios reikalavimų savybės turi būti vertinamos, iš dalies priklauso nuo konkretaus projekto. Kita vertus pasirinkta reikalavimų kokybės matų sistema gali riboti reikalavimų inžinerijos proceso parinkimą, nes procesas privalo sukurti tokius duomenis, kad tuos matus būtų galima panaudoti.

Pagaliau, reikalavimų inžinerijos proceso parinkimas priklauso ir nuo to, kiek pinigų galima skirti to proceso komponentams įsigyti arba sukurti, įdiegti bei palaikyti. Kitaip tariant, tai priklauso nuo projekto biudžeto. Procesas priklauso ir nuo projekto vykdymo terminų. Jei trūksta pinigų ir laiko procesui įdiegti, tenka rinktis paprastesnį ir galbūt prastesnį procesą, negu būtų tikslinga, atsižvelgiant į kitus projekto ypatumus.

Taigi, kalbant apie reikalavimų inžinerijos proceso kokybę, būtina aiškiai atskirti du skirtingus dalykus – pačio proceso kokybę ir jo tinkamumą konkrečiam projektui arba, kaip įprasta sakyti, jo panaudojimo kokybę. Proceso kokybė apibūdina tai, kaip gerai procesas yra padarytas ir parengtas diegti. Kitaip tariant, ji priklauso nuo to, kokių mastų procesas yra baigtas kurti, kiek išsamiai jis yra dokumentuotas,

išbandytas, ar teikiamos mokymo paslaugos, veikia sertifikavimo centrai, galima gauti reikiamas konsultacijas. Nuo ko priklauso projekto tinkamumas, jau aptarėme aukščiau. Nepaisant daugelio mokslinėje spaudoje paskelbtų publikacijų, kuriose aptariami vieni ar kiti proceso kokybės ar jo tinkamumo atributai, ir viena ir kita kol kas tebėra daugiau vertinamojo pobūdžio. Kokio nors visuotinai pripažinto atributų rinkinio kol kas sudaryta nėra. Baigdam, dar kartą trumpai aptarsime kai kuriuos veiksnius, į kuriuos reikia atsižvelgti renkantis reikalavimų inžinerijos procesą ir kurių svarba aukščiau galbūt buvo nepakankamai pabrėžta. Visų pirma reikia įvertinti proceso įdiegiamumą ir įsisavinamumą. Jis priklauso nuo laiko, reikalingo paruošti procesą diegti (parengti diegimo planą, atliskti organizacinius pertvarkymus, organizuoti darbuotojų mokymą ir kt.), tam reikalingą darbo sąnaudą, darbuotojams apmokyti reikalingo laiko ir laiko, reikalingo procesui palaikyti skirtoms instrumentinėms priemonėms įdiegti ir reikalavimų bazei (repozitorijui) sukurti. Toliau reikia panagrinėti proceso funkcionalumą arba, kitaip tariant, kokias reikalavimų inžinerijos veiklas ir kokių mastu procesas palaiko (proceso funkcinis išsamumas) ir kiek lengvai galima susieti procesą su kitais programų sistemų inžinerijos procesais (proceso interoperabilumas). Toliau reikia nagrinėti proceso adaptuojamumą arba, kitaip tariant, kiek lengvai galima procesą „perkirpti“ ir pritaikyti naujiems projektams. Labai svarbūs yra ir proceso efektyvumas ir patikimumas. Kitaip tariant, reikia panagrinėti, kiek laiko reikės sugaišti naudojant proceso numatytas procedūras ir dokumentuojant tą procedūrų rezultatus ir kokios yra garantijos, kad, panaudojus visas tas procedūras bus gauti tikrai kokybiški reikalavimai. Aišku, kaip ir visais kitais atvejais, šie dalykai yra prieštaringi. Juo procesas yra efektyvesnis ir juo mažiau laiko prireikia reikalavimams atskleisti, suformuluoti, specifikuoti, išanalizuoti bei įvertinti, tuo mažesnės garantijos, jog tie reikalavimai tikrai bus kokybiški. Ir atvirkščiai, juo didesnės yra garantijos, kad reikalavimai bus kokybiški, tuo daugiau laiko reikia jiems sugaišti. Dar vienas labai svarbus aspektas yra proceso naudojimo paprastumas. Nereikia galvoti, jog sudėtingais procesais naudotis visuomet yra sudėtinga, o paprastais paprasta. Kaip ir programų sistemos atveju, tai daugiau priklauso nuo interfeisių arba, kitaip tariant, nuo to, kiek apgalvoti ir patogūs žmogui yra rengiami dokumentai bei naudojamos procedūros. Pagaliau, paskutinė svarbi proceso savybė yra jo vertinimo ir tobulinimo galimybės. Tačiau apie tai išsamiau pakalbėsime 3.9 poskyryje.

Reikia pabrėžti, kad reikalavimų inžinerijos proceso kokybė visuomet turi būti vertinama platesniame, ne tik reikalavimų inžinerijos veiklų kontekste. Pavyzdžiu, galima pasirinkti tokį reikalavimų inžinerijos procesą, kuris reikalauja palyginti nedidelių darbo sąnaudų, tačiau negarantuojat priimtinos reikalavimų specifikacijos kokybės. Tuo pačiu viso projekto kainą šis procesas gali padidinti kelis kartus ir todėl jokiu būdu negali būti laikomas kokybišku. Iš tiesų, reikalavimų inžinerijos proceso kokybei įvertinti nepakanka netgi ir viso projekto konteksto. Pavyzdžiu, pasirinktasis procesas gali atrodyti kokybiškas, nes jis gerai dera su kitais programų sistemų inžinerijos procesais ir kartu su jais užtikrina, kad visi suformuluoti reikalavimai bus įgyvendinti laiku, neviršijant projekto biudžeto ir kokybiškai. Tačiau, jeigu tas procesas nenumato priemonių, kaip užtikrinti reikalavimų atitikimą realiems verslo poreikiams, užsakovas jokios realios naudos iš sukurtos sistemos gali ir negauti. Jo pinigai paprasčiausiai bus išsvaistyti. Netgi ir toks kontekstas, apimantis projektą, produktą ir verslą, kurį tas produktas palaiko, kartais gali būti per siauras, nes pasirinktasis reikalavimų inžinerijos procesas gali negarantuoti, jog suformuluoti reikalavimai kaip nors nepažeis viešojo intereso. Tačiau tai jau išeina už mūsų kurso ribų.

3.8 Reikalavimų inžinerijos proceso branda

Kaip jau kalbėjome 3.7 poskyryje, organizacija yra nepajėgi iš karto pereiti nuo primityvių, amatininkiskų darbo metodų prie brandaus reikalavimų inžinerijos proceso naudojimo. Tam reikia laiko. Tačiau kiek gi šio proceso brandos lygmenų galima išskirti ir kaip juos galima apibūdinti? Aišku, jog procesai, kaip ir žmonės bresta laipsniškai ir apskritai susitarimai apie jų brandą yra gana sąlyginiai. Tačiau, kalbėdami apie žmonių brandą, mes įsivedame tam tikrą skalę ir jų brandą matuojamame metais. Toks matas yra gana sąlyginis, nes skirtingi to paties amžiaus žmonės esti subrendę gana skirtingai. Kitas žiūrėk taip ir nesubresta iki pat mirties ir iki senatvės pragyvena paaugliškame detektyvinių filmų, krepšinio varžybų ir alaus gérimo pasaulyje. Todėl psichologai asmenų brandą matuoja ne metais, o visai kitais metais. Žiūrima, kokį atsakomybės ir pareigos jausmą sugebėjo išsiugdyti asmuo, kokiui mastui jis iš tiesų suvokia jį supantį pasaulį, geba priimti racionalius sprendimus ir kokia vertybų sistema yra grindžiamas jo elgesys. Panašiai yra ir su procesais.



48 pav. Gebėjimų brandos modelio numatyti gebėjimų brandos lygiai.

Apie programų sistemų inžinerijos procesų brandą pirmą kartą prabilta apie 1988 m., kuomet JAV gynybos departamentas, patyres didžiulus nuostolius dėl žlugusių projektų, paprašė prie Carnegie Mellon universiteto veikiantį Programų sistemų inžinerijos institutą sukurti modelį, kuris prieš pradedant projektą leistų įvertinti vykdymo organizacijos gebėjimų lygį ir prognozuoti, ką ta organizacija yra pajėgi padaryti. Taip buvo sukurtas vadinamasis Gebėjimų brandos modelis [87]. Šis modelis apibrėžė penkis brandos lygius (48 pav.) ir charakterizavo juos per organizacijoje naudojamas programų sistemų kūrimo praktikas. Trumpai aptarsime šiu lygių ypatumus [88].

Pirmam lygiui priskiriamos organizacijos, kurios jokio rimtesnio produkto sukurti negali iš princiopo. Jokių formalizuotų procesų jose nėra ir visi dirba taip, kaip išmano. Antro lygio organizacijose yra įdiegtas reikalavimų inžinerijos procesas, vyksta reikalavimais grindžiamas projekto planavimas, kontroliuojama projekto eiga, užtikrinama reikiama kuriamos programų sistemos kokybė, vyksta konfigūracijos valdymas. Tokios organizacijos dažniausiai yra pajėgios laiku ir neviršydamos planuoto biudžeto kurti kokybiškas programų sistemas, tačiau nuo nesėkmės vis tik nėra apsaugotos. Be to, jų sukurtos programų sistemos gali būti gerokai per brangios. Taip yra todėl, kad procesai jose nėra formalizuoti (t. y. aprašyti raštu ir kiekvieno darbuotojo pareigybinię instrukciją yra įrašytos atitinkamos užduotys), tinkamai koordinuojami bei vertinami. Organizacijos, kuriose visa tai yra padaryta, priskiriamos trečiam gebėjimų brandos lygiui. Be to, tokiose organizacijose darbuotojai turi būti nuolat mokomi, jų kvalifikacija turi būti nuolat keliama, nes kitaip jų gebėjimai sens ir praras savo vertę. Trečio brandos lygio organizacijose taip pat turi būti integruoti vadybos ir inžineriniai procesai ir vykti vadinamoji produkto inžinerija, užtikrinanti, jog produktas bus sukurtas sunaudojus galimai mažiausiai ištaklių. Jose turi veikti peržiūros ir inspektavimo procedūros, užtikrinančios, kad

laiku bus pastebėti ir pašalinti visi nuokrypiai nuo projekto vykdymo plano ir kad iš karto, pradedant reikalavimų lygmeniu, bus išryškinti ir pašalinti visi kuriamos programų sistemos defektai. Ketvirtam gebėjimų brandos lygiui priskiriamos organizacijos, kuriose įdiegtą kokybės valdymo sistemą ir vyksta kiekybiniais rodikliais grindžiama procesų vadyba. Pagaliau, penktam gebėjimų brandos lygiui yra priskiriamos organizacijos, kurios geba ne tiktais laiku pašalinti kuriamos sistemos defektus, bet ir jų išvengti, valdyti technologinius pokyčius ir nuolat tobulinti savo naudojamus procesus.

Yra sukurta ir daugiau panašių modelių [13, 27, 67, 89, ST18]. Nepriklausomai nuo to, kaip yra vertinama organizacijų gebėjimų branda, skirtinti gebėjimai, taip pat ir reikalavimų inžinerijos gebėjimai, kartais toje pačioje organizacijoje esti skirtinges brandos. Todėl kyla klausimas, kaip įvertinti ne visų, o tiktais tam tikrų organizacijos gebėjimų, mūsų atveju, reikalavimų inžinerijos gebėjimų, brandą. Daugumoje brandos modelių tas klausimas apskritai nėra nagrinėjamas. Šiek tiek dėmesio jam skirta tik Gebėjimų brandos modelyje [88] ir ypač vadinamajame Gebėjimų brandos modelių integravimo projekte [1, 15]. Šis projektas pradėtas, siekiant sukurti metodinį karkasą, integruojantį tiek esamus, tiek ir būsimus brandos modelius. Modelis reikalauja, kad trečio brandos lygio organizacijose būtų įdiegtos reikalavimų tvarkymo procedūros ir kad jos būtų vykdomos lygiagrečiai su kitomis reikalavimų inžinerijos veiklomis. Be to, išreikštiniu būdu pareikalauta, kad reikalavimų tvarkymas apimtu ir reikalavimų trasavimą ir kad trasomis būtų galima judeti abiem kryptim, t. y. ir nuo reikalavimų link kodo bei testų, ir atvirkščiai, nuo testų ir kodo link reikalavimų, pasiekiant ne tik reikalavimų specifikaciją, bet ir pirminius reikalavimų šaltinius. Taip pat yra reikalaujama, kad trečio gebėjimų brandos organizacijose būtinai būtų formuluojami verslo lygmens reikalavimai ir būtų rengiama ne tik reikalavimų, bet ir poreikių specifikacija <Trm35>. Akcentuojamas reikalavimas, kad reikalavimų vertinimas būtų pradedamas kuo anksčiau, gal būt dar nebaigus formuluoti visų reikalavimų, ir jokiui būdu nebūtu paliktas bandymų ir testavimo stadijai. Tačiau ir šiame dokumente nėra pasakyta, kaip įvertinti reikalavimų inžinerijos proceso brandą. Jame taip pat nėra pasakyta, kokiu reikalavimų inžinerijos gyvavimo ciklo modeliu reikia vadovautis. Be to, terminai *reikalavimų aiškinimas* <Trm46>, *reikalavimų analizė* <Trm47>, *reikalavimų specifikavimas* <Trm57>, *reikalavimų vertinimas* <Trm62> ir *reikalavimų tvarkymas* <Trm61> tame yra traktuojami šiek tiek kitaip, negu jie paprastai yra traktuojami literatūroje reikalavimų inžinerijos klausimai ir yra apibrėžti IEEE ir ACM oficialiose rekomendacijose [111]. Todėl reikalavimų inžinerijos proceso brandos lygmenis aptarsime vadovaudamiesi kitų autorių darbais [73, 99, 100, 101, 102, 107, 108].

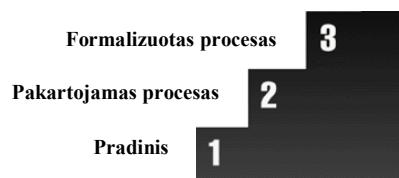
Mažiau brandus procesas	Labiau brandus procesas
<p>Reikalavimų aiškinimas vyksta telefonu su viena iš suinteresuotujų šalių.</p> <p>Visiems projektams ir dirbant su bet kuria iš suinteresuotujų šalių yra naudojamas tas pats reikalavimų aiškinimosi būdas.</p> <p>Pradiniai reikalavimai yra rašomi į repozitorijų ir ten saugomi, bet,</p>	<p>Reikalavimai, išsiaiškinus ir suderinus juos su daugeliu suinteresuotujų šalių, yra surašomi raštu. Reikalavimų aiškinimosi būdas parenkamas, atsižvelgiant į vykdomo projekto ir suinteresuotosios šalies, su kuria aiškinamasi reikalavimus, ypatumus.</p> <p>Viso projekto eigoje repozitorijuje saugomi reikalavimai atspindi vi-</p>

<p>keičiantis reikalavimais, jie ten nėra keičiami.</p> <p>Organizacijoje nėra įdiegtas reikalavimų pokyčių valdymo procesas, o, jei jis ir yra įdiegtas, juo naujojamas tik kartas nuo karto.</p> <p>Nėra jokio būdo išsitinkinti tuo, ar vieną arba kitas konkretus reikalavimas tikrai yra įgyvendintas sistemoje.</p>	<p>sus darytus reikalavimų pakeitimus.</p> <p>Organizacijoje įdiegtas ir reguliarai naudojamas reikalavimų pokyčių valdymo procesas.</p> <p>Yra sukurta ir naudojama reikalavimų trasavimo matrica.</p>
---	---

49 pav. Skirtingos brandos reikalavimų inžinerijos procesų palyginimas.

Pradékime nuo 49 pav. pateikto dviejų reikalavimų inžinerijos procesų brandos palyginimo [73]. Paveikslėlyje pateikti tik keli iš daugelio tų procesų skirtumų, tačiau ir toks palyginimas gana vaizdžiai parodo, kuo gi skiriasi mažiau ir labiau brandūs reikalavimų inžinerijos procesai. Tačiau klausimai, kaip gi reikėtų apibrėžti reikalavimų inžinerijos procesų lygius ir kiek tų lygių prasminga išskirti, kol kas vis dar lieka neatsakyti. Prieš pradēdami ieškoti atsakymų, dar pasvarstykime, ar apskritai tų atsakymų reikia. Kam gi reikia apibrėžti reikalavimų inžinerijos proceso lygius ir kodėl organizacijos turėtų siekti šio proceso didesnės brandos? I šį klausimą atsakyti nesunku. Beje, ši poskyri mes ir pradējome nuo tai parodančio pavyzdžio. Organizacijos siekia didesnės savo procesų brandos gryna dėl praktinių, ekonominių sumetimų. Tai daryti jas verčia gyvenimas. Naudojant žemo brandos lygio procesus, projektais, jeigu apskritai juos pavyksta baigti, tėsiasi per ilgai ir kainuoja per daug. Įdiegus brandesnį procesą ir protingai juo naudojantis, galima ženkliai padidinti darbo našumą, pagerinti kuriamų programų sistemų kokybę ir jas atpiginti. Kadangi nei viena organizacija nepajęgi iš karto pereiti nuo visiškai nebrandžių procesų prie brandžių, tai ir reikia suvokti, kokius inovacinius slenksčius yra įmanoma įveikti (žr. 48 pav.). Aišku, yra ir kitas kelias – ne pereidinėti nuo vieno brandos lygio prie kito, o nuolat tobulinti procesą. Gebėjimų brandos modelių integravimo projektas [1, 15] numato ir tokį procesų brandinimo būdą. Procesas šiuo atveju gali būti neskaidomas į kokias nors smulkesnes dalis, būti tolydus (angl. *continuous*). Tačiau ir šiuo atveju reikia sekti, kokiu greičiu yra judama į priekį ir planuoti tolimesnius veiksmus. Neturint apibrėžtų proceso brandos lygių, tą daryti būtų labai sunku.

Kaip išskirti reikalavimų inžinerijos proceso brandos lygmenis pasiūlė Pete Sawyer, Ian Sommerville ir jų bendraautoriai [99, 100, 101, 102]. Šie pasiūlymai išsamiai yra aprašyti knygoje [107]. Buvo išskirti trys reikalavimų inžinerijos proceso brandos lygiai (50 pav.): pradinis, pakartojamo proceso ir formalizuoto proceso.



50 pav. Reikalavimų inžinerijos proceso brandos lygiai.

Šių lygių turinys atskleistas 51 pav. pateiktoje lentelėje.

Brandos lygis	Lygio aprašas
Pradinis	Procesas nereguliuotas, priklauso nuo konkretaus analitiko gebėjimų, patirties ir įgūdžių. Projektai dažnai vėluoja, viršijamas jų pradinis biudžetas. Reikalavimai nėra tvarkomi. Reikalavimų specifikacijos arba apskritai nėra, arba ji neišsami, pilna dviprasmybių, jos struktūra priklauso nuo to, kas ją rašė.
Pakartojamas procesas	Reikalavimų specifikacija standartizuota, nustatyta, kaip aprašyti reikalavimus, nustatytos reikalavimų tvarkymo procedūros. Žinoma, kaip iš anksto įvertinti projekto kaštus ir terminus. Naudojami kai kurie pažangūs reikalavimų inžinerijos metodai, įdiegtos juos palaikančios programinės priemonės.
Formalizuotas procesas	Reikalavimų inžinerijos procesas, įskaitant ir vadybinės procedūras, yra standartizuotas, dokumentuotas, integruotas į bendrą technologinį programų sistemos kūrimo procesą ir yra nuolat tobulinamas.

51 pav. Reikalavimų inžinerijos proceso brandos lygių aprašai.

Trys (o ne penki) lygiai buvo todėl išskirti, kad tuo metu (t. y. apie 1998 metus) tvarkytis su reikalavimais gebėjo tiek mažai organizacijų, jog sertifikavimo centrams nebuvvo prasmės užsiimti aukštesniu nei trečio lygio sertifikatų išdavimu [108]. Apskritai, lygių apibrėžtys grindžiamos praktikoje pasiteisinusių metodų (angl. *best practices*) naudojimu. Pete Sawyer ir Ian Sommerville savo knygoje [107] apraše 36 bazinius tokio pobūdžio metodus, 21 papildomą, sudėtingesnį metodą ir 9 didžiausio sudėtingumo metodus, paprastai naudojamus tik kuriant kritines sistemas. Šis metodų rinkinys leidžia apibrėžti ir aukštesnius negu trečias reikalavimų inžinerijos proceso lygius. Vertinant organizacijoje įdiegto reikalavimų inžinerijos proceso brandą, yra ne tik žiūrima, kurie iš praktikoje pasiteisinusių metodų yra naudojami, bet ir, atsižvelgiant į tų metodų naudojimo būdą (52 pav.), jie yra vertinami balais. Viso proceso branda yra įvertinama, sumuojuant gautus įverčius.

Įvertis balais	Praktikoje pasiteisinusio metodo naudojimo būdas
0	Nenaudojamas.
1	Naudojamas konkrečių darbuotojų nuožiūra.
2	Naudoja dauguma darbuotojų, bet kiekvienas tai daro savaiip.
3	Kada ir kaip naudoti yra nustatyta organizacijoje veikiančiais standartais.

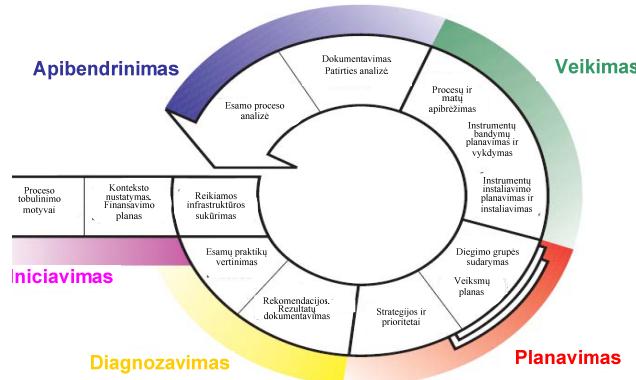
52 pav. Praktikoje pasiteisinusių metodų naudojimo vertinimas.

Pastaruojančiu metu tapo populiaru kurti nedideles programų sistemas, naudojant vadinamojo ekstremaliojo programavimo <Trm10> metodus. Kadangi ši metodika

ignoruoja daugumą tradicinių reikalavimo inžinerijos metodų, kyla klausimas, kaip vertinti šios metodikos reikalavimų inžinerijos proceso brandą ir ar apskritai čia galima kalbėti apie kokį nors reikalavimų inžinerijos procesą. Šį klausimą nagrinėjo keletas autorių [82, 83, 84, 86]. Vieno iš galimybų brandos autorių, Mark C. Paulk nuomone, ekstremaliojo programavimo reikalavimų inžinerijos procesas yra priskirtinas 2 brandos lygiui, nes yra naudojamos naudotojų istorijos <Trm31>, testai naudojami kaip reikalavimų specifikacija, yra nuolatinis kontaktas su užsakovo atstovu <Trm33> ir vyksta permanentinis produkto integravimo ir testavimo procesas [86]. Kai kurie kiti autoriai [82, 83, 84] su tuo nesutinka ir mano, kad ekstremaliojo programavimo reikalavimų inžinerijos procesas, kalbant Gebėjimų brandos modelių integravimo projekto terminais, yra tolydus ir todėl jam apskritai negalima priskirti kokį nors konkretų brandos lygi. Tačiau, kaip jau minėjome, vis vien galima kalbėti apie jo brandą, nagrinėjant kurie iš praktikoje pasiteisinusių metodų yra naudojami. Kaip teigama darbe [84], ekstremaliojo programavimo metodikoje yra naudojamos 7 iš bazinių metodų: vertinamas sistemos įgyvendinamumas, reikalavimai išvedami iš verslo tikslų, planuoojamos konfliktinės situacijos ir konfliktų sprendimas, prioretizuojami reikalavimai, modeliuojama sistemos architektūra, identifikuojamas kiekvienas reikalavimas ir nustatomi reikalavimų tvarkymo būdai. Šitaip vertinant procesą, jis yra tarp 1 ir 2 brandos lygių. Toks procesas gerai tinkta nuolat tobulinamiems produktams, bet yra visiškai netinkamas sistemoms, kurios yra perdirbinėjamos tik retkarčiais, kas keli metai.

3.9 Reikalavimų inžinerijos proceso diegimas ir tobulinimas

Iš esmės, reikalavimų inžinerijos procesas yra tam tikra technologija. Todėl reikalavimų inžinerijos proceso modelio pritaikymas konkrečiam projektui traktuotinas kaip technologijos diegimo procesas. Tas pats pasakytina ir apie reikalavimų inžinerijos proceso tobulinimą. Jis traktuotinas kaip technologijos keitimo arba jos pritaikymo naujiems reikalavimams procesas. Kaip ir kiekvienas procesas, technologijos diegimo, keitimo bei tobulinimo procesas turi būti kruopščiai suprojektuotas. Be to, jis turi būti sėkmingas. Taigi, technologijos diegimo, keitimo bei tobulinimo procesui taip pat reikalingas savas modelis. Darbas vadovaujantis teoriškai pagrįstu proceso modeliu padidina sėkmės tikimybę. Vienas iš tokų modelių yra IDEALSM [41] modelis¹⁴.



53 pav. Informacinės sistemos galimybų susiejimas su jos komponentais (paimta iš [41])

Taikant šį modelį (53 pav.), kartu turėtų būti taikomi du papildomi principai:

- patobulinimai daromi ne iš karto, o mažais žingsneliais;

¹⁴ IDEAL yra Carnegie Mellon universiteto (JAV) paslaugos ženklas.

- patobulinimai turi būti kompleksiniai, t. y. jie turi tenkinti visų vykdytojų kolektyvo narių, iškaitant ir vadybininkus, poreikius, o ne vien tik kokios nors vienos grupės, tarkime, programuotojų ar projektuotojų poreikius.

IDEAL modelis numato, kad reikalavimų inžinerijos proceso tobulinimas turi būti atliekamas per penkis žingsnius:

- Proceso inicijavimas: vadovybė priima sprendimą, kad procesas yra tobulintinas.
- Diagnozavimas: vertinamas esamas reikalavimų inžinerijos procesas.
- Planavimas: rengiamas trūkumų šalinimo planas.
- Veikimas: diegiamos ir naudojamos naujos technologijos.
- Apibendrinimas: apibendrinama įgyta patirtis, grįžtama į diagnozavimo žingsnį.

Tyrimai rodo, kad daugelis organizacijų, diegiančių ar tobulinančių savo reikalavimų inžinerijos procesą nesivadovauja šiuo ar kokiui nors kitu proceso modeliu. Jei nėra perkama kokia nors technologija, vadovybė gali net ir nežinoti, kad vyksta kokie nors reikalavimų inžinerijos proceso pertvarkymai. Aišku, toks proceso tobulinimas dažniausiai jokios konkrečios naudos neduoda. Visai kitaip yra organizacijoje, dirbančiose pagal IDEAL ar kokią nors kitą panašią metodiką. Jos rengia išsamų proceso diegimo planą, stebi patį diegimą, regisruoja iškyylančias problemas ir sprendžia, kaip jas spręsti. Jei perkamos kokios nors instrumentinės priemonės ar kokia nors metodika, su tiekėjais išsiaiškinamos perkamo produkto galimybės ir gaunamos rekomendacijos, kaip tą produktą geriau diegti. Išskyrus vieną išimtį, SW-CMM 5-to brandos lygmens organizacijas, nei viena proceso diegimo metodika nenumato jokių formalų mechanizmų, kaip pasinaudoti ankstesne technologijų diegimo patirtimi. Patartina, kad tiekėjų rekomendacijos taptų funkciniais diegimo plano reikalavimais ir kad tas planas būtų rengiamas, atsižvelgiant į anksčiau diegtų technologijų diegimo patirtį. Be to, rengiant diegimo planą, patartina vadovautis dviem principais, mažų žingsnelių principu ir visų organizacijos lygmenų apėmimo principu.

Mažų žingsnelių principas reikalauja, kad procesas būtų tobulinamas palaipsniui, žingsnelis po žingsnelio. Tokio veikimo būdo sėkmės tikimybė yra daug didesnė, negu vadinamosios „šoko terapijos“, kuomet bandoma keisti viską iš karto. Norint įdiegti reikalavimais valdomą sistemos kūrimo procesą, negalima to padaryti vienu ypu. Visų pirma reikia pradėti kontroliuoti reikalavimų pokyčius. Antras žingsnis turėtų būti bandomasis kokios nors instrumentinės sistemos, padedančios trasuoti reikalavimus, sekti jų statusą ir juos dokumentuoti, diegimas. Paskutinis žingsnis yra pradėti informaciją apie reikalavimų statusą naudoti sistemos kūrimo procesui valdyti.

Organizacijos lygmenų apėmimo principas reikalauja, kad proceso diegimo ar tobulinimo planas apimtų visus organizacijos lygmenis: atskiro darbuotojo, atskiro projekto ir organizacijos kaip visumos.

Pavyzdys

Proceso diegimo planas, parengtas siekiant įgyvendinti SWCMM modeliu numatomus reikalavimų tvarkymo proceso esminės srities tikslus, kiekvienam iš organizacijos lygmenų numato skirtingas užduotis:

- Organizaciją atstovaujančiai aukščiausiajai vadovybei numatomi kursai, kurių metu jai turi būti išaiškintą, tiksliau, įteigta, kodėl yra taip svarbu kontroliuoti reikalavimus.

Organizacijos vadovybė turi būti parengta keisti organizacijos darbo stilių ir, kas yra dar svarbiau, susidūrusi veidas į veidą su užsakovais, jiems pareikšti, kad tie laikai, kada jie galėjo nevaržomai kaitalioti reikalavimus, negrįztamai baigęsi.

- Projekto lygmenyje planas numato įsigytis instrumentinės priemonės ir įdiegti procesus, reikalingus reikalavimų baziniams komplektui tvarkyti.
- Darbuotojų lygmenyje planas numato, kaip bus baudžiami darbuotojai, kurie bandys įgyvendinti reikalavimus, neįrašytus į bazinį reikalavimų komplektą.

Šis planas taip pat pabrėžia organizacijos lygmenyje vykdomų užduočių svarbą. Jei pavyks indoktrinuoti aukščiausią organizacijos vadovybę, visa kitą įgyvendinti bus pakankamai paprasta. Tačiau, toks požiūris yra ne visuomet geras. Kartais svarbesni yra kiti lygmenys. Tai priklauso nuo diegiamų proceso patobulinimų pobūdžio. Pavyzdžiui, jei yra diegiamos analitiko darbo technologijos, svarbiausiu tampa individualių darbuotojų lygmuo, nes šiuo atveju sėkmė priklauso nuo analitikų įgūdžių ir suinteresuotumo.



Technologija	Lygmuo		
	Darbuotojo	Projekto	Organizacijos
Aiškinimasis	Treniravimas Mokymas	Technikų parinkimas. Vertinimo ir tikrinimo procedūrų adaptavimas	Nuolatinio mokymo programa
Analizė	Lavinimas Treniravimas Mokymas	Technikų parinkimas. Vertinimo ir tikrinimo procedūrų adaptavimas	Nuolatinio mokymo programa
Tvarkymas (kontrolė)	Bausmės už taisyklių pažeidimus	Proceso adaptavimas. Instrumentų adaptavimas (jei reikia)	Organizacijos darbo stilius Disciplinavimo priemonės
Tvarkymas (statusas)	Treniravimas	Instrumentinių priemonių bandymai	Organizacijos standartai
Tikrinimas ir vertinimas	Treniravimas (peržiūros, inspektavimo ir testavimo priemonės)	Procedūrų adaptavimas	Disciplinavimo priemonės
Dokumentavimas	Treniravimas	Procedūrų adaptavimas	Organizacijos standartai

54 pav. Reikalavimų inžinerijos proceso tobulinimas

54 paveikslėlyje pateiktoje lentelėje aptariamos svarbiausių reikalavimų inžinerijos technologijų diegimo bei tobulinimo priemonės. Paryškintos tos priemonės, kurios lemia diegimo (tobulinimo) sėkmę.

Technologija	Adaptavimo pastangos	Adaptavimo tikrinimo pastangos
Aiškinimasis	Vidutinės nesudėtingiems projektams Didelės sudėtingiems projektams	Didelės
Analizė	Didelės	Vidutinės
Tvarkymas (kontrolė)	Nedidelės, tačiau kartais yra sunku susitarti	Nedidelės
Tvarkymas (statuso sekimas)	Treniravimas	Nedidelės
Tikrinimas ir vertinimas	Nedidelės	Vidutinės
Dokumentavimas	Nedidelės. Vidutinės, jei dokumentai generuojami automatiškai	Nedidelės

55 pav. Pastangos reikalavimų inžinerijos procesui patobulinti.

Kaip parodyta 55 paveikslėlyje, proceso diegimo ar tobulinimo sėkmę priklauso ir nuo to, kiek pastangų reikia technologijoms bei instrumentams adaptuoti. Kai kurioms technologijoms, pavyzdžiui, reikalavimų analizės technologijoms pritaikyti reikia labai didelių pastangų. Reikalavimų aiškinimosi technologijos adaptuojanos lengviau, vertinama, kad tam reikia vidutinių pastangų. Kitos technologijos yra adaptuojanos paprastai, tam pakanka gana mažų pastangų. Tačiau vien adaptuoti technologijas nepakanka. Dar reikia įsitikinti, kad tai atlikta teisingai. Tuo įsitikinti labai sudėtinga reikalavimų aiškinimosi technologijoms, vidutiniškai sudėtinga analizės technologijoms ir gana paprasta visoms kitoms technologijoms.

55 paveikslėlyje kalbama tik apie technologijų adaptavimą, bet ne apie jų panaudojimą. Tačiau, tai yra susiję dalykai. Praktinė patirtis liudija, kad lengvai adaptuojamomis technologijomis žmonės sugeba tinkamai naudotis, net ir negaudami ekspertų konsultacijų. Ir atvirkščiai, sunkiai adaptuojamomis technologijomis dažniausiai neišmokstama tinkamai naudotis net ir tais atvejais, kuomet darbuotojai yra konsultuojami išorinių ekspertų.

4 Reikalavimų išsiaiškinimas ir formulavimas

Pradedant aiškintis ir formuluoti reikalavimus, visų pirma reikia išsiaiškinti reikalavimų šaltinius ir kaip iš tų šaltinių galima išgauti reikalavimus. Kadangi reikalavimai kyla iš verslo poreikių, o tų poreikių negalima perprasti be užsakovų ir dalykinės srities specialistų pagalbos, todėl čia ypač svarbus yra sąveikos su dalykinės srities specialistais aspektas. Kita vertus, būtų naivu tikėtis, kad užsakovai ar dalykinės srities specialistai gali paimti ir suformuluoti programų sistemos reikalavimus. Tačiau informacinių sistemų ir programų sistemų inžinieriai neretai taip galvoja ir piktinasi, kad užsakovas nežino, ko nori. Užsakovas gi nori pagerinti savo verslą, nežino programų sistemų galimybių ir iš tiesų negali pasakyti, kokios sistemos jam reikia. Arba, kaip gerai demonstruoja jau mūsų minėtas JAV oro erdvės kontrolės pavyzdys [3], skirtingi užsakovo atstovai turi skirtingus interesus ir negali tarpusavyje susitarti. Taigi, reikalavimus formuluojančią informacinių sistemų ir programų sistemų inžinierių vaidmuo čia yra labai didelis. Jie veikia kaip tarpininkai ir, paaiškėjus, kad kompromisas principiniais klausimais negali būti pasiektas, turėtų užsiimti ne antraeilių reikalavimų formulavimu, o paprasčiausiai siekti, kad projektas būtų nutrauktas. Nors visi informacinių sistemų ir programų sistemų inžinieriai pripažista, kad reikalavimų išsiaiškinimas yra labai svarbus, praktikoje jie tam vis dėlto dažniausiai skiria per mažai dėmesio. Galbūt taip yra todėl, kad čia nėra jokių griežtų standartų. Reikalavimų aiškinimasis vyksta bendraujant žmonėms vieni su kitais ir, savaimė aišku, negali būti formalizuotas. Tačiau ir šioje srityje yra sukurta pakankamai efektyvių neformalių metodų, kuriais sėkmingai galima pasinaudoti reikalavimams aiškintis. Bene svarbiausia yra išaiškinti užsakovui, kad reikalavimų neišsamumas ar netgi jų prieštaringumas gali būti pastebėti tik atliekant jau suformuluotų reikalavimų analizę ar netgi jau pradėjus naudotis sukurta programų sistema ir tada perdarymai kainuos didelius pinigus arba gali būti apskritai nebejmanomi. Todėl užsakovai turėtų traktuoti reikalavimų formulavimą labai rimtai ir atsakingai, o ne bandyti atsikratyti besistengiančių surinkti reikiama informaciją ir dalykinės srities specialistams trukdančių dirbtį jų einamajį darbą analitikų.

4.1 Reikalavimų aiškinimosi metodai

Aiškinantis reikalavimus, juos tenka išgauti iš dalykinės srities specialistų, kurie gerai išmano savo dalykinę sritį, bet paprastai mažai ką nutuokia apie tai, kaip yra kuriamos programų sistemos. Taigi, viena iš svarbiausių reikalavimo aiškinimosi problemų yra analitikų ir dalykinės srities specialistų susikalbėjimo problema. Viena vertus, analitikai ne visuomet yra įvaldę dalykinės srities terminiją ir išmano, kas dalykinės srities specialistų požiūriu yra protinė, o kas ne. Kita vertus, dalykinės srities specialistai beveik niekuomet nesupranta programuotojų vartojamos terminijos ir niekuomet neišmano, kas programų sistemų inžinierių požiūriu yra protinė, o kas ne. Trumpai tariant, analitikų ir dalykinės srities specialistų bendravimas nėra paprastas. Techniškai ir netechniškai mastančių asmenų bendravimo problemas yra vienas iš sunkiausiai įveikiamų iššūkių, su kuriais susiduriama kuriant programų sistemas. Dideliuose projektuose šią problemą bandoma spręsti analitikų grupę papildant psichologais ar netgi viešųjų ryšių specialistais. Taip pat yra kuriami įvairūs reikalavimų aiškinimosi metodai, vienu ar kitu mastu formalizuojantys bendravimo procedūras ir padedantys geriau vieniams kitus suprasti. Yra naudojami ir įvairūs neformalūs reikalavimo aiškinimosi metodai, pavyzdžiui, "smegenų šurmas" <Trm76>, prototipai, bendro darbo grupės <Trm1>, kokybės funkcijų sklaida

<Trm24>. Tokie metodai dažniausiai yra naudojami bendraujant su motyvuotais ir patyrusiais specialistais, kurie žino, ko nori, bet patys vieni nesugeba suformuluoti savo poreikių.

Viena iš analitikų ir dalykinės srities specialistų nesusikalbėjimo priežasčių, yra tai, kad abi pusės skirtingai supranta kas tai yra „reikalavimai“. Terminas „reikalavimas“, įvardijantis tai, kas gimsta bendraujant su užsakovais, programų sistemų inžinieriams turi specialią prasmę: jis aprašo realų vartotojo poreikį, jam galima priskirti prioritetą, jo įgyvendinimą galima patikrinti testais, jį gali įvertinti užsakovas. Dalykinės srities specialistas „reikalavimus“ suvokia kaip savo norus. Formalizuotos reikalavimų aiškinimosi procedūros padeda spręsti šią bendaravimo problemą, nes jos padeda dalykinės srities specialistams suprasti savo poreikius ir juos tinkamai suformuluoti. Be to, tokios procedūros turi tam tikrą papildomą efektą. Pasinaudojės jomis, dalykinės srities specialistas pradeda geriau suprasti, ką gi jis iš tiesų daro, ir su kokiomis realiomis problemomis jis susiduria. Jis pradeda skirti savo norus ir savo poreikius. Analitikas iš to taip pat gauna naudos. Bendravimas tampa dalykiškesnis, suformulojamos realios problemos ir atsiranda galimybė kartu svarstyti, kokiu mastu tas problemas yra realu išspręsti. Tačiau, nepaisant šių privalumų, ir formalizuoto bendaravimo procedūros visų bendaravimo problemų toli gražu neišsprendžia. Jos padeda lengviau susikalbęti dėl vienų ar kitų ribojimų, bet nepadeda įtraukti dalykinės srities specialistų į reikalavimų formulavimo procesą. Jos papildo aukščiau minėtus neformalius reikalavimų aiškinimosi metodus, bet jokiu būdu jų nepakeičia.

Norint įdiegti formalizuoto bendaravimo procedūras, visų pirma reikia, kad abi pusės pripažintų, jog reikalavimų išsiaiškinimas yra problema ir kad skirtingi žmonės ne vienodai supranta, kas tai yra „reikalavimas“. Po to, reikia išmokyti analitikus naudotis tomis procedūromis. Be abejo, tam reikia ne tik paskaitų, bet ir praktinių užsiėmimų. Be to, jie jau turi turėti išugdytus bendaravimo su nepažįstamais žmonėmis įgūdžius apskritai. Tačiau, net ir visa tai padarius, nereikia tikėtis, kad kokia nors viena metodika ar procedūra išspręs visas reikalavimų aiškinimosi problemas. Tikimybę, kad užsakovas tinkamai suformuluos savo poreikius ir analitikas iš tiesų juos supras, galima padidinti tik derinant tarpusavyje kelis skirtingus reikalavimų aiškinimosi ir vertinimo metodus, tačiau šimtaprocentinės garantijos, kad tikrai šitaip įvyks, nėra ir tuomet. Analitikais tampama aiškinantis reikalavimus, o ne mokantis, kaip juos reikia aiškintis. Išklausę atitinkamus kursus ir įgiję reikiamus praktinius įgūdžius, žmonės tiktais įgyja analitiko darbui reikalingą kompetenciją ir gali pradėti dirbti, bet tikrais analitikais jie taps tik po kelių metų, padirbėję 3-4 realiuose projektuose. Be abejo, mokant formalizuotų bendaravimo procedūrų jau turinčius praktinę patirtį, bet iki tol niekada tokį procedūrų nenaudojusių, viskas vyks greičiau, bet vis vien tam prireiks mažiausiai pusės metų.

Dabar pakalbékime apie pačius reikalavimų aiškinimosi metodus. Svarbiausieji reikalavimų aiškinimosi metodai aptarti daugelyje šaltinių, pavyzdžiui, [23, 36, 54, 63, 74, 90]. Apie daugumą iš jų buvo išsamiai kalbėta bakalauro studijų studentams skirtame programų sistemų inžinerijos kurse ir šioje knygoje išsamiau jų neaptarinėsime. Jų esmė pateikta 3 lentelėje.

3 lentelė. Svarbiausieji reikalavimų aiškinimosi metodai.

Metodo pavadinimas	Trumpas metodo apibūdinimas
Interviu	„Tradicinis“ reikalavimų aiškinimosi metodas. Iš anksto kruopščiai suplanuotas analitiko pokalbis su užsakovo ar kitos suinteresuotos šalies atstovu, vedamas tikslu gauti

Metodo pavadinimas	Trumpas metodo apibūdinimas
	<p>atsakymus į analitiką dominančius konkrečius klausimus.</p> <ul style="list-style-type: none"> • <u>Struktūruotas interviu.</u> Atsakinėjama į analitiko iš anksto parengtus ir kruopščiai apgalvotus klausimus. • <u>Nestruktūruotas interviu.</u> Analitikas turi parengę kelis klausimus, bet kitus klausimus jis sugalvoja pokalbio eigoje. • <u>Telefoninis interviu.</u> Tinka, kuomet reikia gauti labai konkrečius atsakymus į nesudėtingus klausimus.
Apklausa	Informacijos rinkimas kompiuteriniu arba paprastu paštu. Išsiuntinėjamas iš anksto parengtas klausimynas ir prašoma raštu atsakyti į ten pateiktus klausimus. Geriausiai tinka kiekybinio pobūdžio informacijai rinkti. Paprastai atsako tik apie 10% respondentų.
Kolektyvinis svarstymas <Trm27>	Reikalavimų aiškinimosi metodas, padedantis gauti tam tikrą sinergetinį, arba, kitaip tariant, sumarinį efektą. Dirbdama kartu grupė žmonių gali išsiaiškinti greičiau ir galbūt daugiau, negu tai pavyktų padaryti analitikui, dirbant su kiekvienu iš tų žmonių atskirai. Svarstymo metu gali vykti „smegenų šturmo“ sesijos, gali būti tikslinamos taip gimusios idėjos ir gali būti išsiaiškinta daugelis dalykų, kurių nepavyktų išsiaiškinti imant interviu, nes analitikas nieko apie tai nežinojo ir, savaimė aišku, apie tai nieko neklause. Be to, tokio svarstymo metu išryškėja skirtinges nuomonės bei prieštaringi reikalavimai ir galbūt netgi gali būti pasiekti tam tikri kompromisai. Kita vertus, analitikui nėra paprasta tokius pasitarimus vesti, nes galima „uzstrigtī“ ant smulkmenų, nukrypti į šoną, arba priimti neteisingus sprendimus, spaudžiant kokiai nors interesų grupei arba bijant paprieštarauti kokio nors autoritetingo grupės nario nuomonei.
Fokuso grupė <Trm11>	Kolektyvinio svarstymo metodas. Surenkama nedidelė suinteresuotų šalių atstovų grupė ir jai pateikiamas koks nors vienas klausimas. Siekiama suvienodinti galimai skirtinges požiūrius. Gerai tinka nuomonėms bei paprastiems ar labai išdetalizuotiems duomenims rinkti.
Kolektyvinė reikalavimų analizė <Trm26>	Fokuso grupę pramenantis reikalavimų aiškinimosi būdas. Suinteresuotų šalių atstovai yra suburiami į darbo grupę, vadovaujamą analitiko-moderatoriaus, ir, analizuodami verslo funkcijas, procesus, veiklas bei duomenis, formuluoja sistemos reikalavimus. Nuo fokuso grupės skiriasi savo paskirtimi ir tuo, kad grupė yra ilgalaikė.
„Smegenų šturm“ <Trm76>	Grupinis idėjų generavimo, tikslinimo ir tolesnio vystymo būdas. Grindžiamas bendra diskusija kokiui nors klausimu. Diskusijos dalyviams leidžiama išsakyti viską, kas jiems ateina į galvą. Nei viena idėja nėra atmetama. Idėjos pradedamos svarstyti, kritikuoti ir vertinti, po to kai visi

Metodo pavadinimas	Trumpas metodo apibūdinimas
Maketavimas	išsako savo idėjas. Idėjų išsakymui skiriama ne daugiau kaip valanda.
Rašytinių šaltinių analizė	Reikalavimų aiškinimosi, panaudojant veikiantį arba neveikiantį sistemos modelį (maketą), metodas. Paprastai minketas neatlieka jokių realių skaičiavimų, modeliuoja naudotojo interfeiso ekranus, užklausas bei generuojamas ataskaitas ir yra naudojamas naudotojo interfeisų reikalavimams aiškintis. Naudojant minketą, patogu detalizuoti ir toliau aiškintis ne iki galo suprastus reikalavimus. Minketas sukuria tam tikrą kontekstą, padedantį dalykinės srities specialistams suprasti, kokios informacijos iš jų tikisi gauti analitikas. Minketą galima įgyvendinti daugeliu būdų, nuo vartotojo interfeiso langų braižymo popieriuje iki programų sistemų beta versijų panaudojimo. Minketai yra naudingi ne tik aiškinantis reikalavimus, bet ir juos vertinant.
Stebėjimas	Reikalavimų aiškinimosi metodas, padedantis „pamatyti“ tai, ko negalima „išgirsti“. Analitikas aiškinasi dalykinės srities specialistų užduotis, stebėdamas kaip tie specialistai dirba, kaip jie naudojasi turima programine įranga ir kaip jie bendrauja tarpusavyje. Metodas yra gana brangus, bet kai kurių verslo proceso detalių kitaip išsiaiškinti nepavyksta. Be to, iš šalies stebėdamos verslo procesą, analitikas gali pastebeti trūkumus, kurių patys verslo proceso dalyviai nebepastebi, nes yra prie jų pripratę ir mano, jog kitaip negali ir būti.
Darbo vietų lankymas	Vienas iš tiesioginio stebėjimo metodų. Naudojamas, norint tiesiogiai susipažinti su darbo procesais, užduočių vykdymu, darbo aplinka ir darbo sąlygomis. Tinka, kai reikia išsiaiškinti kitais metodais sunkiai išsiaiškinamus niuansus.
Dalykiniai specialistų darbo scenarijai	Maketavimą primenantis metodas, padedantis sukurti reikalavimų aiškinimosi kontekstą. Aprašius dalykinio specialisto darbo scenarijų, galima kalbėtis su tuo specialistu apie scenarijuje aprašytas jo vykdomas užduotis ir uždavinėti jam klausimus „O kaip tai daroma?“ ir „O kas jeigu?“. Scenarijams aprašyti paprastai yra naudojamos UML užduočių ir sekų diagramos ir jas papildantys formalizuoti tekstiniai užduočių ir scenarijų aprašai.
Konceptinis modeliavimas	Reikalavimų aiškinimosi metodas, panaudojant konceptinį kompiuterizuojamo verslo modelį. Modelis aprašomas UML arba kokia nors kita modeliavimo kalba. Konstruodamas ir analizuodamas tokį modelį, analitikas

Metodo pavadinimas	Trumpas metodo apibūdinimas
Kognityviniai metodai	išsiaiškina verslo transakcijas, duomenis ir kitas verslo sistemos detales.

4.2 Reikalavimų šaltiniai

Pasinaudoti 3 lentelėje pateiktais metodais ir pradėti aiškintis reikalavimus galima tiktais po to, kai išaiškėja reikalavimų šaltiniai. Kitaip tariant, visų pirma reikia išsiaiškinti, su kuo gi reikalavimus reikia aiškintis. Analitikas turi suprasti, kad išsiaiškinti reikalavimų šaltinius yra labai svarbu. Tai svarbu dėl daugelio priežasčių ir visų pirma todėl, jog kai kurie dalykinės srities specialistai nesugeba papasakoti, ką jie dirba, praleidinėja esmines detales arba apskritai nenori ar nesugeba bendrauti su analitikais. Todėl būtina pasirinkti tinkamus žmones. Tačiau net ir pasirinkus bendrauti nusiteikusius ir komunikalius žmones, reikalavimų aiškinimasis nėra paprastas darbas. Kad būtų surinkta korektiška reikiama informacija, analitikui tenka „padirbėti iki prakaito“. Be abejo, reikalavimų aiškinimosi metodai tą darbą šiek tiek palengvina.

Yra penki svarbiausieji programų sistemų reikalavimų šaltiniai: verslo tikslai; dalykinės žinios; šalys, vienaip ar kitaip suinteresuotos kuriama programų sistema; operacinė kuriamos programų sistemos aplinka ir verslo sistemos organizacinė aplinka. Apie šiuos šaltinius jau buvo kalbėta bakalauro pakopos studentams skirtame programų sistemų inžinerijos kurse, tačiau ten jie buvo apžvelgti nepakankamai išsamiai. Todėl čia pakalbėsime apie juos dar kartą.

4.2.1 Verslo tikslai

Kaip jau kalbėjome 2 skyriuje, didžioji dauguma versle naudojamų programų sistemų reikalavimų yra išvedami iš verslo sistemos tobulinimo tikslų. Tiksliau kalbant, šie tikslai yra pačio aukščiausio lygmens programų sistemos reikalavimai. Būtent jie pagrindžia būtinybę kurti sistemą, t. y. atsako į klausimą „Kodėl sistemą reikia kurti?“. Tai pats svarbiausias reikalavimų šaltinis. Verslo vizijos ir verslo tobulinimo tikslų formulavimas yra strateginio planavimo uždaviniai ir, vadovaujantis turinių atskyrimo principu, jie turėtų būti atskirti nuo reikalavimų inžinerijos problematikos. Kitaip tariant, tai turėtų būti organizacijos vadovybės, verslo konsultantu, verslo inžinierių, o ne programų sistemas kuriančių analitikų rūpestis. Analitikai turėtų tuos tikslus tiktai suvokti, detaliai išsiaiškinti ir perteikti sistemą kuriančiam inžineriniam personalui, visų pirma sistemos ir jos posistemų architektams. Tačiau praktikoje dažnai esti kitaip. Daugelyje organizacijų verslo sistemos tobulinimo tikslai arba apskritai nėra suformuluoti, arba yra suformuluoti netiksliai ir prieštaragingai. Todėl analitikui, bendradarbiaujant su užsakovais ir, jeigu yra tokia galimybė, su verslo konsultantais, paprastai tenka nemažai paplušeti, kol

verslo tikslų aprašas igauna reikiamą pavidalą. Formuluojant verslo sistemos tobulinimo tikslus, ypatingą dėmesį reikia atkreipti į kiekvieno tiksloto svarbą, lyginant jį su kitais tikslais, ir į programinių jo palaikymo priemonių įgyvendinimo kaštus. Paprasčiausias būdas išsiaiškinti tuos dalykus yra atliki vadinamąją įgyvendinamumo analizę¹⁵ (angl. *feasibility study*) [74]. Kaip tai yra daroma, išsamiai buvo aptarta bakalauro pakopos studentams skirtame programų sistemų inžinerijos kurse.

4.2.2 Dalykinės žinios

Analitikui reikia suprasti, kas vyksta dalykinėje srityje arba, kitaip tariant, kuo vadovaudamiesi dalykinės srities specialistai daro vienus ar kitus sprendimus, kodėl jie dirba taip, kaip jie dirba. Kaip, tarkime, buhalteris skaiciuoja darbo užmokestį ar sandėlininkas apskaito gautus ir išduotus gaminius? Be abejo, aiškindamas verslo ypatumus ir nagrinėdamas jo kompiuterizavimo tikslingumą ir galimybes, analitikas privalo visa tai pakankamai gerai išmanysti. Todėl jam reikia išstudijuoti šaltinius (vadovėlius, žinynus, teisinius aktus ir kt.), kuriuose yra aprašytos atitinkamos dalykinės žinios. Viena vertus, tos žinios padeda geriau suprasti užsakovų bei kitų sistema suinteresuotų subjektų poreikius, kaip sakoma, „suprasti juos iš pusės žodžio“ ir įgalina išspręsti kai kuriuos reikalavimų prieštaravimus. Kita vertus, kai kurios žinios apie dalykinę sritį, pavyzdžiu, kokių duomenų reikia vienai ar kitai verslo užduočiai vykdyti, tampa kuriamos programų sistemos reikalavimais. Taigi, atliekant kompiuterizuojamas verslo sistemos analizę, analitikui tenka ne tik tyrinėti verslą, bet ir mokytis. Šiuo metu, kada dauguma dalykinės srities specialistų savo kasdieniniame darbe naudojasi kompiuteriais, jie apskritai gali nebeišmanyti kaip kurių savo darbo procedūrų, nes jos yra kompiuterizuotos ir jiems pakanka tiktais mokėti jomis pasinaudoti. Todėl, keičiant vieną programų sistemą kita, analitikui gali nepavykti išsiaiškinti daugelio dalykų kitaip, kaip studijuojant dalykines žinias.

4.2.3 Šalys, suinteresuotos kuriama programų sistema

Dažnas projektas baigiasi nesėkmė vien todėl, kad stengiamasi išsiaiškinti tik kurios nors vienos kuriamos programų sistema suinteresuotos šalies, paprastai, užsakovo, keliamus reikalavimus. Kitos suinteresuotos šalys (sistemos aptarnavimo ir priežiūros tarnybos, verslo sistemas aptarnaujami klientai, verslo partneriai, valstybinės įstaigos ir kt.) paprasčiausiai yra ignoruojamos. Analitikas privalu nustatyti visus, ką vienaip ar kitaip palies kuriamoji sistema, ir išsiaiškinti visų jų keliamus reikalavimus [63]. Suinteresuotąsių šalis aptarėme 3.4 poskyryje, taip pat kalbėdami apie „Volere“ reikalavimų inžinerijos proceso modelį. Tačiau vienas svarbus klausimas ten liko beveik neaptartas. Tai skirtumai tarp skirtingu organizacijos darbuotojų požiūrių į kuriamą programų sistemą. Beje, tai glaudžiai susiję ir su pirmuoju reikalavimų šaltiniu – verslo tikslais. Visus kompiuterizuojamas organizacijos darbuotojus grubiai galima suskirstyti į tris grupes – į aukščiausią vadovybę, vidutiniosios grandies vadovus ir operatyvinio lygmens darbuotojus. Jų nuomonės apie tai, kokia turėtų būti kuriamoji programų sistema, gali gerokai skirtis. Tai sėlygota objektyvių priežasčių. Aukščiausioji vadovybė formuluoja verslo sistemos viziją ir sprendžia, kaip tą viziją įgyvendinti. Ji masto strateginio lygmens terminais ir tiktais ji gali nuspręsti, kokia iš tiesų yra kiekvienos išorinės analizės metu išryškintos verslo problemos ar grėsmės svarba. Kita vertus, aukščiausiosios grandies vadovai dažniausiai nežino konkretių verslo procedūrų detalių ir tos detalės jiems gali atrodyti neesminės. Tuo tarpu vidutiniosios grandies vadovai yra daug arčiau realios veiklos, bet tik iš dalies suvokia strateginius verslo tikslus, nes jie yra atsakingi tik už

¹⁵ Pastaruoju metu Lietuvoje pradėta vartoti pažodinį vertalą *galimybų studija*.

tam tikrą jiems patikėtą veiklos barą. Būtent jie gali objektyviausiai vertinti, ar verslo problemos tikrai yra iššauktos tų priežasčių, kurios buvo išryškintos verslo sistemos vidinės analizės metu. Taip pat būtent jie gali geriausiai pasakyti, kaip vienus ar kitus reikalavimus tenkinanti sistema paveiks jiems patikėtą veiklos barą. Kita vertus, vidutiniosios grandies vadovai linkę pervertinti jiems patikėto veiklos baro svarbą ir poveikį visai verslo sistemai. Jiems gali atrodyti, kad reikalavimai, nusakantys kitiems veiklos barams palaikyti reikalingas sistemos savybes, yra ne tokie svarbūs, o gal ir apskritai neesminiai. Pagaliau operatyvinio lygmens darbuotojams nelabai rūpi strateginiai verslo tikslai. Jiems svarbiausiai, kaip programų sistema keis kasdienines jų darbo operacijas. Be to, kuriama programų sistema neretai gali padidinti jų darbo krūvį ir sukelti kitų papildomų rūpesčių. Jiems reikės vesti duomenis į kompiuterį, rūpintis jų apsauga, mokytis pasinaudoti programų sistema, nervintis dėl techninės ar programinės įrangos trikčių, trukdančių laiku padaryti vienus ar kitus darbus. Todėl būtent šios problemos jiems atrodo svarbiausios. Tai, kad bus pašalintos vienos ar kitos strateginio lygmens verslo problemos, pavyzdžiui, verslas taps pelningesnis, operatyvinio lygmens darbuotojams mažai rūpi. Kaip jau rašėme 3.4 poskyryje, skirtingi yra ir kitų kuriama programų sistema suinteresuotų šalių keliami reikalavimai. Todėl, dirbdamas su visomis suinteresuotomis šalimis, analitikas turi suvokti jų nuomonų skirtumų objektyvią prigimtį, gebeti išryškinti galimus prieštaravimus ir padėti integruti skirtingus požiūrius.

4.2.4 Operacinė kuriamos programų sistemos aplinka

Didelė dalis kuriamos programų sistemos reikalavimų yra vienaip ar kitaip išvedama iš aplinkos, kurioje veiks (operuos) kuriamoji programų sistema. Pažymėtina, kad operacinė aplinka be kita ko apima ir kitas programų sistemas, su kuriomis turės keistis duomenimis ar kitaip „bendrauti“ kuriamoji programų sistema. Iš operacinės aplinkos, visų pirma iš verslo taisyklių, yra išvedami, pavyzdžiui, kuriamos programų sistemos našumo, patikimumo, apsaugos ir sąveikos su kitomis sistemomis reikalavimai. Apie tai, kaip yra atliekamas išvedimas, kalbėjome 2 skyriuje. Taigi, iš operacinės aplinkos išvesti reikalavimai yra ne mažiau svarbūs, negu iš kitų šaltinių gauti reikalavimai. Didele dalimi būtent nuo jų priklauso sistemos įgyvendinamumas ir jos kūrimo kaštai. Be to, šie reikalavimai riboja sprendimus, priimamus sistemos projektavimo metu [112].

4.2.5 Organizacinė verslo sistemos aplinka

Programų sistemos sėkmė gana dideliu laipsniu priklauso nuo to, ar buvo atsižvelgta į organizacijos korporacinę kultūrą, tradicijas ir darbo stilių. Analitikas privalo visa tai išsiaiškinti ir suformuluoti tokius programų sistemos reikalavimus, kad sistema būtų pritaikyta organizacijai, o ne, atvirkščiai, organizacijai tektų taikytis prie sistemos.

4.3 Reikalavimų formulavimo būdai

Programų sistemos reikalavimai gali būti formuluojami dviem būdais:

- nusakant pageidaujamas kuriamos programų sistemos savybes,
- nusakant nepageidaujamas kuriamos programų sistemos savybes.

Pirmuoju atveju reikalavimai yra vadinami *tiesioginiai*, antruoju – *invertuoti*. Tiesioginiai reikalavimai taip pat gali būti formuluojami keliais skirtingais būdais:

- deklaratyviai,
- procedūriškai,

- pateikiant reprezentatyvių pavyzdžių rinkini,
- pateikiant kuriamos sistemos prototipą,
- pateikiant kuriamos sistemos modelį.

Be to, reikalavimai gali būti neformalūs, iš dalies formalūs (formalizuoti) arba formalūs. Neformalūs reikalavimai formuluojami kokia nors natūralija kalba, pavyzdžiui, lietuvių kalba. Iš dalies formalūs ir formalūs reikalavimai formuluojami dirbtinėmis kalbomis. Jei kalba yra formali, t. y. turi ir formalią sintaksę ir formalią semantiką, tai ja suformuluoti reikalavimai irgi yra formalūs. Formalios specifikavimo kalbos pavyzdys yra Z kalba. Jei kalba turi tik formalią sintaksę, o jos semantika yra tik iš dalies formali arba netgi visiškai neformali, tai ja suformuluoti reikalavimai taip pat yra iš dalies formalūs. Iš dalies formalios specifikavimo kalbos pavyzdys yra UML kalbos pavyzdys.

4.4 Nefunkcinių reikalavimų formulavimo problemos

Nefunkciniai reikalavimai <Trm32> specifikuojant pageidaujamas sistemos kokybės charakteristikas, įvairius ribojimus ir kitus panašius dalykus. Tiesiogiai įgyvendinti jų negalima. Jie turi būti arba *operacionalizuoti*, t. y. išreikšti kokiais nors funkciniais ar projekto reikalavimais, arba paversti projektavimo ar programavimo technologijos reikalavimais. Pavyzdžiui, vienas iš būdų įgyvendinti programų sistemos patikimumo reikalavimus yra performuluoti juos į reikalavimus padidinti tam tikrų sistemos komponentų testavimo apimtį.

Tikslios nefunkcinių reikalavimų apibréžties nėra. Nėra ir išsamaus jų sąrašo. 2.3 poskyryje ISO/IEC 9126 standarto [ST20] pasiūlyta nefunkcinių reikalavimų klasifikacija apima tik pakankamai aukšto lygio reikalavimus ir taip pat nėra išsami. Visus nefunkcinius reikalavimus yra priimta skirtysti į tuos, kurių reikia vartotojui, ir tuos, kurių reikia sistemos aptarnavimo ir priežiūros tarnyboms. Pirmają grupę priimta vadinti *su produktu susijusiais nefunkciniais reikalavimais*, antrają – *su procesu susijusiais nefunkciniais reikalavimais*. Pirmajai grupei priskiriami patikimumo, našumo ir panaudojamumo reikalavimai, antrajai grupei – prižiūrimumo ir perkeliamumo reikalavimai (žr. 2.3 poskyrį). Pirmosios grupės reikalavimai gali būti susieti su sistemos kokybės charakteristikomis, kurios bent jau teoriškai gali būti matuojamos. Todėl jie gali būti formuluojami kiekybiškai, t. y. charakteristikų terminais. Antrosios grupės reikalavimų įgyvendinimo laipsnio tiesiogiai pamatuoti negalima. Jis matuojamas tik netiesiogiai, vidutinėmis darbo sąnaudomis, reikalingomis sistemai aptarnauti, prižiūrėti ar perkelti iš vienos aplinkos į kitą terminą. Tiktai šitaip juos ir galima kiekybiškai suformuluoti. Tai yra tam tikra problema, nes testuojant sistemos tinkamumą praktiškai yra neįmanoma atlikti tokius bandymus, kaip sistemos kėlimas iš vienos platformos į kitą, sistemos trikių priežasčių paieška ar kokie nors sistemos perdarymai. Tokie bandymai būtų per daug brangūs. Todėl yra labai sunku ir patikrinti, kokiu mastu tokie reikalavimai iš tiesų yra įgyvendinti sistemoje.

Viena iš svarbiausių nefunkcinių reikalavimų formulavimo problemų yra ta, kad dažniausiai jie stipriai priklauso vieni nuo kitų ir tas priklausomybes paprastai yra sunku ižvelgti. Todėl yra labai sunku taip suformuluoti nefunkcinius reikalavimus, kad jie būtų neprieštarangi. Šiuo metu šiai problemai spręsti dažniausiai naudojama vadinamoji Toronto metodika [14]. Ši metodika nefunkcinius reikalavimus nagrinėja proceso lygmenyje, t. y. nefunkciniai reikalavimai joje naudojami projektavimo procesui valdyti. Reikalavimai yra modeliuojame specialiai tam skirta GRL (angl. *Goal-Oriented Requirement Language*) kalba. Modeliuojami ir funkcioniai, ir nefunkcioniai reikalavimai. Kalba turi specialų samprotavimų aparatą, leidžiantį

samprotauti apie nefunkcinių reikalavimų savybes. Tai reikalavimų koncepcinio modeliavimo kalba. Joje yra trys pagrindinės konceptų rūšys: intenciniai elementai (angl. *intentional elements*), intenciniai ryšiai (angl. *intentional relationships*) ir aktoriai (angl. *actors*).

Intenciniai GRL elementai – tai tikslai (angl. *goals*), užduotys (angl. *tasks*), siekiai (angl. *softgoals*), įsitikinimai (angl. *beliefs*) ir ištekliai (angl. *resource*). Jie vadinami intenciniais, nes padeda aprašyti užsakovo ketinimus arba, kitaip tariant, intencijas. Tai yra daroma kuriant modelius, panaudojant kuriuos galima gauti atsakymus į tokio pobūdžio klausimus:

- kodėl sistemos reikalavimais buvo aprašyti tam tikros sistemos elgsenos arba buo aprašyti vieni ar kiti informacinių ar struktūriniai sistemos aspektai?
- kokios alternatyvos buvo nagrinėtos?
- kokiais kriterijais vadovaujantis buvo sprendžiama, kurias alternatyvas parinkti?
- kokie buvo argumentai parinkti tas, o ne kitas alternatyvas?



56 pav. Tikslų vaizdavimas GRL kalba.



Modelyje suprodyta kaip rečiai turėti sujungimą (tinklo bus būtinausias, kokią medžia nurodoti į kt.).

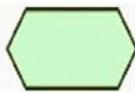
57 pav. Tikslo vaizdavimo GRL kalba pavyzdys.

Tikslas (56 pav.) yra suprantamas kaip tai, ko siekia kuri nors sistemos kūrimu suinteresuota šalis. Tai gali būti kokia nors sąlyga, kurią turėtų tenkinti sistema arba tam tikra realaus pasaulio būklė, kurios siekia suinteresuotoji šalis. Kaip pasiekti tikslą modelyje nėra specifikuojama. Tai galima padaryti skirtingais, alternatyviais būdais. 57 pav. parodytas vienas iš telekomunikacinės sistemos tikslų.

Tikslai gali turėti argumentus. Kartu su tikslo grafiniu žymeniu GRL yra vartojami šie raktiniai žodžiai:

- CONTAINS: nurodo, kad žymens dešinijį argumentą reikia patalpinti jo kairiojo elemento viduje;
- IS CONNECTED TO... BY: nurodo, kad žymens dešinysis argumentas yra susietas su jo kairiuoju argumentu ryšiu nurodytu prieš žodelį BY;
- IS ASSOCIATED WITH: nurodo, kad žymens dešinysis argumentas turi būti patalpintas šalia jo kairiojo argumento.

Tikslai yra skirstomi į verslo tikslus ir sistemos tikslus. Verslo tikslas – tai siekiama verslo reikalų būklę, o sistemos tikslas aprašo kokią nors tikslinę sistemos būseną. Paprastai sistemos tikslai yra naudojami funkciniams sistemos reikalavimams modeliuoti.



58 pav. Užduočių vaizdavimas GRL kalba.



Užduotimi nusakytas vienos iš galimų balso ryšio jungimo būdų

59 pav. Užduoties vaizdavimo GRL kalba pavyzdys.

Kaip reikia įgyvendinti tikslus parodo užduotys (58 pav.). Pavyzdžiui, 59 pav. parodyta, kaip reikia įgyvendinti 57 pav. pavaizduotą tikslą. Jei užduotis yra kokios nors aukštesnio lygmens užduoties komponentas, tai turi būti ir atitinkamas aukštesnio lygmens tikslas, kurio įgyvendinimo būdą ši aukštesnio lygmens užduotis aprašo.



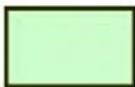
60 pav. Siekių vaizdavimas GRL kalba.



Maršrutizatorius patikimus – tai vienos iš siekių, kuriam siekiama projektuojant telekomunikacinių sistemų.

61 pav. Siekio vaizdavimo GRL kalba pavyzdys.

Užduotys – tai kuriamos sistemos projektavimo sprendimai. Jie turi tenkinti ribojimus, nusakytus nurodytais siekiais. Siekis (60 pav.) – tai salyga arba pasaulio reikalų būklė, kurią nori pasiekti aktorius. Siekis panašus į tikslą, bet, skirtingai nuo tiksllo, čia nėra griežto kriterijaus, vadovaujantis kuriuo galima pasakyti, ar jis yra pasiektas. Paprastai siekiu modeliuojamas koks nors nefunkcinis reikalavimas, ribojantis vieno ar kelių tikslų įgyvendinimo būdą. Ar siekį pavyko sėkmingai pasiekti tiesiog sprendžia vykdytojas. 61 pav. parodytas vienas telekomunikacinės sistemos projektavimo siekių.



62 pav. Ištaklių vaizdavimas GRL kalba.



Balso perdavimo vieninių tinklu sistemoje turi būti prisinaudoti plačiajuostis ryšio kanalas.

63 pav. Ištaklio vaizdavimo GRL kalba pavyzdys.

Ištaklis (62 pav.) – tai kokia nors fizinė arba informacinė esybė, be kurios negalima įgyvendinti kuriamos sistemos. Pagrindinė ištaklių problema yra jų

prieinamumas, t. y. galimybė pasinaudoti tuo ištekliu. 63 pav. parodytas vienas iš išteklių, neturint kurio neįmanoma sukurti telekomunikacinių sistemų, kurioje balsas būtų perduodamas vietiniu kompiuteriu tinklu kaip to reikalauja atitinkama užduotis (59 pav.).



64 pav. Įsitikinimų vaizdavimas GRL kalba.



Čia pateiktas argumentas, kad sprendimas parodo balsą vietinių tinklų atspindinė sistemą.

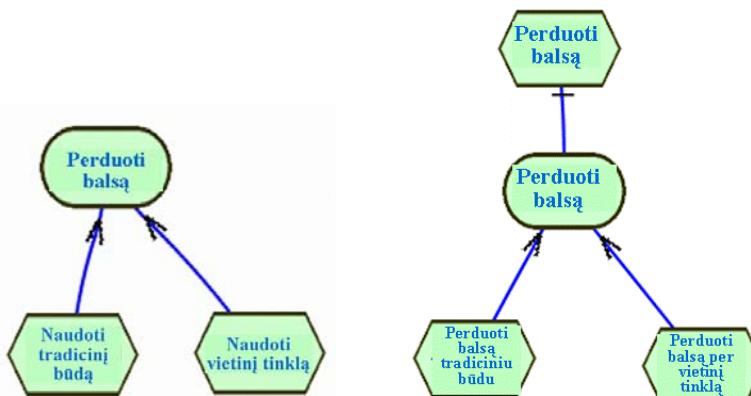
65 pav. Įsitikinimo vaizdavimo GRL kalba pavyzdys.

Įsitikinimai (64 pav.) arba, tiksliau kalbant, nuomonės ar argumentai, naudojami projektavimo sprendimams pagrįsti. Įsitikinimai įgalina nagrinėti dalykinės srities charakteristikas ir jas teisingai panaudoti priimant sprendimus. Šitaip palengvinami būsimi sistemos vertinimai, peržiūros, keitimai, pagerinamas reikalavimų trasuojamumas. 65 pav. parodytas argumentas, pagrindžiantis projektavimo sprendimą perduoti balsą vietiniu kompiuteriu tinklu (59 pav.).

Kaip jau buvo minėta greta intencinių elementų GRL yra intenciniai ryšiai. Kalboje yra tokie intenciniai ryšiai:

- tikslų ir priemonių (angl. *means-ends*),
- dekompozicijos (angl. *decomposition*),
- indėlio (angl. *contribution*),
- koreliacijos (angl. *corellation*),
- priklausomybė (angl. *dependency*).

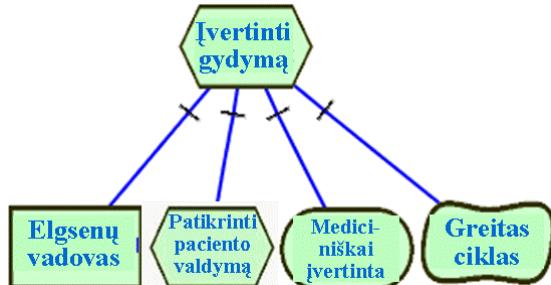
GRL modelis yra konstruojamas jungiant poromis intencinius elementus. Jie jungiami ryšiais. Šitaip gaunamas kuriamos sistemos reikalavimų modelis, kuris yra vadinamas tikslų modeliu. Taigi ryšiai yra labai svarbūs. GRL kalboje jie vartojami kaip formulų arba, kitaip tariant, teiginių ir tvirtinimų konstruktoriai.



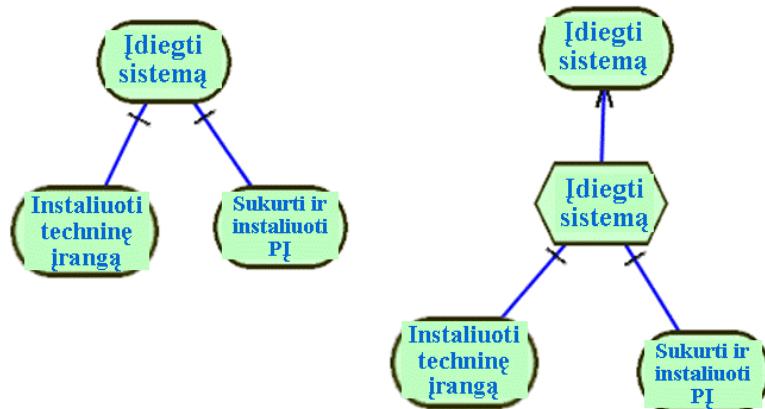
66 pav. Tikslo ir priemonių ryšys, siejantis tikslą su užduotimi, ir tos modelio konstrukcijos trumpinys.

Tikslo ir priemonių ryšys (66 pav.) aprašo, kaip yra pasiekiami tikslai. Kiekviena užduotis yra alternatyvi priemonė tikslui pasiekti. Paprastai kiekviena užduotis daro skirtingą poveikį siekiams, kuo vadovaujantis ir yra parenkama

atitinkamo projektavimo sprendimo alternatyva. Tikslo ir priemonių ir ryšiai jungia tikslo viršūnę su tam tikslui pasiekti skirtomis priemonėmis. Iš tiesų tikslo viršūnėmis gali būti tik tikslai, tačiau patogumo dėlei modelyje leidžiami trumpiniai (66 pav.) ir tuomet tikslo viršūne gali būti užduotis ar išteklis.

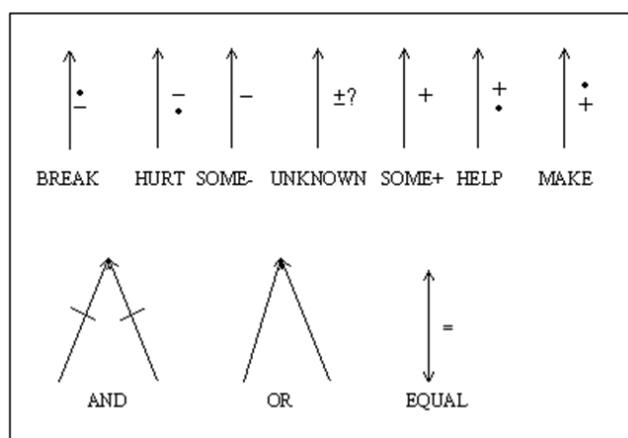


67 pav. Užduoties dekomponavimas į ištekli (elgsenų vadovas), užduotį, tikslą ir sieki.



68 pav. Kairėje parodytas dešinėje pateiktos tikslo dekompozicijos trumpinys.

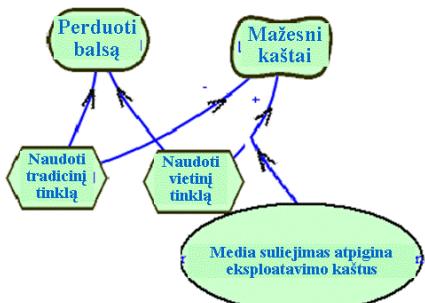
Dekompozicijos ryšys (67 pav.) aprašo, kokių kitų elementų reikia arba kokie kiti elementai turi būti prieinami, kad būtų galima įvykdyti užduotį. Užduotį galima dekomponuoti į potikslus (jie turi būti pasiekti), použduotis (jos turi būti įvykdytos), išteklius (jie turi būti prieinami) ir siekius (jie turi būti tenkinami). Tikslą galima dekomponuoti į potikslius. Iš tiesų gali būti dekomponuojamos tik užduotys, bet patogumo dėlei leidžiami modelio konstrukcijų trumpinimai ir tuomet vaizduojama taip tarsi tikslai būtų tiesiogiai dekomponuojami į potikslius (68 pav.).



69 pav. Indėlio ryšio vaizdavimo būdai.

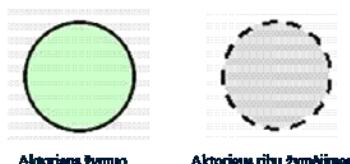
Indėlio ryšys aprašo, kokį poveikį siekiai, užduotys, įsitikinimai bei jungtys daro kitiems modelio elementams. Indėlis - tai efektas, kurį visų pirma ir norima išsiaiškinti modeliuojant reikalavimus. Galimi tokie indėlio ryšio variantai (69 pav.):

- AND: visi indėlio komponentai yra būtini ir pozityvūs;
- OR: visi indėlio komponentai yra pozityvūs, tačiau pakanka bet kurio vieno iš jų;
- MAKE: indėlio elementas yra pozityvus ir pakankamas;
- BREAK: indėlio elementas yra negatyvus ir pakankamas;
- HELP: indėlio elementas yra pozityvus, bet nepakankamas;
- HURT: indėlio elementas yra negatyvus, bet nepakankamas;
- SOME+: indėlis yra pozityvus, bet jo pakankamumo laipsnis nežinomas;
- SOME-: indėlis yra negatyvus, bet jo pakankamumo laipsnis nežinomas;
- EQUAL: vienodo dydžio indėlis abiem kryptim;
- UNKNOWN: yra tam tikras indėlis, bet nei jo pobūdis (pozityvus ar negatyvus), nei jo pakankamumo laipsnis yra nežinomi.



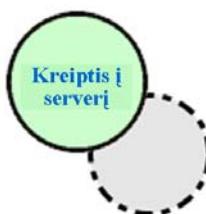
70 pav. Indėliai į siekio realizavimą.

70 pav. parodyta, kad perduoti balsą galima panaudojant tradicinę telefonų tinklą arba kompiuteriniu būdu. Pirmasis būdas sistemos kaštams turi nežinomo dydžio neigiamą poveikį, o antrasis – teigiamą, nes dviejų media suliejimas atpi gina eksplorativimo kaštus.



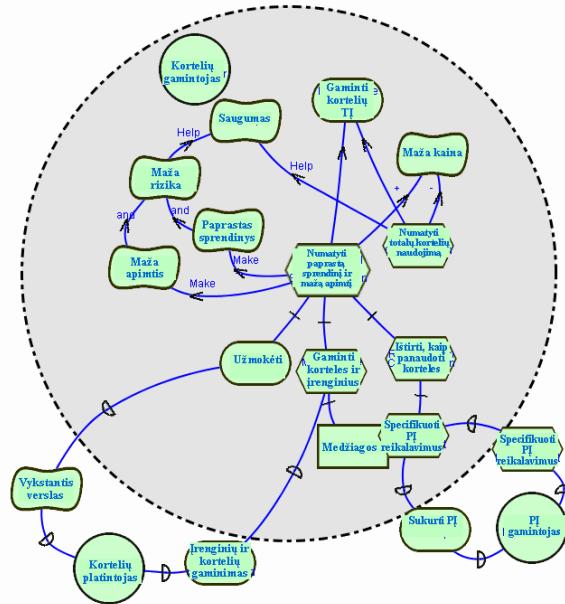
71 pav. Aktorių ir jų ribų vaizdavimas GRL kalba.

Aktorius - tai aktyvi esybė, kuri panaudodama savo žinojimus (angl. *know-how*) imasi veiksmų savo tikslams pasiekti. Vaizduojant grafiškai (71 pav.), aktoriui gali būti nustatytos ribos. Jų viduje talpinami atitinkami intenciniai elementai.



72 pav. Aktoriaus ir jų ribų vaizdavimo GRL kalba pavyzdys.

72 pav. parodytame pavyzdje balsui perduoti skirtame vietiniame kompiuterių tinkle aktorius *Kreiptis į serverį* numato tokį funkcionalumą, kuri paprastai numato tradicinis balso perdavimo tinklas. Tai parodoma pavaizduojant aktoriaus ribas ir ten pateikiant atitinkamus intencinius elementus.



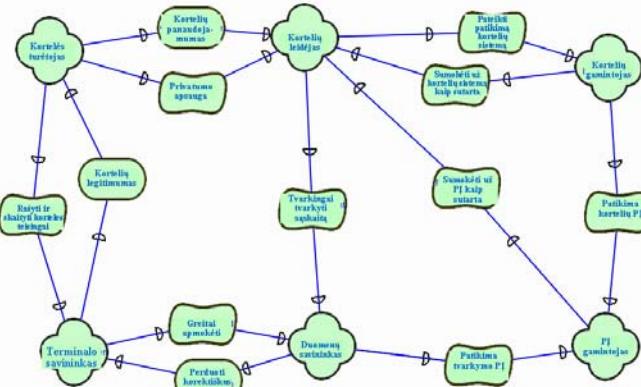
73 pav. Aktoriaus *Kortelių gamintojas* funkcionalumo modelis.

73 pav. parodytas aktoriaus *Kortelių gamintojas* funkcionalumo modelis. Šio aktoriaus tikslas yra gaminti kortelių techninę įrangą, kuri bus naudojama kuriamoje apmokėjimų kortelėmis sistemoje. Ši įranga turi būti saugi ir pigi. Galimos dvi projektavimo alternatyvos. Viena numatanti totalų kortelių naudojimą už viską atskaitant kortelėmis, kita – ribotą paprastesnių kortelių naudojimą. Parinkus antrają alternatyvą numatoma gaminti ne tik įrangą, bet ir pačias korteles. Abi alternatyvos prisideda prie siekio *Saugumas* realizavimo, bet nei viena iš jų pati savaime visiško saugumo neužtikrina. Be to, modelyje parodyta, kad sistema priklauso dar ir nuo kitų dviejų aktorių – *Programinės įrangos gamintojo* ir *Kortelių platintojo* – esančių jau už aktoriaus *Kortelių gamintojas* funkcionalumo ribų.

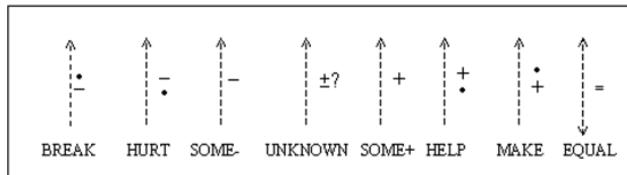


74 pav. Priklasomybių vaizdavimas GRL kalba.

Priklasomybė (74 pav.) - tai intencinis ryšys, jungiantis du aktorius. Tai reiškia, kad vienas aktorius (angl. *dependee*) kažkaip priklauso nuo kito aktoriaus (angl. *dependee*). Kaip priklasomybės yra naudojamos modeliuojant reikalavimus parodyta 73 pav. ir 75 pav.

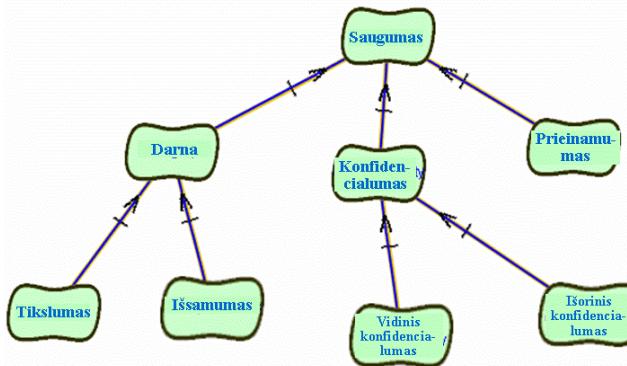


75 pav. Prisklausomybių naudojimas modeliuojant reikalavimus GRL kalba.

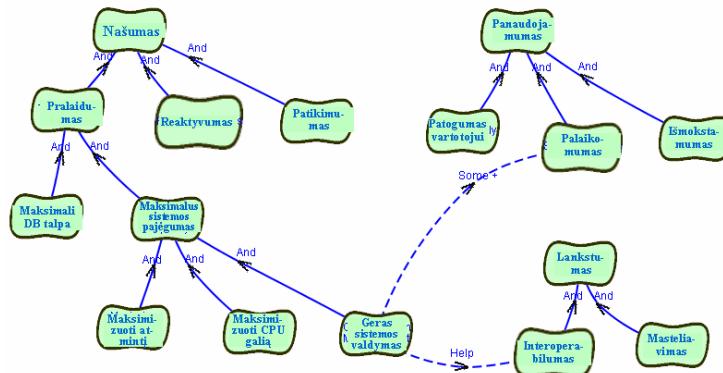


76 pav. Koreliacijų vaizdavimas GRL kalba.

Dar viena intencinių ryšių rūsis yra koreliacijos (76 pav.). Jos yra labai panašios į indėlio ryšius. Koreliacijos nuo indėlio ryšių skiriasi tik tuo, kad jos aprašo ne išreikštinį norą, bet šalutinį efektą. Koreliacijos leidžia modeliuoti skirtinį kategorijų intencinių elementų sąveiką.



77 pav. Nefunkcinių reikalavimų modelio fragmentas pavaizduotas GRL kalba.

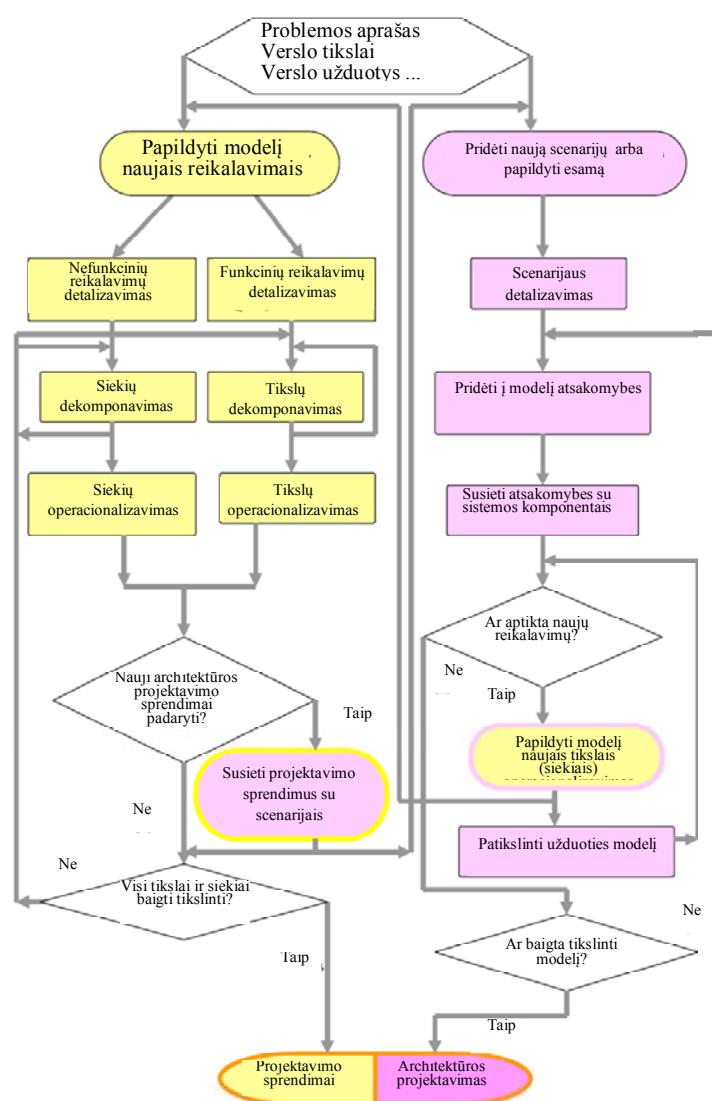


78 pav. Modelio fragmentų jungimas koreliacijos ryšiu.

77 pav. pavaizduotas sistemos nefunkcinių reikalavimų modelio fragmentas. Tokių fragmentų modelyje gali būti keletas, jie jungiami koreliacijos ryšiu (78 pav.).

Turint reikalavimų modelį GRL kalba galima atlikti tam tikrus formalius samprotavimus apie skirtingus sistemos įgyvendinimo scenarijus. Tai daroma nustatant atitinkamus tarp intencinių ir neintencinių elementų.

Be abejo GRL modeliai nepakečia detalių reikalavimų specifikacijų, aprašančių kokią sistemą reikia sukurti. Jie papildo tas specifikacijas ir padeda analizuoti bei vertinti nefunkcinus reikalavimus. Modeliuose stengiamasi parodyti kodėl vienos ar kitos elgsenos ir/arba struktūros buvo parinktos bei kodėl vieni ar kiti ribojimai buvo įvesti. Operacinėmis procesų detalėmis, sistemos komponentais bei jų sąveika kuriant tokius modelius nesidomima. Tai daroma sąmoningai. Pradiniuose analizės bei projektavimo etapuose ignoruojant tokias detales, esamas ir kuriamas programų sistemas ir jų verslo aplinkas galima modeliuoti aukštesniu, strateginiu lygmeniu. Atsakant į „Kodėl?“ klausimus, išaiškėja kokių galimybių reikia užsakovams ir ko jie nori išvengti.



79 pav. Toronto metodikos bendroji schema.

Toronto metodikos bendroji schema parodyta 79 pav. Metodika susideda iš modelio ir vertinimo procedūros, kuri nustato, kokiu laipsniu nefunkciniai reikalavimai yra tenkinami turint konkrečią projektavimo sprendimų aibę. Modelį galima vaizduoti grafiškai, taip kaip mes aptarėme. Toks modelis vadinamas *siekių*

tarpusavio priklausomybių grafu (angl. *softgoal interdependency graph (SIG)*). Modelį galima užrašyti ir teksto forma. Tačiau šioje knygoje nenagrinėsime, kaip tai yra daroma. Apie tai galima paskaityti Toronto metodiką išsamiai aprašančioje knygoje [14].

Kaip parodyta 79 pav., metodika numato tokią nefunkcinių reikalavimų operacionalizavimo schemą:

- remiantis sistemos funkciniais reikalavimais ir dalykinės srities analizės rezultatais formuluojami abstraktūs nefunkciniai reikalavimai;
- kiekvienas nefunkcinis reikalavimas išreiškiamas tam tikru siekiu;
- siekiai dekomponuojami į žemesnio lygmens siekius;
- siekiai operacionalizuojami, išreiškiami operacionalizavimo siekiais;
- operacionalizacijos dekomponuojamos toliau, tai tėsiama iki gaunamų konkretūs ir vienareikšmiškai suprantami siekiai;
- iš žemiausio lygmens siekių aibės išrenkamas poaibis, kuriuo tikimasi tenkinti pradinius reikalavimus;
- vertinimo procedūra nustato, ar pasirinktieji siekiai tikrai tenkina pradinius reikalavimus;
- jei taip nėra, tai renkamas kitas poaibis ir kartojama vertinimo procedūra;
- gauti rezultatai susiejami su funkciniais reikalavimais ir išreiškiami konkrečiais, nefunkcinius reikalavimus, kurių įgyvendinamumą galima patikrinti.

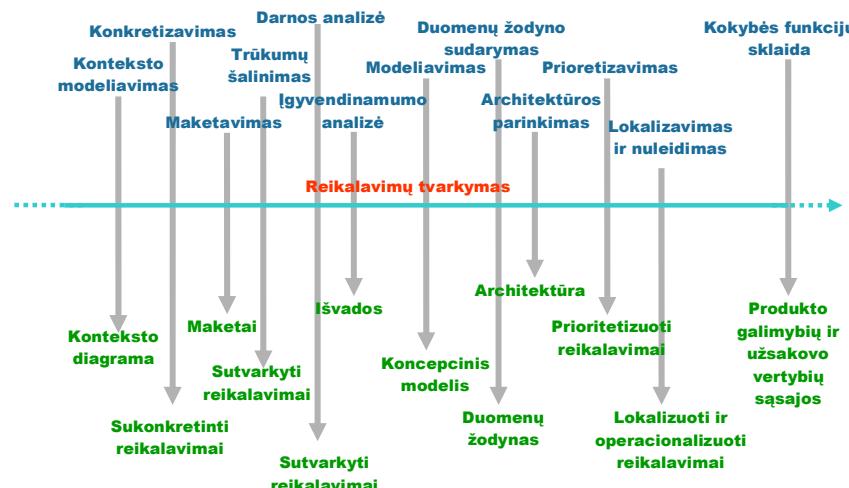
Metodika nereikalauja, kad modeliuojant reikalavimus būtų naudojamos visos GRL kalbos priemonės. Pakanka tik siekių ir priklausomybių. To pakanka siekiams dekomponuoti, operacionalizuoti ir vertinti. Tačiau šitaip dirbant prieikia daugiau laiko ir pastangu.

Kaip matome iš 79 pav., metodikoje reikalavimų inžinerijos darbai yra labai susipynę su projektavimo darbais. Tačiau schemaje parodytų projektavimo darbų neaptarinėsime, nes tai jau išeina už mūsų kurso ribų. Apie juos galima paskaityti išsamiai metodiką aprašančioje knygoje [14].

5 Reikalavimų analizė

5.1 Reikalavimų analizės esmė

Pagrindinė reikalavimų analizės <Trm47> paskirtis yra taip „atšlifoti“ reikalavimus, kad jie taptų suprantami visoms kuriamaja sistema suinteresuotoms šalims, ir „apvalyti“ juos nuo klaidų, praleidimų bei kitų trūkumų. Analizuojami visų lygmenų – verslo, vartotojo, sistemos ir programų sistemos – reikalavimai. Atliekant sistemos reikalavimų analizę, ieškomi ir šalinami tų reikalavimų prieštaravimai, nustatoma projekto apimtis arba, kitaip tariant, apibrėžiamos kuriamosios sistemos ribos ir išsiaiškinama, kaip ta sistema bendraus su savo operacine aplinka. Be to, sistemos reikalavimai yra detalizuojami iki tokio laipsnio, kad iš jų būtų galima išvesti tai sistemai reikalingos programinės įrangos reikalavimus. Priminsime, kad šioje knygoje daugiausiai dėmesio skiriamas verslo sistemoms palaikyti kuriamoms informacinėms ir programų sistemoms. Todėl ir kalbant apie reikalavimų analizę daugiausiai bus kalbama būtent apie tokio pobūdžio sistemų reikalavimų analizę. Tačiau tai nereiškia, kad kalbėsime tik apie informacinių sistemų lygmens ir programų sistemų lygmens reikalavimų analizę. Tiesiog norima pabrėžti, apie kokio pobūdžio sistemas daugiausiai kalbėsime. Analizuoti reikia visų lygmenų reikalavimus ir šiame skyriuje aptarsime visų lygmenų reikalavimų analizės klausimus.



80 pav. Reikalavimų analizė.

Bendruoju atveju, reikalavimų analizė apima šias veiklas (80 pav.):

- konteksto modeliavimą,
- reikalavimų konkretizavimą arba, kitaip tariant, aukšto abstrakcijos lygmens terminais suformuluotų reikalavimų išreiškimą žemesnio abstrakcijos lygmens terminais (tai vadinama reikalavimų nuleidimu žemyn);
- reikalavimų maketavimą <Trm51>,
- analizuojamo lygmens reikalavimų prieštaravimų bei kitų trūkumų paiešką ir šalinimą,
- analizuojamojo lygmens reikalavimų ir aukštesniojo lygmenų reikalavimų bei proceso reikalavimų darnos analizę,
- reikalavimų įgyvendinamumo analizę <Trm49>,

- koncepcinį reikalavimų modeliavimą <Trm52>,
- duomenų žodyno sudarymą,
- reikalavimams įgyvendinti geriausiai tinkančios architektūros parinkimą;
- reikalavimų prioretizavimą <Trm55>,
- reikalavimų lokalizavimą <Trm50>, nuleidimą žemyn <Trm53> ir operacionalizavimą <Trm54>,
- kokybės funkcijų sklidą <Trm24>.

Šios veiklos vadinamos vertikaliomis veiklomis.

Kaip parodyta 80 pav., greta šių veiklų yra dar viena, horizontali veikla – reikalavimų tvarkymas <Trm61>. Tai nėra tik su reikalavimų analize susijusi veikla. Reikalavimai tvarkomi visą projekto vykdymo laiką, taip pat ir reikalavimų analizės metu. Tai nenutrūkstama ir labai svarbi veikla, nemaža dalimi lemianti viso projekto sėkmę. Išsamiau apie ją pakalbėsime 8 skyriuje. Kitas analizės veiklas aptarsime atitinkamuose šio skyriaus poskyriuose.

Atliekant reikalavimų analizę, visų pirma yra siekiama pašalinti svarbiausius jų trūkumus, gauti tokios kokybės ir tokio detalumo reikalavimus, kad organizacijos vadovybė galėtų pakankamai objektyviai įvertinti projekto kainą, trukmę bei kitų projektui vykdyti reikalingą išteklių apimtis. Antrasis tikslas yra gauti tokius reikalavimus, kad, jeigu jie bus aprobuoti, inžinerinis personalas galėtų pradėti projektuoti, konstruoti bei testuoti sistemą. Reikalavimai yra aprobuojami tik atlikus jų vertinimą. Reikalavimų vertinimas <Trm62> reikalavimų analizei paprastai nėra priskiriamas. Tai savarankiška veikla. Kita vertus, reikalavimų analizei yra priskiriamas architektūros parinkimas ir reikalavimų lokalizavimas, nuleidimas žemyn bei operacionalizavimas, t. y. preliminarinio sistemos projektavimo darbai. Be abejo, nėra prasmės juos pradėti kol reikalavimai nėra galutinai aprobuoti. Iš pirmo žvilgsnio atrodo, kad čia yra prieštaravimas. Iš tiesų taip nėra. Vadovaujantis IEEE ir ACM, rekomendacijomis [111], reikalavimų analizė yra traktuojama kaip žinių sritis, o ne kaip vienas iš reikalavimų inžinerijos stadijos etapų. Užtat šiame skyriuje ir yra kalbama apie viską, kas yra priskiriamas reikalavimų analizei. Kaip nagrinėjamos veiklos bus siejamos su konkretais projekto etapais, yra visiškai kitas klausimas, atsakymas į kurį priklauso nuo to, koks programų sistemos gyvavimo ciklo modelis yra pasirinktas konkretiame projekte.

Taigi, grįžtant prie reikalavimų analizės esmės, reikalavimų analizės metu yra stengiamasi galutinai suprasti surinktus ir suformuluotus analizuojamo lygmens reikalavimus, pašalinti jų prieštaravimus ir kitus trūkumus, parengti reikalavimus derinimui su užsakovu ir kitomis suinteresuotomis šalimis. Kadangi, neturint reikalavimų modelio, galutinai išsiaiškinti reikalavimus ir rasti jų prieštaravimus praktiškai yra beveik neįmanoma, tai reikalavimų analizės metu dar yra atliekamas ir reikalavimų modeliavimas. Be to, parenkama atitinkamo lygmens sistemos architektūra ir atliekamas reikalavimų lokalizavimas bei nuleidimas žemyn, t. y. į žemesnį lygmenį. Kitaip tariant, reikalavimai yra analizuojami pereinant nuo verslo lygmens reikalavimų prie vartotojo lygmens reikalavimų, nuo vartotojo lygmens reikalavimų prie informacinės sistemos lygmens reikalavimų, nuo informacinės sistemos lygmens reikalavimų prie ją palaikančių programų sistemų reikalavimų ir, pagaliau, kiekvienai sistemių, pereinant nuo jos pradinių reikalavimų prie jos projektinių reikalavimų. Vertinimas gali būti atliekamas ir žemesniuose – posistemiu, komponentu – lygmenyse.

5.2 Reikalavimų analizės problemos

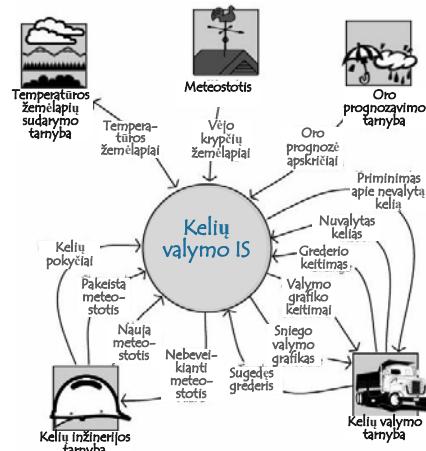
Reikalavimų analizė yra kritinė veikla. Sukurta ne viena metodika, padedanti analizuoti reikalavimus. Visos jos turi gerą instrumentinį palaikymą. Organizacijoje, kaip taisyklė, yra pakankamai daug ekspertų, gebančių daryti reikalavimų analizę. Tačiau praktikoje vis vien gana dažnai susiduriama su rimtomis reikalavimų analizės problemomis. Kodėl taip atsitinka? Pagrindinės priežastys yra dvi. Pirmoji priežastis slypi tame, kad reikalavimų analizei yra skiriama arba per mažai, arba per daug dėmesio. Gerai finansuojuamuose projektuose, stebima per daug didelio dėmesio reikalavimų analizei tendencija. Gerai neišsiaiškinus, ką iš tiesų tikslina ir verta yra analizuoti, pradedama analizuoti viską, ką tik yra įmanoma analizuoti. Vietoj preliminarinio sistemos projektavimo, kurį privalu atlikti reikalavimų analizės metu, iš esmės yra pradedamas išsamus tos sistemos projektavimas. Kadangi šiuo atveju projektavimu užsiima analitikai, beveik visuomet jie tą padaro netinkamai. Be to, analitikų parengtas išsamus, kartais netgi detalaus lygmens projektas, vis dar yra vadinamas kuriamos programų sistemų reikalavimais ir būtent taip yra pateikiamas projektuotojams, reikalaujant tuos „reikalavimus“ įgyvendinti. Šitaip reikalavimai tampa nebesuprantami nei užsakovui, nei projektuotojams, kyla konfliktai, ir apskritai visas projektas yra sujaukiamas. Nepakankamai finansuojuamuose projektuose viskas vyksta atvirkščiai. Kadangi trūksta lėšų, jas dažniausiai bandoma taupyti reikalavimų analizės ir testavimo sąskaita. Skiriant reikalavimų analizei per mažai dėmesio, yra neišsiaiškinami esminiai dalykai, reikalavimai esti neišsamūs bei netikslūs, dėl ko projektuotojai arba pradeda dirbtini analitikų nepabaigtą daryti darbą ir pradeda patys aiškintis reikalavimus su kuriama programų sistema suinteresuotomis šalimis, arba, kas dar blogiau, pradeda fantazuoti, kokia, jų manymu, turėtų būti ta sistema ir suprojektuoja visai ne tai, ko iš tiesų reikia verslui.

Antroji reikalavimų analizės problemų priežastis yra instrumentinių priemonių parinkimas. Netinkamai parinkus instrumentines priemones, jos gali būti nesuderintos su organizacijoje naudojamais projekto vykdymo standartais, dėl ko susidaro įvairios konfliktinės situacijos.. Nors pastaraisiais metais instrumentinių priemonių gamintojai šiai problemai skyrė gana daug dėmesio, ji vis dar nėra išspresta. Jei instrumentinės priemonės primeta reikalavimų inžinerijos arba netgi programų inžinerijos procesą, kuris nėra priderintas prie organizacijoje naudojamų vadybinių procedūrų, paprastai nieko gero iš to nesigauna. Net ir tose organizacijose, kuriose iki to laiko joks programų inžinerijos procesas nebuvo įdiegtas ir programų buvo kuriamas amatiniais metodais. Norint išvengti tokį nesėkmę, pirmiausiai reikia pasirinkti reikalavimų analizės metodiką, išbandyti tą metodiką rankiniu būdu (t. y. nenaudojant jokių instrumentinių priemonių) ir tik po to spręsti, kuriuos tos metodikos etapus yra verta automatizuoti ir kokios instrumentinės priemonės tam galėtų geriausiai tikt. Gali atsitikti ir taip, kad, pabandžius pasirinktają metodiką, paaiškėja, jog ji pati yra netinkama. Tada reikia rinktis kitą metodiką ir bandymus pakartoti. Šitaip pamažu yra tobulinamas organizacijos reikalavimų inžinerijos procesas. Be to, reikia turėti omenyje, kad nepakanka vien tik treniruoti darbuotojus naudotis pasirinkta metodika ir instrumentinėmis priemonėmis. Universitete ne visi jie gavo pakankamai fundamentinių žinių, reikalingų pasirinktajai metodikai suprasti, o jei tokie kursai ir buvo skaitomi, tai ne visi darbuotojai juos iš tiesų įsisavino. Be to, vyresni darbuotojai, jeigu kol kas tokį žinių jiems neprireikė jų praktiniame darbe, beveik viską jau spėjo pamiršti. Todėl, diegiant naujas metodikas ar naujas instrumentines priemones, darbuotojus reikia ne tik treniruoti, bet ir lavinti, organizuoti jiems paskaitas, kuriose būtų pateiktas bent jau svarbiausių teorinių žinių minimumas, būtinas suvokti principams, kuriais yra grindžiama diegiamą metodika ar diegama

instrumentinė sistema. Tarkime, struktūrines metodikas keičiant objektinėmis, būtina supažindinti darbuotojus su objektinės paradigmos pagrindais. Reikalingas ir konkretus mokymas, t. y. kas nors turi pademonstruoti, kaip praktiškai pasinaudoti nauja metodika ar nauja priemone. Taigi, diegiant naujas metodikas ar naujas priemones, visada reikia planuoti darbuotojų lavinimą <Trm4>, mokymą <Trm5> ir treniravimą <Trm6>.

5.3 Konteksto modeliavimas

Konteksto modeliavimas – tai veikla, vykdoma tikslu išsiaiškinti kuriamosios sistemos kontekstą <Trm71> ir sudaryti vadinamąjį konteksto diagramą <Trm29>, parodančią kaip kuriamoji sistema yra susieta su tais jos aplinkos <Trm65> elementais, su kuriais ji keičiasi kokia nors informacija. Konteksto diagrama nustato kuriamosios sistemos ribas (t. y. jos apimtį) ir apibrėžia tos sistemos informacijos mainų interfeisus su jos išorėje esančiomis esybėmis, tokiomis kaip jos naudotojai, jos valdomi ar aptarnaujami įrenginiai, kitos sistemos, iš kurių ji gauna arba kurioms ji pateikia kokią nors informaciją, ir pan. Kaip pavyzdži, panagrinėkime kelių valymo informacinės sistemos konteksto diagramą (81 pav.).



81 pav. Konteksto diagramos pavyzdys (paimta iš [95]).

Šiame pavyzdje sistemos kontekstas apima penkis jos aplinkos elementus: metereologijos stotį, orų prognozavimo tarnybą, kelių valymo tarnybą, kelių inžinerijos tarnybą ir temperatūros žemėlapių sudarymo tarnybą. Kiekvieną parą metereologijos stotys pateikia sistemai vėjo krypčių žemėlapį, temperatūros žemėlapių sudarymo tarnybos – temperatūros žemėlapius, orų prognozavimo tarnyba – oro prognozes pagal apskritis. Metereologijos stočių ir temperatūros žemėlapių sudarymo stočių yra daug, jos veikia automatiškai ir teikia duomenis pagal joms patikėtus regionus. Šias stotis steigia ir prižiūri kelių inžinerijos tarnyba. Ji teikia duomenis sistemai apie visas įsteigtas stotis bei apie esamus kelius ir pokyčius kelių žemėlapyje. Kita vertus, sistema informuoja kelių inžinerijos tarnybą apie tas stotis, kurios nustojo teikti duomenis sistemai. Remdamasi kelių, temperatūros ir vėjų krypčių žemėlapiais bei oro prognozėmis ir duomenimis apie esamą kelių valymo techniką, kiekvieną parą sistema sudaro kelių valymo grafiką ir ji pateikia kelių valymo tarnybai. Ši tarnyba informuoja sistemą apie nuvalytus kelius bei apie jos turimos kelių valymo technikos gedimus ir pokyčius (įsigijo, nurašė arba pardavė). Laiku negavusi pranešimo apie nuvalytą kelią, sistema primena kelių valymo tarnybai apie grafiko pažeidimus. Atsižvelgdama į vėlavimus bei į technikos gedimus, laikas nuo laiko sistema atnaujina grafiką ir vėl ji pateikia kelių valymo tarnybai. Kaip

matome, konteksto diagrama yra labai informatyvi. Jai aprašyti prireikė beveik dvidešimties eilučių teksto.

5.4 Reikalavimų konkretizavimas

Reikalavimai aprašo nagrinėjamo lygmens – verslo, vartotojo, informacinės sistemos, programų sistemas – sistemas ar jos komponento iš išorės stebimą elgseną. Tačiau šią elgseną galima aprašyti skirtingu abstraktumo laipsniu arba, kaip įprasta sakyti, abstrakcijos lygmenyse. Šių lygmenų nereikėtų painioti su reikalavimų lygmenimis. Tai skirtinių dalykai. Specifikavimo abstrakcijos lygmenų esmę paaiškinsime tokiu pavyzdžiu. Tarkime, kuriant judančių objektų valdymo sistemą galima suformuluoti tokį reikalavimą:

„Gavusi duomenis apie objekto judėjimo greitį, sistema privalo pateikti juos displejės ekrane.“

Šis reikalavimas yra suformuluotas tinkamai ir nepažeidžia juodosios dėžės principo. Tačiau, taip pat nepažeidžiant juodosios dėžės principo, jį galima sukonkretinti, suformuluoti žemesniame abstrakcijos lygmenyje, pavyzdžiui, nurodant, kaip duomenys turi būti išdėstyti ekrane bei kokio dydžio ir kokios spalvos šriftas turi būti naudojamas tiems duomenims vaizduoti. Apskritai yra skiriamos konceptinio lygmens, loginio lygmens ir fizinio lygmens specifikacijos, bet gali būti įvesti ir tarpiniai specifikacijų abstrakcijos lymenys.

Pereinant nuo vieno abstrakcijos lygmens prie kito, natūraliai kyla klausimas, kuo skiriasi aukštesnio ir žemesnio lygmens funkciniai reikalavimai. Kartais tokio skirtumo beveik nėra. Pavyzdžiui, liftų valdymo sistemas aukštesnio lygmens reikalavimas gali būti suformuluotas šitaip:

„Esant iškvietimui, reikalaujančiam keisti važiuojančio lifto judėjimo kryptį, kryptis gali būti keičiama tik visiems liftu važiuojantiems žmonėms pasiekus reikiamus aukštus ir iš lifto išlipus“.

Atitinkamas žemesnio lygmens reikalavimas gali būti formuluojamas labai panašiai:

„Esant iškvietimui, reikalaujančiam keisti važiuojančio lifto judėjimo kryptį, sistema privalo ji prisiminti ir pakeisti lifto judėjimo kryptį tik po to, kai bus baigtos vykdyti visos tam liftui jau duotos komandas.“

Šiuo atveju reikalavimas yra tik patikslintas, sistemoje jis įgyvendinamas kaip atskira sistemos funkcija. Tačiau kai kuriais kitais atvejais aukštesnio lygmens reikalavimą galima įgyvendinti keliais skirtingais būdais. Taigi nuleidžiant reikalavimą į žemesnį abstrakcijos lygmenį reikia parinkti vieną iš jų. Tarkime, reikalavimą

„Visi liftai turėtų būti apkrauti daugmaž tolygiai“

galima įgyvendinti keliais skirtingais būdais ir, priklausomai nuo to, koks būdas yra parenkamas, žemesnio lygmens reikalavimai formuluojami visiškai kitaip, pavyzdžiui,

„Esant iškvietimui, sistema turi paveсти ji aptarnauti ilgiausią laiką neveikusiam liftui“.

arba

„Esant iškvietimui, sistema turi paveсти ji aptarnauti arčiausiai nuo iškvietimo taško esančiam liftui“.

Dar kitais atvejais reikalavimui įgyvendinti gali prieikti kelių viena po kitos vykdomų sistemos funkcijų. Aišku, kad tokiais atvejais nuleidžiamą į žemesnį abstrakcijos lygmenį reikalavimą taip pat reikia suskaidyti į kelis. Pavyzdžiui, toks yra reikalavimas

„Praėjus 20 sekundžių po iškiesto lifto durų atidarymo ir nesant tarpduryje jokių kliūčių, lifto durys yra uždaromos ir liftas pradeda judėti į artimiausią iš aukštų, kurių numeriai yra nuspausti lifto valdymo pulte“.

Šie pavyzdžiai dar kartą akivaizdžiai parodo, kad reikalavimo inžinerijos ir projektavimo veiklų griežtai atskirti vieną nuo kitų negalima ir vadovėliuose tai yra daroma tik tam, kad, pritaikius turinių atskyrimo principą, studentų dėmesys laikinai būtų sutelktas į vieną ar kitą programų sistemų inžinerijos proceso aspektą. Šitaip tiesiog paprasčiau organizuoti studijas.

Beje, ne visos reikalavimų analizės metodikos reikalauja, kad reikalavimai būtų sukonkretninti iki sistemos nagrinėamojo lygmens fizinių specifikacijų. Pavyzdžiui, kokybės funkcijų sklaidos metodika (žr. 5.13 poskyri) netgi reikalauja kad verslo lygmens reikalavimai būtų pateikti koncepcinėmis specifikacijomis. Be to, reikia turėti omenyje, kad kiekvieno lygmens reikalavimais aprašoma sistema gali būti specifikuota tiek koncepcinėmis, tiek loginėmis, tiek fizinėmis specifikacijomis. Todėl, pavyzdžiui, galima kalbėti apie informacinės sistemos fizines specifikacijas, apie programų sistemos fizines specifikacijas ir kt.

5.5 Reikalavimų maketavimas

Maketai kuriami tuomet, kuomet vykdytojai, užsakovai ar kitos suinteresuotosios šalys nepajėgia kokių nors reikalavimų suformuluoti kitais būdais. Maketu <Trm30> vadinama bet kuri dalinė, preliminarinė arba potencialiai galima sistemos realizacija. Turint tokią realizaciją galima „pačiupinėti“ reikalavimus ir juos geriau suvokti. Taigi, bent jau kol kas, paprastai yra maketuojami tik kuriamų programų sistemų reikalavimai. Jei į maketavimo procesą pavyksta įtraukti dalykinės srities specialistus ar kitus būsimuosius sistemos naudotojus, tai, aptarinėjant maketą kartu su jais, vykdytojams paprastai pavyksta geriau suprasti uždavinius, kuriuos iš tiesų turėtų spręsti kuriamoji sistema. Taigi, priešingai plačiai paplitusiai nuomonei, maketavimas padeda analizuoti ne tik interfeiso, bet ir funkcinius reikalavimus.

Apskritai, maketai yra kuriami siekiant trijų tikslų:

- geriau suprasti ir papildyti kuriamos programų sistemos reikalavimus,
- projektuojant tą sistemą priimamų sprendimų alternatyvoms nagrinėti,
- sukurti tos sistemos prototipą <Trm73>, t. y. tokį maketą, kuris po to palaipsniui būtų perdaromas į galutinį produktą.

Prototipai kuriami todėl, kad yra dirbama pagal iteracinių arba pagal evoliuciinių programų sistemų gyvavimo ciklo modelį. Dirbant pagal iteracinių gyvavimo ciklo modelį, maketas yra neišbaigta, tarpinė sistemos realizacija. Sukūrus maketą, jis yra išbandomas, realizacija patikslinama ir šitaip gaunamas naujas tobulesnis maketas. Procesas tēsiamas tol, kol nėra sukuriama galutinis sistemos variantas. Šiuo atveju, užsakovui tarpiniai maketai nepateikiami, su jais dirba tik vykdytojai. Dirbant pagal evoliuciinių gyvavimo ciklo modelį, taip pat yra kuriamas vis tobulesnių ir tobulesnių maketų serija. Tačiau šiuo atveju kiekvienas maketas realizuoja tam tikrą sistemos funkcionalumo poaibį, kuris po to, pridedant papildomus sistemos priaugius (angl.

increment), išplečiamas iki galutinio funkcionalumo. Visos tarpinės versijos yra atiduodamos užsakovui ir yra diegiamos. Kol yra kuriama nauja versija, naudotojai dirba su jiems pateikta versija, ją išbando ir galbūt patikslina savo reikalavimus. Taigi, abu šie maketavimo būdai yra susiję su pasirinktu sistemos kūrimo būdu ir tiesiogiai su reikalavimų analize nesusiję. Reikalavimų inžinerijos procesui jie daro tik tokį poveikį, kad reikalavimus reikia versijuoti ir atsižvelgti į tai, kad pereinant nuo vienos versijos prie kitos reikalavimai bus tikslinami ir galbūt keičiami. Kuriant sistemas šitokiais būdais, paprastai yra analizuojami kiekvienos versijos reikalavimai.

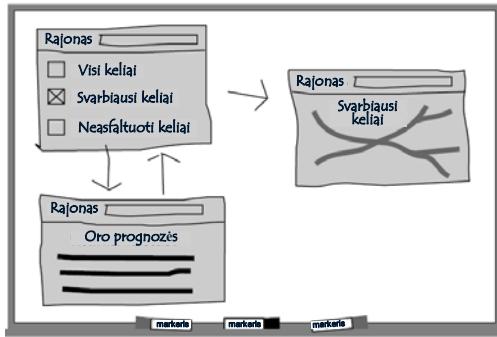
Kiti du tikslai tiesiogiai siejasi su reikalavimų analizę. Tam naudojami išmetamieji maketai <Trm19>. Jei maketas naudojamas kuriamos programų sistemos reikalavimams aiškintis ir papildyti, jis traktuojamas kaip grubi kurio nors nepakankamai gerai perprasto sistemos komponento, dažniausiai kurio nors iš sistemos funkcinių posistemių, realizacija. Kitaip tariant, maketas įgyvendina visus tame komponente lokalizuotus funkcinius reikalavimus, tačiau ignoruoja su jais susijusius nefunkcinius reikalavimus. Jis gali veikti labai lėtai, būti nepatikimas ir nepatogus naudotis. Pasirenkamas pats paprasčiausias ir pigiausias funkcinių reikalavimų realizavimo būdas. Be kita ko, maketas pademonstruoja principinį atitinkamų funkcinių reikalavimų įgyvendinamumą. Naudotojams įvertinus maketo funkcionalumą bei patikslinus ir papildžius jame realizuotus funkcinius reikalavimus, maketas tampa neberekalingas ir yra išmetamas. Tai kainuoja daug pigiau, negu perdarinėti realią sistemą. Panaudoti maketą kaip prototipą šiuo atveju paprastai nepavyksta, nes, norint įgyvendinti nefunkcinius reikalavimus, realią sistemą tenka programuoti kitomis programavimo kalbomis, naudoti joje kitas, sudėtingesnes, DBVS ir apskritai kurti kitaip, negu buvo kuriamas maketas. Tiesa, čia gali paaiškėti, kad, atsižvelgiant į nefunkcinių reikalavimų ribojimus, funkcinių reikalavimų įgyvendinti neįmanoma. Todėl reikalavimų įgyvendinamumo analizei vien maketavimo nepakanka. Tam reikalingi ir kiti metodai.

Naudojant maketą priimamų projektavimo sprendimų alternatyvoms nagrinėti, dažniausiai, bent jau reikalavimų inžinerijos kontekste, analizuojami naudotojo interfeisių reikalavimai. Šiuo atveju maketas paprastai jokio funkcionalumo nerealizuojas, jokių skaičiavimų su juo atliki negalima. Maketams kurti yra naudojamos įvairios priemonės:

- programavimo kalbos Microsoft Visual Basic, IBM VisualAge Smalltalk, Inprise Delphi ir kt.,
- skriptų rašymo kalbos Perl, Python, PHP, Rexx ir kt.,
- žymių kalbos HTML, XML ir kt.,
- kompiuterinės brėžinių braižymo priemonės Microsoft Visio, Microsoft PowerPoint ir kt.,
- specialiai maketams kurti skirti komerciniai paketai, vadinamieji maketų generatoriai (angl. *screen painters, graphical user interface builders*).

Dauguma šiuolaikinių programų sistemų inžinerijai palaikyti skirtų instrumentinių sistemų <Trm28> taip pat turi specialiai maketams kurti pritaikytas priemones.

Naudotojo interfeisių reikalavimams aiškintis naudojami maketai vadinami horizontaliaisiais arba elgsenos maketais <Trm16>. Horizontaliaisiais jie vadinami todėl, kad makteuoja tik vieną sistemos architektūros sluoksnį – naudotojo interfeiso sluoksnį. Maketas makteuoja sistemos interfeiso langus ir leidžia juos keisti, naudotis meniu ir kitomis galimybėmis, tačiau jokio funkcionalumo nerealizuojas. Jis tik imituoja sistemos darbą, pateikdamas atitinkamus pranešimus.

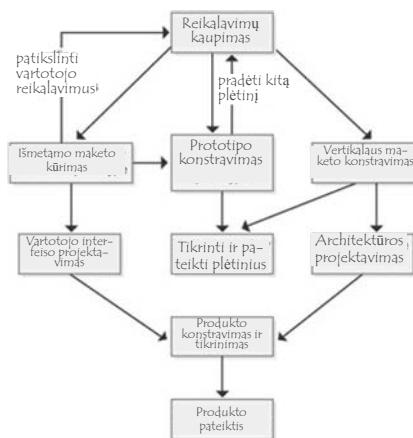


82 pav. Kelių valymo informacinės sistemos reikalavimams aiškintis skirto maketo ant lento pavyzdys (paimta iš [95]).

Neturint specialių makedavimo priemonių arba norint dar labiau atpiginti makedavimo darbus, galima naudoti popierinius maketus (angl. *paper prototype, lo-fi prototype*). Galima išbraižyti interfeiso langus ant popieriaus lapų ir demonstruoti kaip jie bus keičiami priklausomai nuo naudotojo atliekamų veiksmų. Nors tokis makedavimo būdas yra labai primityvus dažnai jo pakanka kai kuriems reikalavimams išsiaiškinti (82 pav.).

Horizontalieji makedai leidžia pademonstruoti naudotojams skirtingais būdais suprojektuotus interfeisus ir jie padeda įvertinti tų interfeisių teikiamas galimybes bei patogumą. Būsimieji sistemos naudotojai, vadovai ir kiti techninio išsilavinimo neturintys asmenys daug geriau gali suprasti, kas jiems yra siūloma, „pačiupinėj“ makedą, negu skaitydami kokį nors techniniai terminais prisotintą tekštą. Net ir tais atvejais, kuomet tą tekštą buvo stengtasi rašyti verslo terminais. Viena vertus, šitaip yra tikslinami ir papildomi naudotojo interfeisių reikalavimai ir, kita vertus, parenkami priimtiniausi tų interfeisių projektavimo sprendimai. Maketas pademonstruoja interfeiso reikalavimų įgyvendinamumą, leidžia įvertinti skirtinges naudotojo sąveikos su sistema būdus, optimizuoti sistemos panaudojamumą ir atliliki kai kuriuos kitus vertinimus. Baigus šį darbą, makedas yra traktuojamas kaip priedas prie programų sistemos reikalavimų specifikacijos. Kaip ir anksčiau aptartu atveju, tokį makedą panaudoti kaip prototipą dažniausiai nepavyksta.

Panašūs makedai gali būti kuriami ne tik interfeiso, bet ir kitų nefunkcinių reikalavimų įgyvendinamumui analizuoti bei šiemis reikalavimams patikslinti ir papildyti. Tačiau tai yra jau ne horizontalieji, bet vertikalieji <Trm93> makedai, nes jie realizuoja tam tikrą visus architektūros sluoksnius apimančią programų sistemas išpjovą. Vertikaliesiems makedams kurti pritaikytų makedavimo priemonių nėra ir todėl tokius makedus tenka kurti rankiniu būdu, o tai gali būti nepigu.



83 pav. Skirtingų makedavimo technikų kombinavimas tarpusavyje.

Kaip parodyta 83 pav., viename projekte gali būti kombinuojamos visos maketavimo technikos. Pateiktame pavyzdyme, sistema yra kuriama evoliuciniu būdu. Kiekvienam sistemos plėtiniui yra kuriami išmetami horizontalusis ir vertikalusis sistemos maketai. Naudojant tuos maketus yra ne tik tikslinami eilinės prototipo versijos (t. y. eilinio sistemos prieaugio) reikalavimai, bet taip pat vykdomas galutinių sistemos interfeisų ir jos architektūros projektavimas.

5.6 Prieštaravimų radimas ir šalinimas

Vienas iš reikalavimų analizės tikslų yra rasti ir pašalinti analizuojamo lygmens reikalavimų prieštaravimus. Reikalavimai pradeda prieštarauti vienas kitam, kuomet dvi sistema suinteresuotos šalys pradeda reikalauti prieštaringuų dalykų. Reikalavimų konfliktas gali būti neakivaizdus, pavyzdžiui, gali vieni kitiems prieštarauti funkciniai ir nefunkciniai reikalavimai [63, 107]. Kilusių prieštaravimų programų sistemų inžinierius dažniausiai pats vienas išspręsti negali ir privalo veikti kaip moderatorius, padedantis prieštarangus reikalavimus keliančioms šalims padėti pasiekti kompromisą. Sandorio požiūriu pageidautina, kad priimti sprendimai būtų susieti trasomis su juos priėmusiais subjektais. Paprastai šios užduotys yra siejamos su reikalavimų analize, nes prieštaravimai yra randami analizuojant reikalavimus, bet gali būti siejamos ir su reikalavimų vertinimu. Be abejo, atliekant reikalavimų analizę yra ieškoma ne tik prieštaringuų reikalavimų. Taip pat yra siekiama nustatyti, ar reikalavimai yra išsamūs, ar jie nėra daugiareikšmiai, ar gerai padarytas turinių atskyrimas, t. y. ar kurie nors reikalavimai nepersikloja, ir ar nėra nerealių, neįgyvendinamų reikalavimų. Apie reikalavimų įgyvendinamumo analizę kalbėsime 5.8 poskyryje. Dabar aptarsime, kaip reikia ieškoti kitų galimų reikalavimų trūkumų.

Visų pirma reikia pabrėžti, kad reikalavimų aiškinimasis ir jų analizė yra labai susipynę. Be abejo, niekuomet nėra dirbama taip, kad pradžioje būtų surenkti ir suformuluojami visi kurio nors lygmens reikalavimai ir tik po to būtų pradedama juos analizuoti ir ieškoti jų trūkumų. Apskritai tai yra ilgas iteracinis procesas, kurį išsamiau aptarsime vėliau. Kol kas pakanka žinoti, jog visa tai vyksta kartu ir kad mes atskirai kalbame apie tai kaip aiškintis, analizuoti ar tvarkyti reikalavimus tiktais dėl to, kad galėtume sutelkti dėmesį į kurį nors vieną klausimą ir jį išsamiai išsiaiškinti.

Toliau reikia pabrėžti, kad reikalavimų analizė, išskaitant ir aukščiau išvardintų trūkumų paiešką, nėra paprastas, lengvai formalizuojamas procesas. Tiktai dalis trūkumų atsiranda dėl techninių priežasčių, t. y. dėl to, kad kas nors ko nors nesuprato, apsižioplino ar pan. Kitų trūkumų priežastys slypi pačiame versle arba jo organizacinėje aplinkoje. Nei viena sudėtingesnė verslo sistema nėra iki galo sutvarkyta, apvalyta nuo prieštaravimų ir dviprasmybių. Verslo taisyklių sistemos dažnai esti neišsamios arba, kitaip tariant, ten yra daugybė neišspręstų klausimų ir, susidarius atitinkamoms situacijoms, niekas nežino, kaip reikia spręsti staiga iškilusią problemą. Jos sprendimo ieškoma pasitarimuose, kuriant kokias nors komisijas ar darbo grupes. Be abejo, tokios situacijos nėra dažnos. Juo brandesnė verslo sistema, tuo mažiau problemų joje slypi, tačiau tam tikras neišspręstų problemų kiekis lieka visuomet. Visos jos vienaip ar kitaip yra perkeliamos į reikalavimus. Todėl analizuojant reikalavimus, ypač verslo, vartotojo ar informacinių sistemos lygmens reikalavimus, iš tiesų vyksta ne tik pačių reikalavimų, bet ir esamų verslo bei informacijos apdorojimo taisyklių sistemų analizė. Kompiuterizuotos sistemos kuriamos ne tam, kad išaldyti esamas verslo taisykles, o tam, kad tobulinti esamą verslą, keičiant, kartais netgi kardinaliai keičiant, esamas verslo taisykles. Todėl, mano nuomone, daugelio autorių duodamos rekomendacijos, kad sistemos kūrimą reikia pradėti ne nuo verslo problemų, o nuo verslo taisyklių aiškinimosi ir kad

sistemos privalo veikti sutinkamai su esamomis verslo taisyklėmis, yra gerokai pasenusios ir nepagrįstos. Tačiau net ir tais atvejais, kuomet einama tokiu keliu, vis vien yra susiduriama su verslo taisyklių sistemos prieštaragingumo, neišsamumo bei dviprasmybių problemomis ir tas problemas tenka vienaip ar kitaip spręsti. To negalima padaryti paprastai, remiantis vien loginiais ir techniniais argumentais. Čia visuomet iškyla politiniai, organizacijos verslo politikos prasme, ir organizaciniai argumentai. Be to didelę reikšmę turi sprendimus priimančių asmenų nuostatos, nuomonės, jų užimamos pareigos, turimas autoritetas, kompetencija, o kartais net ir charakterio savybės. Dėl visų šių priežasčių reikalavimų analizės formalizuoti nepavyksta, todėl žemiau aprašomi reikalavimų trūkumų paieškos būdai turi būti traktuojami daugiau kaip tam tikri principai, kuriais reikia vadovautis, o ne kaip visiškai susistemintas, struktūruotas ir formalizuotas procesas.

Svarbiausiai reikalavimų trūkumų paieškos metodai yra du. Tai analizės klausimynų (angl. *checklists*) naudojimas ir sąveikos matricos (angl. *interaction matrices*). Abu jie priskiriami prie pasiteisinusių reikalavimų inžinerijos metodų [107] ir turėtų būti naudojami bet kuriam reikalavimų inžinerijos procese.

Analizės klausimynų naudojimas padeda bent šiek tiek sutvarkyti reikalavimų trūkumų paieškos procedūras ir sumažina tikimybę, kad kas nors ką nors pražiopsos, nes ką reikia analizuoti yra apgalvojama iš anksto ir fiksuojama klausimyne. Taip pat sumažėja tikimybė palikti neišanalizuotus kokius nors reikalavimus, nes yra reikalaujama, kad kiekvieno reikalavimo analizės rezultatai būtų fiksuoti raštu, pateikiant atsakymus į visus klausimyno klausimus. Dar vienas klausimyno pliusas, yra galimybė panaudoti jį vėlesniuose projektuose, nes jis mažai priklauso nuo konkretaus projekto pobūdžio ir paprastai kiekviename naujame projekte dar yra šiek tiek patobulinamas. Taigi po 3-4 projektų jau galima turėti visai neblogą klausimyną. Aišku, pakartotinai pasinaudoti klausimynu galima tiktais tuomet, kuomet klausimai Jame formuluojami pakankamai bendrai ir nėra specifiniai. Apskritai klausimyną rekomenduojama pradėti kurti vadovaujantis 4 lentelėje pateiktu pradiniu variantu.

4 lentelė. Pradinis reikalavimų analizės klausimynas.

Klausimas	Paaškinimas
Ar nepažeistas reikalavimo abstraktumas?	Reikalavimas vadinamas abstrakčiu, jei jis suformuluotas vadovaujantis juodosios dėžės principu, t. y. nusako iš išorės stebimą funkcinę arba nefunkcinę sistemos savybę ir nieko nekalba apie tai, kaip tą savybę įgyvendinti sistemoje.
Ar reikalavimas yra elementarus?	Reikalavimas vadinamas elementariu, jeigu jo negalima suskaidyti į kelis paprastesnius reikalavimus.
Ar reikalavimas nėra perteklinis?	Reikalavimas vadinamas pertekliniu, jei tame kartojama kituose reikalavimuose pateikta informacija arba jei negalima pasakyti, kokie verslo poreikiai bus patenkinti jį įgyvendinus.
Ar reikalavimas nėra daugiaprasmis?	Reikalavimas vadinamas vienareikšmiu, jei jo negalima interpretuoti (suprasti) keliais skirtingais būdais. Tai, be abejo, nereiškia, kad jį galima įgyvendinti tik vienu būdu.

Klausimas	Paaškinimas
Ar reikalavimas yra patikrinamas?	Reikalavimas vadinamas patikrinamu, jei galima pasakyti, kaip (testu, stebint sistemos veikimą, analizuojant sistemos darbo rezultatus ar kaip nors kitaip) bus galima patikrinti, ar reikalavimas tikrai yra įgyvendintas. Be to, tikrinimo būdas turi būti realiai įgyvendinamas, t. y. jis turi nebūti per daug brangus, neužtruktū per daug ilgai ir nereikalauti kokios nors išskirtinės tikrintojo kompetencijos ar kokių nors ypatingų techninių ar programinių priemonių.
Ar reikalavimas yra išsamus?	Reikalavimas vadinamas išsamiu, jei tame pasakyta viskas, kas turėtų būti pasakyta, ir ji skaitant nekyla jokių klausimų. Išsamus reikalavimas turi būti suprantamas ne tik tuomet, kuomet jis yra nagrinėjamas kartu su kitais reikalavimais, bet ir tuomet, kuomet jis nagrinėjamas atskirai.
Ar reikalavimas yra tikslus?	Reikalavimas vadinamas tiksliu, jeigu visi tame vartojami terminai yra apibrėžti ir jeigu tame nėra vartojama kokių nors netikslų posakių (pvz., <u>maždaug</u> , <u>beveik</u> , <u>apytiksliai</u> , turi būti <u>patogus</u> , naudoti <u>nedaug atminties</u> , <u>veikti greitai</u> ir pan.).
Ar reikalavimas yra glaustas?	Reikalavimas vadinamas glaustu, jei tame nėra patekta jo pagrindimo, kokių nors apibrėžčių ar kitų nebūtinų dalykų.
Ar reikalavimas parašytas suprantamai visoms suinteresuotosioms šalima?	Reikalavimas vadinamas suprantamu, jei jis parašytas nevartojant kokių nors specialių, duomenų žodyne neapibrėžtų terminų ir tame yra aiškiai pasakyta, kokią funkcinę ar nefunkcinę savybę privalo turėti sistema.

Reikalavimų analizė yra atliekama siekiant rasti ir pašalinti reikalavimų trūkumus. Ji nėra kokybės valdymo proceso dalis. Todėl klausimyne neturi būti klausimų, susijusių su kokybės valdymu. Be to, klausimyno nereikėtų labai išpūsti, tame neturėtų būti daugiau kaip 15 klausimų, nes kitaip beveik niekas tų klausimų neprisimins mintinai ir reikalavimų analizė labai pasunkės. Be to, vargu ar galima sudaryti ilgesnį klausimyną, nededant ten klausimų, susijusių tik su tam tikro pobūdžio reikalavimais. Esant reikalui tikrinti vienas ar kitas reikalavimų grupes papildomai, tam reikia parengti papildomus klausimynus. Jais turėtų naudotis kiti tikrintojai. Vienam ir tam pačiam asmeniui sunku dirbti iš karto su keliais klausimynais, o skaityti dokumentą kelis kartus (dirbant su kiekvienu klausimynu) būtų labai nuobodu. Ir vienu ir kitu atveju analizės kokybė greičiausiai nukentėtų.

Darbui su klausimynais palengvinti, patariama naudotis kompiuterinėmis lentelėmis. Lentelės eilutės, pavyzdžiui, gali atitikti analizuojamus reikalavimus, o stulpeliai – klausimyno klausimus. Atitinkamuose langeliuose dedami atsakymai į klausimyno klausimus. Toks darbo būdas leidžia lengvai papildyti lentelę naujais reikalavimais ar klausimais, matyt, kas jau buvo patikrinta, o kas dar ne, perduoti darbą iš vieno asmens kitam.

Be abejo, klausimynų naudojimas turi ne tik privalumų, bet ir trūkumų. Didžiausias trūkumas yra tas, kad dirbdami su klausimynu analitikai pradeda dirbtį mechaniskai. Jie tik atsakinėja į klausimyne surašytus klausimus ir lengvai praleidžia tokius reikalavimų trūkumus, kurių paieškos klausimynas nenumatė.

Reikalavimų sąveikos matrica vadinama matrica, kurioje ir stulpeliai, ir eilutės atitinka analizuojamus reikalavimus. Matricos langeliuose žymima ar tame langelyje susikertantys reikalavimai yra nepriklausomi, persiklojantys arba prieštaringi. Taigi sąveikos matrica papildo klausimynus ir padeda rasti tokius reikalavimų trūkumus, kurių negalima rasti analizuojant reikalavimus po vieną. Beje, tokia matrica yra naudinga ne tik analizuojant reikalavimus, bet ir derantis su suinteresuotomis šalimis dėl reikalavimų aprobabimo, nes ji labai akivaizdžiai parodo, kur reikalavimuose, o greičiausiai taip pat ir atitinkamose verslo taisyklėse, slypi nedarnos problemos. Pats paprasčiausias būdas pasidaryti sąveikos matricą yra panaudoti tam kompiuterinę lentelę. Tokia lentelė, kaip ir klausimyno atveju patogi yra tuo, kad ją lengva plėsti, pridedant naujus reikalavimus. Somerville ir Sawyer rekomenduoja [107] prieštaringus reikalavimus žymeti lentelėje reikšme 1, persiklojančius reikalavimus – 1000 ir nepriklausomus – 0. Šitaip parinkus reikšmes ir susumavus jas pagal eilutes ir pagal stulpelius, liekana gauta sumą dalinant iš 1000 parodo kiek tame stulpelyje ar toje eilutėje yra konfliktuojančių reikalavimų, o dalybos rezultatas – kiek yra persiklojančių reikalavimų. Kadangi ir matricos stulpeliai, ir jos eilutės atitinka reikalavimus, tai gautieji skaičiai parodo, kurie reikalavimai kelia daugiausiai problemų ir todėl turi būti analizuojami kruopščiausiai.

Reikalavimų sąveikos matricos turi vieną esminį trūkumą. Jas galima sukonstruoti tik sistemoms, turinčioms palyginti nedaug, nedaugiau 200 reikalavimų. Esant daugiau reikalavimų, juos reikia vienaip ar kitaip skaidyti į nepriklausomas susijusią reikalavimų grupes ir kiekvienai tokiai grupei konstruoti savą sąveikos matricą. Tačiau šitaip suskaidyti reikalavimus pavyksta toli gražu ne visuomet.

Pažymėtina, kad reikalavimų sąveikos matricos turi daug bendrybių su kokybės funkcijų sklaida, aptariama 5.13 poskyryje.

5.7 Skirtingų lygmenų reikalavimų darnos analizė

Projekto reikalavimai, tame projekte kuriamos informacinių sistemos reikalavimai ir tos sistemos programinės įrangos reikalavimai turi būti tarpusavyje suderinti. Informacinių sistemos reikalavimus ir jos programinės įrangos reikalavimus suderinti vienus su kitais yra palyginti paprasta, nes programinės įrangos reikalavimai yra išvedami iš informacinių sistemos reikalavimų. Tačiau ir čia susiduriamo su tam tikromis problemomis, nes gali paaiškėti, kad kai kurie šitaip gauti programų sistemos reikalavimai yra arba neigvendinami, arba jų įgyvendinimas kainuoja labai brangiai. Todėl, suformulavus programų sistemos reikalavimus ir atlikus jų įgyvendinamumo analizę, gali tekti grįžti atgal ir formuliuoti kai kuriuos informacinių sistemos reikalavimus ir netgi kai kuriuos vartotojo ar verslo reikalavimus. Projekto ir Jame kuriamo produkto reikalavimus suderinti yra gerokai sunkiau. Gali konfliktuoti projekto reikalavimais numatytu testavimo apimčių ir kuriamos programų sistemos patikimumo reikalavimai, o ir kiti produkto kokybės reikalavimai gali konfliktuoti su

projekto reikalavimais numatytomis finansavimo apimtimis arba su tais reikalavimais numatytais projekto vykdymo terminais. Galimos ir kitos projekto ir produkto reikalavimų nedarnos priežastys. Atrasti visus tokios nedarnos atvejus ir juos pašalinti yra labai sunku. Kita vertus, projekto vykdymo eigoje jie išaiškėja savaime, bet tada jau gali būti per vėlai juos pašalinti ir projektas gali baigtis visiška nesėkmė. Išsamiau skirtinį lygmenį reikalavimų darnos klausimus aptarsime 5.13 poskyryje.

5.8 Reikalavimų įgyvendinamumo analizė

Reikalavimų įgyvendinamumo analizė <Trm49> iš esmės yra rizikos veiksnių analizė. Ji atliekama siekiant įsitikinti, kad suformuluoti reikalavimai yra realistiški arba, kitaip tariant, kad sistemą tikrai galima sukurti. Be abejo, turima omenyje ne tai, kad tokią sistemą apskritai įmanoma sukurti, o tai, kad tą gali padaryti esami specialistai per užplanuotą laiką ir su užplanuotais pinigais. Be to, reikia ne tik parodyti reikalavimų įgyvendinamumą, bet ir išnagrinėti galimus jų įgyvendinimo būdus, nuspręsti, kokiais kriterijais tikslina vadovautis pasirenkant vieną iš tų būdų ir, pagaliau, tokį būdą parinkti. Ši veikla yra labai svarbi, kritinė visam projektui.

Reikia skirti reikalavimų įgyvendinamumo analizę ir sistemos įgyvendinamumo analizę. Pastaroji parodo, ar sistemą apskritai yra verta kurti ir išryškina tikrąjų tos sistemos vertę verslui. Tiktai po to užsakovas gali galutinai apsispręsti, ar projektą apskritai verta testi. Kadangi ne visi užsakovai šitą supranta, tai kartais sistemos įgyvendinamumo analizę apskritai nėra daroma. Užsakovas pasirašo sandorį su vykdytoju, kuriuo pastarasis įsipareigoja sukurti sistemą, nors jokie tos sistemos reikalavimai apskritai dar nėra išsiaiškinti ir suformuluoti. Tik po to vykdytojas pradeda aiškinti ir rinkti reikalavimus. Jis atmata tuos reikalavimus, kurių nepajégia įgyvendinti ir kuria sistemą, nesukdamas sau galvos ar užsakovui ji tikrai atsipirkis. Iš tiesų turėtų būti dirbama kitaip. Pirmiausiai užsakovas turėtų sudaryti sandorį sistemos reikalavimų specifikacijai parengti ir reikalavimų įgyvendinamumui išanalizuoti. Jei analizės rezultatai yra teigiami, sandoris yra pratešiamas arba galbūt skelbiamas konkursas naujam vykdytojui parinkti. Aišku, tokia projekto vykdymo schema taip pat nėra ideali, nes, kaip jau ne kartą matėme, reikalavimų inžinerijos ir sistemos projektavimo procesai yra labai susipynę. Todėl šiuo požiūriu geriausias projekto vykdymo būdas yra viską daryti savo jėgomis, t. y. turėti savą IT padalinį, kuris rengtų reikalavimų specifikaciją, atliktų įgyvendinamumo analizę ir kurtų sistemą. Tačiau toks būdas irgi turi savų trūkumų: sunku sukomplektuoti pakankamai aukštostos kvalifikacijos specialistus; neskelbiant konkurso, projekto savikaina dažnai išauga; IT padalinui reikalingos darbo vietas ir jų aptarnavimas ir kt. Be to, šiuo metu vyrauja nuostata pirkti rinkoje visas paslaugas, kurias tik yra įmanoma nupirkti. Taigi, kaip ir daugeliu kitų atveju, idealaus sprendimo čia nėra ir tenka rinktis konkrečiam atvejui geriausią iš apskritai blogų sprendimų.

Apskritai, atliekant sistemos įgyvendinamumo analizę bandoma atsakyti į šiuos klausimus:

- ar apskritai yra žinoma, kaip galima įgyvendinti reikalavimus?
- ar vykdytojų kolektyvo turimų žinių ir mokėjimų tam pakanka?
- ar projektui skirtų pinigų tam pakanka?
- ar projektui skirto laiko tam pakanka?
- ar nėra tam nepašalinamų juridinių ar kitokio pobūdžio kliūčių?
- ar įgyvendinus reikalavimus verslas tikrai gaus iš to konkrečią apčiuopiamą naudą?

Sistemos įgyvendinamumo analizė išsamiai aptarinėjama bakalauro studijų pakopos studentams skirtame programų sistemų inžinerijos kurse. Čia tik priminsime trumpai jos esmę. Visų pirma priminsime, kad sistemos įgyvendinamumas yra analizuojamas penkiais svarbiausiais aspektais:

- Operaciniu: Ar užsakovas pajęgus eksplloatuoti sukurtą sistemą? Ar numatomas sistemos naudojimo scenarijus tikrai veiks? Ar dalykinės srities specialistai suinteresuoti laikytis scenarijumi nustatyty darbo taisylių? Atliekant operacinių sistemos įgyvendinamumą, analizuojamos įvairios tą sutrukdyti galinčios kliūtys, tokios kaip, pavyzdžiu, darbo su klaviatūra apimtys, kompiuterių baimė, tradicijos, korporacinė kultūra ir pan.
- Techniniu: Ar žinoma problemos sprendimo teorija ir ar yra ją palaikanti technologija? Ar vykdytojai pajęgūs sukurti sistemą?
- Ekonominiu: Ar projektas atsipirk? Per kiek laiko ir ar apskritai grįš į sistemą įdėtos investicijos?
- Plano: Ar galima su turimais vykdytojais ir kitais turimais ištakliais baigti projektą laiku?
- Teisinii etiniu: Ar projektas nepažeis kokių nors galiojančių teisės ar kokių nors pripažintų etinių normų?

Kaip matome, dauguma čia nagrinėjamų klausimų su reikalavimų analize turi nedaug ką bendro. Reikalavimų įgyvendinamumo analizė yra tik gana nedidelė visos sistemos įgyvendinamumo analizės dalis. Ji apima tik dalį sistemos įgyvendinamumo techninio ir plano aspektų klausimų. Be to, sistemos įgyvendinamumo analizė yra daroma iš užsakovo požiūrio taško. Pagal jos rezultatus užsakovas sprendžia, ar jam verta pradėti projektą. Reikalavimų įgyvendinamumo analizė yra daroma iš vykdytojo požiūrio taško. Pagal jos rezultatus vykdytojas sprendžia, ar jam verta imtis projekto, ar realūs yra projekto vykdymo terminai ir jo finansavimas ir kaip maksimaliai apsaugoti nuo rizikos veiksnių keliamų grėsmių.

Galima išskirti dešimt svarbiausių analizuotinų reikalavimo įgyvendinamumo rizikos veiksnių:

- reikalavimo kintamumą,
- poveikį sistemos našumui,
- poveikį sistemos patikimumui,
- poveikį sistemos naudojimo saugumui ir jos apsaugai nuo nesankcionuoto panaudojimo,
- pokyčius, kuriuos gali tekti daryti šiuo metu naudojamame programų sistemų inžinerijos procese,
- būtinybę naudoti vykdytojams neįprastas technologijas,
- būtinybę dirbt su nestandardiniais duomenimis,
- galimybę pažeisti projekto vykdymo terminus,
- būtinybę samdytis subvykdytojus,
- priklausomybę nuo trečiųjų šalių.

Atliekant reikalavimų įgyvendinamumo analizę, kiekvieną reikalavimą reikia analizuoti visų išvardintų rizikos veiksnių požiūriu. Be abejų, reikia taip pat įvertinti to reikalavimo įgyvendinimo kaštus ir jo įgyvendinimo duodamą naudą verslui. Tačiau apie tai kalbėsime 5.12 poskyryje.

Analizuojant prognozuojamą reikalavimo kintamumą, sprendžiama kiek yra tikėtina, kad jis pakankamai ilgai nekis, išliks stabilus. Nuo to priklauso reikalavimo įgyvendinimo būdas. Stabilūs reikalavimai turi būti įgyvendinami labai kruopščiai,

siekiant maksimalaus realizacijos efektyvumo. Nestabilius reikalavimus geriausiai iš viso atmesti, o jei to negalima padaryti, juos reikia taip realizuoti, kad realizaciją būtų galima paprastai ir greitai pakeisti.

Analizuojant reikalavimo poveikį sistemos našumui, patikimumui, naudojimo saugumui bei apsaugai nuo nesankcionuoto naudojimo, aiškinamasi ar reikalavimo įgyvendinimas nesukels tokio pobūdžio problemų. Kilus problemoms, gali paaiškėti, kad reikalavimas iš tiesų konfliktuoja su vienais ar kitais nefunkciniais reikalavimais, ir tekti atitinkamai papildyti reikalavimų sąveikos matricą. Be abejo, išsiaiškinti tokius dalykus nėra paprasta. Tam gali prireikti kurti atitinkamus vertikaliuosius maketus ir atlirkti su jais bandymus.

Gali paaiškėti, kad reikalavimo neįmanoma įgyvendinti nekeičiant šiuo metu naudojamo programų sistemų inžinerijos proceso, pavyzdžiui, gali prireikti vartoti formaliašias specifikavimo kalbas ar matematiškai įrodinėti kokias nors sistemos saugumo ar patikimumo savybes. Aišku, kad įdiegti tokius pokyčius į standartinį programų sistemų inžinerijos procesą nėra paprasta. Tam reikia ne tik papildomų lėšų ir papildomo laiko, bet ir atitinkamos kvalifikacijos specialistų, kurių apskritai gali būti neįmanoma pasisamdyti. Panašiai yra ir tais atvejais, kuomet paaiškėja, kad vykdytojai neturi reikalavimui įgyvendinti reikalingų gebėjimų ir teks tam darbui ieškoti subvykdytojų. Aišku, kad subvykdytojų paieška bei tolimesnis darbas su jais reikalauja papildomų darbo ir laiko sąnaudų, o tuo pačiu ir papildomų lėšų. I visą tai turi būti atsižvelgta planuojant projekto vykdymo terminus ir jo finansavimą. Dar blogiau yra tuomet, kuomet galimybė įgyvendinti reikalavimą priklauso nuo kokios nors nepriklausomos trečiosios šalies, pavyzdžiui, nuo kokio nors techninės ar programinės įrangos tiekėjo. Tokiais atvejais kaip nors sumažinti riziką apskritai yra neįmanoma. Tiesiog reikia pabandyti įvertinti rizikos dydį ir spręsti, ar apskritai verta imtis tokio projekto.

Analizuojant reikalavimą gali paaiškėti, kad jam įgyvendinti reikia panaudoti kokias nors vykdytojams neįprastas technologijas, pavyzdžiui, ekspertinę sistemą, ar kokius nors modernius duomenų analizės metodus. Tokiais atvejais kylančios grėsmės yra labai panašios į tas, kurios kyla keičiant standartinį programų sistemų inžinerijos procesą, ir nuo jų reikia saugotis taip pat panašiai. Panašiai yra ir tais atvejais, kuomet paaiškėja, kad teks dirbti su geografine informacija, audiovizualine informacija ar kokiai nors kitokiai nestandardiniai duomenimis. Šiuo atveju teks įsigyti, įdiegti ir įsisavinti nestandardines duomenų tvarkymo priemones ir aišku, kad tam taip pat prireiks papildomo laiko ir papildomų lėšų.

Taigi reikalavimų įgyvendinamumo analizė yra sudėtingas ir komplikuotas procesas. Jai atlirkti reikalingi patyrę, išmanantys ir kūrybingi specialistai. Kol kas jokių standartinių receptų tam sukurta nėra. Tiktais yra patariama [107] nesistengti įvertinti riziką labai tiksliai. Tai padaryti yra neįmanoma. Pakanka ją įvertinti kokybiškai, pavyzdžiui, skalėje „nėra“, „nedidelė“, „vidutinė“, „didelė“, „labai didelė“.

5.9 Koncepcinis reikalavimų modeliavimas

Programavimo eros pradžioje, Lietuvoje tai buvo apie 1965-1968 metus, buvo dirbama pagal žodines reikalavimų specifikacijas, palaikant glaudžius kontaktus su užsakovais ir skubiai išsiaiškinant visus kilusius neaiškumus. Tačiau jau apie 1970 metus visos rimtesnius projektus vykdančios organizacijos, tuo metu tai buvo Vilniaus skaičiavimo mašinų gamykla ir dabartinių Kauno technologijos universiteto, Matematikos ir informatikos instituto ir Vilniaus universiteto skaičiavimo centrai, perėjo prie darbo pagal rašytines specifikacijas. Vakarų šalyse, visų pirma JAV, tai

įvyko beveik dešimtmečiu anksčiau. Specifikacijos tuo metu buvo rašomos kokia nors natūraliaja kalba, Lietuvoje – rusų arba lietuvių kalba. Tačiau, laikui bėgant, specifikacijas pradėta struktūruoti (t. y. rašyti pagal tam tikrus šablonus), o vėliau ir formalizuoti. Formalizuota specifikacija iš tiesų yra koncepcinis kuriamos programų sistemos modelis, aprašantis tą sistemą kaip juodąją dėžę <Trm23>. Nereikėtų painioti koncepcinio verslo sistemos ar dalykinės srities modelio ir koncepcinio kuriamos programų sistemos reikalavimų modelio. Nors tai yra ir tampriai susiję, tačiau skirtinių dalykai, naudojami skirtiniams tikslams pasiekti.

Koncepcinis verslo sistemos ar koncepcinės srities modeliavimas yra vienas iš svarbiausių programų sistemų reikalavimų aiškinimosi metodų. Tokie modeliai yra kuriami tam, kad būtų galima geriau suprasti kuriamos sistemos probleminę sritį, o ne tam, kad būtų pradėta tą sistemą projektuoti. Yra keli standartai, aprašantys tokiam modeliavimui skirtas modeliavimo kalbas. Funkciniam modeliavimui skirtas kalbas aprašo standartai IEEE Std 1320.1 [ST10] ir IDEF0 [ST10], informaciniam modeliavimui skirtas modeliavimo kalbas – standartai IEEE Std 1320.2 [ST11] ir IDEF1X97 (IDEFOobject) [ST11]. Tačiau šiandieninėje praktikoje tam tikslui dažniau yra naudojamos tokios modeliavimo kalbos kaip UML [91] arba įvairios esybių ryšių diagramos [13], naudojamos kartu su jas papildančiomis duomenų srautų [25], būsenų [44] ir gal būt kitomis diagramomis. Kuriami ir mažiau formalūs modeliai, pavyzdžiui, struktūruoti verslo užduočių aprašai [17, 66]. Vienas iš galimų koncepcinio reikalavimų modeliavimo būdų buvo pademonstruotas 4.4 poskyryje.

Koncepciniai programų sistemų reikalavimų modeliai yra kuriami dėl dviejų tikslų. Reikalavimų analizės metu tokie modeliai yra naudojami reikalavimų išsamumui ir neprieštaragingumui tikrinti. Kita vertus, tai yra kuriamų programų sistemų aukščiausio, koncepcinio, lygmens projektais, kurie, juos detalizuojant, sukonkretinant ir papildant, yra pertvarkomi į detaliuosius tų sistemų projektus.

5.10 Duomenų žodyno sudarymas

Svarbiausioji duomenų žodyno paskirtis yra suvienodinti dalykinės srities terminiją. Dažnai skirtinės suinteresuotosios šalys tiems patiemis dalykams aprašyti vartoja skirtinės terminus. Esti ir atvirkščiai, t. y. tam pačiam terminui yra suteikiamas skirtinės prasmės. Neretai netgi užsakovo organizacijos skirtinėse padaliniuose dirbantys specialistai vartoja šiek tiek skirtinės terminus. Duomenų žodynai padeda išvengti nesusikalbėjimo problemų. Sudarius žodyną, visi reikalavimai yra formuluojami žodyno terminais. Po to, tie patys terminai yra vartojuami projektinėje dokumentacijoje ir netgi programų tekstuose. Duomenų žodynai yra automatizuoti, saugomi kompiuteriniu pavidalu ir importuojami į rengiamus dokumentus bei rašomas programas. Kai kurie iš jų patys kaupia visus naujus terminus, kuriuos kas nors pavartoja kokiame nors dokumente ar kokioje nors programe. Tai leidžia už projekto informaciją aprūpinimą atsakingiems darbuotojams laiku pastebeti visus nuokrypius nuo projekte oficialiai aprobuotos terminijos. Reikia pasakyti, kad duomenų žodynai naudingi ne tik dirbant su reikalavimais, bei projektuojant ir rašant programas. Juose apibrežti terminai yra vartojuami ir vartotojo interfeisiuose, vartotojui skirtose dokumentacijoje, įskaitant vadinamuosius pagalbos failus, bei sistemos spausdinamuose pranešimuose. Pastaraisiais metais duomenų žodynus sparčiai keičia dalykinės srities ontologijos. Nuo duomenų žodynų jos skiriasi tuo, kad yra labiau formalios ir kad jose apibrėžiami ne tik atitinkamomis duomenų struktūromis vaizduojamos dalykinės srities esybės, bet ir kiti dalykinės srities terminai.

Paprastai duomenų žodyne duodamos visų su sistema susijusių dalykinės srities esybių ir jas sistemoje modeliuojančių duomenų struktūrų apibrėžtys. Kiekvienai esybei aprašomi:

- jos pavadinimas, žymuo,
- prasmė,
- tipas (t. y. procesas, objektas, atributas ir pan.),
- ribojimai, išskaitant jai vaizduoti naudojamą duomenų ženkliškumą, formatus, vaizdavimo tikslumą, leistinas reikšmes ir kt.
- ryšiai su kitais žodyne apibrėžtais terminais.

Duomenų žodynas susieja tarpusavyje skirtingus reikalavimų vaizdavimo būdus (reikalavimų specifikacija, koncepcinis modelis ir kt.), nes visuose juose duomenys ir duomenų struktūros yra vadinami tais pačiais vardais.

Daugelis darbui su reikalavimais skirtų instrumentinių sistemų (žr. 8.8 poskyri) bei didžioji dauguma programų sistemų inžinerijos palaikymo sistemų <Trm28> turi duomenų žodynams kurti ir jais naudotis skirtas priemonės. Taip pat gana dažnai duomenų žodynai yra kuriami panaudojant reliacinių duomenų bazės valdymo sistemas. Kai kurios iš jų taip pat turi priemones duomenų žodynams kurti ir dirbti su jais.

5.11 Architektūros parinkimas

Tam tikru programų sistemos inžinerijos proceso momentu iškyla reikalas pasiūlyti kuriamos informacinės sistemos, o vėliau ir tą sistemą palaikančią programų sistemų architektūras. Nenusprendus, kokia bus IS architektūra, neįmanoma nuspręsti, kiek programų sistemų reikia sukurti tai IS palaikyti ir suformuluoti tų sistemų reikalavimus. Kita vertus, architektūros parinkimas neabejotinai yra ne reikalavimų inžinerijos, o projektavimo veikla ir todėl tarsi neturėtų būti nagrinėjama, kalbant apie reikalavimų inžinerijos problemas. Be abejo, būtų galima taip ir padaryti, t. y. griežtai vadovautis turinių atskyrimo principu ir apie architektūrų parinkimą kalbėti tik informacinių sistemų ir programų sistemų projektavimo kursuose. Tačiau taip būtų nepatogu, nes iš tiesų projektavimo ir reikalavimų inžinerijos veiklos susipina viena su kita ir yra neįmanoma tas veiklas griežtai atskirti viena nuo kitos. Nors architektūros parinkimas priskirtinas projektavimo procesui, ši veikla įsiterpia į reikalavimų inžinerijos procesą ir, nutylint apie tai, kai kurie reikalavimų inžinerijos proceso momentai liktų neaiškūs.

Beje, programų sistemų reikalavimų negalima galutinai suformuluoti ne tik neparinkus jos palaikomas IS architektūros, bet ir neparinkus pačios programų sistemos architektūros. Programų sistemos architektūra daro poveikį tos sistemos patikimumui, našumui, prižiūrumui ir galimybėms keisti tos sistemos mastą <Trm13>. O tai atsiliepia visoms sistemos savybėms. Kitaip tariant, neparinkus programų sistemos architektūros, iš tiesų negalima atliki nei programų sistemos, nei visos IS reikalavimų įgyvendinamumo analizės. Gali paaiškėti, kad tiesiog nėra tokų architektūrinių sprendimų, kurie leistų įgyvendinti norimas sistemos nefunkcines savybes. Beje, su analogiška problema susiduriama ir tuomet, kuomet kuri nors programų sistema yra ne kuriama, bet įsigijama rinkoje kaip gatavas produktas. Šiuo atveju gali paaiškėti, kad rinkoje nėra tokų produktų, kuriuos įsigijus ir įdiegus būtų galima įgyvendinti vienus ar kitus IS nefunkcinius reikalavimus. JAV gynybos departamento standarto DoD 5000.2-R [ST27] 2.3.1 skyrellyje rašoma:

„Išsiaiškinimas, kur būtų galima rinkoje įsigyti produktus, tenkinančius našumo reikalavimus, yra esminis pagrįstų reikalavimų rinkinio formulavimo elementas. Rengiant sistemos našumo reikalavimus, būtina įvertinti, kokiu

būdu norimus našumo reikalavimus galima protingai modifikuoti, siekiant palengvinti potencialų komercinių ar kitų ne pačių sukurtų gaminių, komponentų, specifikacijų, atvirujų standartų, procesų, technologijų ar informacijos šaltinių panaudojimą.“

Kai kurių IS reikalavimų, pavyzdžiui, apsaugos reikalavimų, negalima suformuluoti neatsižvelgiant ir į tokius veiksnius, kaip kompiuterių tinklas, kurio paslaugomis IS, dažniausiai, kartu su kokiomis nors kitomis sistemomis, naudosis, ir duomenų bazių valdymo sistems, kurias ji, taip pat kartu su kitomis sistemomis, naudos savo duomenų bazėms kurti ir tvarkyti. Taigi, išskirti kokią nors programų sistemą ir bandyti autonomiškai formuluoti jos reikalavimus yra visiškai neprasminka. Tai turi būti daroma organizacijos integruotos informacinės sistemos <Trm35> kontekste. Be to, reikalavimų inžinerijos procesas nėra tiesinis procesas, kurį naudojant visą laiką yra judama iš viršaus žemyn. Kartas nuo karto tenka užbėgti į priekį, užsiimti tam tikrais projektavimo ar rinkos analizės darbais, po to grįžti atgal ir galbūt keisti jau suformuluotus reikalavimus, pakylant iki pačio aukščiausio, verslo reikalavimų lygmens.

5.12 Reikalavimų prioritetizavimas

Prioritetai reikalavimams visų pirma yra suteikiami, siekiant nustatyti tų reikalavimų įgyvendinimo eilės tvarką. Aišku, vien prioritetų tam nepakanka, dar reikia atsižvelgti ir į reikalavimų įgyvendinimo rizikos laipsnį. Yra siekiama ir kitų tikslų. Reikalavimų prioritetai padeda parinkti sistemos architektūrą ir priimti kitus projektavimo sprendimus.

Reikalavimų prioretizavimas yra vienas iš reikalavimų klasifikavimo būdų. Reikalavimus galima klasifikuoti pagal daugelį skirtinį kriterijų. Pavyzdžiui, juos galima skirstyti į:

- produkto ir proceso reikalavimus;
- funkcinius ir nefunkcinius reikalavimus;
- pradinius (suformuluotus užsakovų) ir išvestinius (išvestus iš aukštesnio lygmens reikalavimų) reikalavimus;
- privalomus, pageidaujamus ir papildomus reikalavimus arba į kitokius nustatytus prioritetus turinčias reikalavimų grupes;
- lokalizuojamus (susiejamus su konkretais sistemos komponentais) ir nelokalizuojamus (tokius, kurių negalima susieti su konkretais sistemos komponentais) reikalavimus;
- stabilius ir kintančius reikalavimus.

Yra ir kitokių reikalavimų klasifikavimo būdų. Apie juos kalbėsime 8.2 poskyryje. Visi reikalavimų klasifikavimo būdai vienaip ar kitaip siejasi su reikalavimų atributais. Apie tai kalbėsime 8.5 poskyryje. Dabar gi pakalbėsime, kaip galima suskirstyti reikalavimus į grupes pagal jų svarbą, t. y. apie tai, kaip priskirti reikalavimams prioritetus.

Pagal savo svarbą reikalavimai dažniausiai yra skirstomi į tris grupes: aukšto prioriteto reikalavimai, vidutinio prioriteto reikalavimai, žemo prioriteto reikalavimai. Prioritetai reikalavimams yra priskiriami derantis su užsakovu bei kitomis suinteresuotomis šalimis. Be abejo, kažkiek tai padeda spręsti apie vieną ar kitų reikalavimų svarbą užsakovui, tačiau tokia skalė yra labai netiksli ir subjektyvi.

Kitas būdas priskirti reikalavimams prioritetus yra nagrinėti reikalavimų svarbą ir kiek skubiai juos reikia įgyvendinti. Šiuo atveju, kiekvienam reikalavimui priskiriami du įverčiai: „svarbus“ arba „nesvarbus“ ir „skubiai įgyvendintinas“ arba

„neskubiai įgyvendintinas“. Pagal šiuos įverčius reikalavimai suskirstomi į aukšto, vidutinio ir žemo prioritetų grupes (84 pav.).

	Svarbus	Nesvarbus
Skubiai įgyvendintinas	Aukštas	Žemas
Neskubiai įgyvendintinas	Vidutinis	Žemas

84 pav. Reikalavimų prioretizavimas

Toks prioritetų priskyrimo reikalavimams būdas padeda geriau planuoti projektą. Paaiškėja, kad neverta gaišti laiko įgyvendinant skubius, bet nesvarbius reikalavimus, nes jie beveik nedidina kuriamos sistemos vertės. Tokių reikalavimų reikia arba apskritai atsisakyti, arba su jais elgtis taip, tarsi jie būtų neskubūs. Kadangi užsakovas paprastai teigia, kad tokius reikalavimus reikia įgyvendinti kuo skubiau, tai, neturint tokios lentelės, tiems reikalavimams gali būti suteiktas aukštas prioritetas ir tik gavęs pirmąjį sistemos versiją ir neradęs ten jam svarbių dalykų, užsakovas pradės reikšti savo nepasitenkinimą ir paaiškės kad vykdytojai ji suprato ne taip.

Reikia turėti omenyje, kad net ir vidutinio dydžio projektuose esti šimtai funkcinių reikalavimų. Todėl yra neįmanoma aptarti su suinteresuotomis šalimis kiekvieną reikalavimą ir jam priskirti prioritetą. Ši problema paprastai yra sprendžiama prioretizuojant ne pačius reikalavimus, bet sistemos galimybes arba jos vykdomas užduotis, tuo pačiu suteikiant prioritetus ištisoms atitinkamų reikalavimų grupėms.

Nedideliuose projektuose dėl to, kokius prioritetus reikėtų priskirti reikalavimams, galima tiesiog susitarti su užsakovu, pavyzdžiui, naudojant 84 pav. pateiktą lentelę. Dideliuose projektuose reikia formalesnių metodų, padedančių išvengti aistringų ginčų ir sumažinti tiems ginčams neproduktyviai sugaištą laiką. Yra keletas matematinių analitinių metodų, padedančių nustatyti prioritetus, atsižvelgiant į reikalavimų vertę ir jų įgyvendinimo kaštus. Kita alternatyva yra kokybės funkcijų sklaidos metodas, kurį aptarsime 5.13 poskyryje. Dar viena alternatyva yra vadinamasis totalusis kokybės valdymas (angl. *total quality management*). Naudojant šį būdą reikalavimai reitinguojami, atsižvelgiant į kelis projekto sékmės kriterijus.

Taigi, prioretizuojant reikalavimus, yra nustatomi produkto galimybų, užduočių bei konkrečių reikalavimų realizavimo prioritetai. Remiantis tais prioritetais yra sprendžiama, kokioje kuriamos sistemos versijoje ar atmainoje tikslina realizuoti tą ar kitą galimybę, užduotį ar reikalavimą. Kitaip tariant, produkto galimybės, sistemos vykdomos užduotys ir sistemos reikalavimai yra lokalizuojami konkrečiose sistemos versijose bei atmainose. Šitaip yra sudaromi sistemos versijų bei atmainų planai. Aprobavus kokius nors reikalavimų pakeitimus, juos taip pat reikia lokalizuoti versijose bei atmainose ir atitinkamai pakoreguoti tą versiją bei atmainų planus, įskaitant terminus ir kaštus.

Kaip ir rizikos veiksnių analizė, reikalavimų prioretizavimas nėra koks nors atskiras reikalavimų inžinerijos etapas. Dėl prioritetų priskyrimo su suinteresuotomis šalimis reikia tartis jau besiaiškinant reikalavimus. Tačiau iš karto, neturint viso vaizdo, priskirti prioritetus yra labai sunku ir tuo metu išsakyta užsakovo ar kitų suinteresuotų šalių nuomonė turi būti traktuojama tik kaip preliminarus reikalavimų svarbos vertinimas. Realiai vertinti prioritetus galima tik surinkus svarbiausius reikalavimus ir atlikus bent jau preliminarinę jų trūkumų ir rizikos veiksnių analizę. Be to, prioritetų priskyrimas nėra vienkartinis procesas. Prioritetai kinta visą projekto laiką, priklausomai nuo to, kaip kinta verslo tikslai, vartotojų poreikiai, o, kalbant apie rinkoje parduoti kuriamus produktus, ir rinkos sąlygos.

5.13 Kokybės funkcijų sklaida

Kokybės funkcijų sklaida (KFS) <Trm24> – tai gana griežta metodika, naudojant kurią galima susieti kuriamos sistemos galimybes ir savybes su suinteresuotujų šalių tikslais arba, vartojant šio metodo terminiją, su tuo, kas toms šalims yra svarbiausia ir vertingiausia. Ši metodika leidžia prognozuoti, kokias sistemos galimybes ar savybes reikia įgyvendinti, norint susilaikti didžiausio suinteresuotujų šalių, visų pirma, užsakovo, pasitenkinimo. Metodika nėra specifinė. Ji pritaikyta ne tik programų sistemų reikalavimų inžinerijos poreikiams. Tai bendrosios reikalavimų inžinerijos metodika, kuri gali būti ir yra naudojama praktiškai visose gyvenimo srityse, susijusiose su ko nors kūrimu, gamyba ar kokių nors paslaugų teikimu, išskaitant netgi tokias sritys, kaip medicina, maitinimas ir universitetinės studijos. Tačiau svarbiausia šios metodikos taikymo sritis buvo ir tebéra techninės paskirties įrenginių, prietaisų bei gaminiių kūrimas, gamyba ir platinimas.

KFS metodika sukurta Japonijoje praeito amžiaus septintojo dešimtmečio pabaigoje, Europą ji pasiekė devintojo dešimtmečio pabaigoje, o į Lietuvą dar tik ateina. Trumpa KFS istorija pateikta 5 lentelėje.

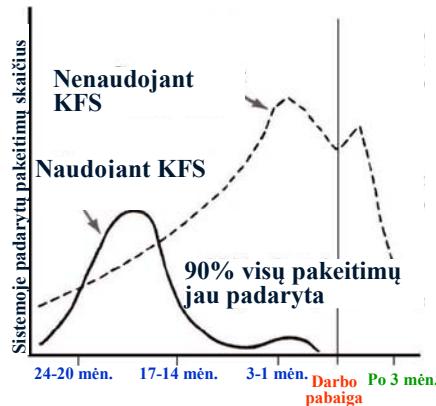
5 lentelė. Trumpa kokybės funkcijų sklaidos metodo istorija.

Metai	Kas svarbaus įvyko tais metais
1954 m	Vienas iš šiuolaikinės kokybės teorijos pradininkų, J.M. Juran, atkreipė dėmesį į kokybės kontrolės svarbą vadyboje.
1961 m.	Kitas šiuolaikinės kokybės teorijos pradininkas, Feigenbaum, pasiūlė totaliojo kokybės valdymo konцепciją.
1966 m	Japonų mokslininkas Yoji Akao, dirbantis koncerne <i>Mitsubishi Kobe Shipyard Japan</i> , pasiūlo pirmąsias KFS idėjas ir panaudoja jas perėjimui nuo produkto maketo prie jo gamybos planuoti. KFS dar nėra metodika. Ji užmanyta kaip vienkartinio panaudojimo strateginio planavimo instrumentas, skirtas specialiai tam konkretiam atveju. Tačiau, gavus gerus rezultatus, šiuo instrumentu susidomima visame koncerne.
1972 m.	Koncernas <i>Mitsubishi Kobe Shipyard Japan</i> paverčia KFS metodika. I metodiką įvedamos kokybės kortos, kurios šiuo metu traktuojamos kaip svarbiausias šios metodikos įrankis.
1975 m.	Japonijoje įsteigiamas vadinamas Kompiuterinių tyrimų komitetas, kurio pagrindinė užduotis tobulinti ir populiarinti KFS metodiką.
1978 m.	Išeina Shigeru Mizuno ir Yoji Akao knyga <i>Quality Function Deployment</i> . Tai pirmoji publikacija šiuo klausimu.
1979 m.	KFS pradeda naudoti kitas japonų koncernas <i>Toyota Group</i> .
1983 m.	Japonijoje organizuojamas pirmasis mokslinis simpoziumas, skirtas KFS ir jos naudojimo klausimams. Masao Kogure ir Yoji Akao paskelbus straipsnį žurnale <i>Quality Progress</i> , apie metodika pirmą kartą sužinoma Jungtinėse Amerikos Valstijose. 1984-1985 m. organizuojami pirmieji mokymo kursai, metodiką pradeda diegti <i>Fords</i> , <i>GM</i> , <i>Chrysler</i> ir kitos JAV bendrovės.
1987 m.	KFS pirmą kartą pristatoma Europoje. Tai vyksta Italijoje. Japonijoje pradedamas KFS standartizavimo procesas.
1988 m.	KFS metodiką pradeda naudoti programinę įrangą kuriančios JAV bendrovės. Iki 1991 m. jis buvo įdiegta IBM, Hewlett Packard, AT&T DEC, ITT ir daugelyje kitų bendrovii.
1989 m.	KFS pradedama naudoti Brazilijoje
1993 m.	Europoje organizuojamas pirmasis mokslinis simpoziumas, skirtas KFS ir jos naudojimo klausimams.
1997 m.	ISO pradeda KFS standartizavimo procesą ISO 9000 standartų šeimos rėmuose (kol kas standartizavimas nėra baigtas).
2006 m.	KFS pradedama dėstyti VU MIF reikalavimų inžinerijos kurso.

Šiuo metu literatūros KFS klausimais yra pakankamai daug. Tačiau apie tai, kaip šią metodiką pritaikyti programų sistemų reikalavimų inžinerijos poreikiams,

rašoma dar gana mažai. Aptarsime šią metodiką, remdamiesi darbais [19, 61, 80, 81, 92, 110, 116].

Kaip jau buvo ne kartą sakyta, dėl reikalavimų, neatitinkančių tikrujų verslo poreikių, vėlesnėse sukurtose sistemos gyvavimo ciklo stadijose tenka daryti daug pakeitimų. Dėl to prarandamos lėšos ir laikas, išauga sistemos kaštai, vėluoja sistemos diegimo terminai, užsakovai lieka nepatenkinti sistema ir ją kūrusia organizacija. Mokslinėje literatūroje aprašyta KFS naudojimo patirtis leidžia teigti, kad ši metodika sistemos perdarymų skaičių padeda ženkliai sumažinti (85 pav.).



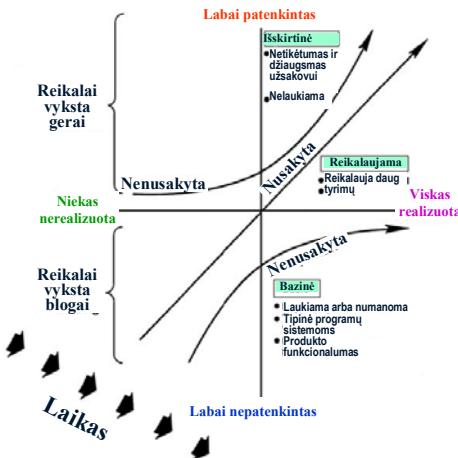
85 pav. Sistemos perdarymų ir taisymų skaičiaus sumažėjimas panaudojus KFS metodiką (paimta iš [110]).

Kaip parodyta 85 pav. pateiktame pavyzdyste, dviejų metų trukmės projekte, dirbant išprastais metodais, sistemos perdarymų skaičius visą laiką auga ir pasiekia piką likus maždaug 1-3 mėnesiams iki projekto pabaigos. Po to jis šiek tiek sumažėja, bet pradėjus diegti sistemą vėl padidėja. Galų gale sistemą perdarinėti baigama tik praėjus maždaug 3 mėnesiams po to, kai naudotojai su ja jau pradėjo dirbti. Naudojant KFS metodiką, stebime visiškai kitą vaizdą. Visų pirma tenka maždaug dvigubai mažiau kartų ką nors keisti ar perdaryti. Antra, daugiausiai pakeitimų daroma ne projekto pabaigoje, o praėjus tik maždaug 7-10 mėnesių nuo jo pradžios. Trečia, su viskuo susitvarkoma prieš sistemą atiduodant užsakovui, o ne po to, kai jis ją gavo ir keletą mėnesių su ja padirbėjo.

Taigi aptarkime kas tai per metodika ir kaip ją reikia panaudoti.

Naudojant KFS, reikalavimai yra skirtomi į tris grupes:

- į laukiamus reikalavimus, t. y. tokius reikalavimus, kurių gali neformuluoti nei viena iš suinteresuotujų šalių, bet apie kuriuos visas jos mano, kad jie yra akivaizdūs, ir neįgyvendinlus kurių bus susilaukta ne tik suinteresuotujų šalių nepasitenkinimo, bet ir nusistebėjimo vykdytojų neišmanymu;
- į normalius reikalavimus, t. y. tokius reikalavimus, kuriuos formuluoja bent viena iš suinteresuotujų šalių, kurie joms visoms atrodo esą aktualūs ir neįgyvendinlus kurių bus susilaukta didelio nepasitenkinimo;
- į išskirtinius reikalavimus, t. y. tokius reikalavimus, kurie suinteresuotosioms šalims yra svarbūs, bet nerealizavus kurių jos nenustebs ir jų nepasitenkinimas nebus labai didelis.



86 pav. Projekto eiga, žvelgiant per kuriamos sistemos kokybės prizmę (paimta iš [80]).

Atitinkamai galima kalbėti ir apie tris kuriamos sistemos kokybės rūšis: bazinę, reikalaujamą ir išskirtinę (86 pav.). Tai vadinamas Kano modelis [60], pritaikytas KFS metodikos atvejui [77]. Bazinė kokybė yra nenusakyta reikalavimais. Ji susijusi su sistemos funkcionalumu ir paprastai visos panašios paskirties programų sistemos ją turi. Suinteresuotosios šalys tikisi tokios kokybės, yra nepatenkintos, jeigu jos nėra, neapsidžiaugia ir traktuoją kaip savaime suprantamą dalyką, jeigu ją gauna. Pavyzdžiui, taip atsitiktų, jeigu, tarkime, kuriant darbo užmokesčio skaičiavimo sistemą, vykdytojai neatsižvelgtų į tai, kad darbo užmokesčis yra apmokestinamas pagal atitinkamais įstatymais nustatytais taisykles.

Reikalaujama kokybė yra nusakoma reikalavimais. Paprastai ji yra sunkiausiai įgyvendinama. Suinteresuotosios šalys tikisi tokios kokybės, yra nepatenkintos, jeigu jos nėra, tačiau apsidžiaugia ją laiku gavusios.

Išskirtinė kokybė taip pat nėra nusakyta reikalavimais. Suinteresuotosios šalys apskritai nesitiki jos gauti ir, savaime suprantama, nereiškia jokio nepasitenkinimo, jeigu jos nėra. Tačiau jos esti labai patenkintos vykdytojų darbu, jeigu tokia kokybė joms pateikiamā.

Kaip gali vykti projektas, žiūrint į jį per taip klasifikuojamą reikalavimą ir jais apibūdinamos kuriamos sistemos kokybės prizmę, patogu parodyti dviejų matavimų koordinacių sistemoje (86 pav.). Vertikaliai koordinacių ašis parodo suinteresuotųjų šalių pasitenkinimo arba nepasitenkinimo laipsnį, o horizontalioji ašis – suformuluotų ir nesuformuluotų reikalavimų įgyvendinimo laipsnį. Jei laikui bėgant nėra įgyvendinama net ir bazinė kokybė, tai suinteresuotųjų šalių susierzinimas auga ir tai reiškia, kad projektas vyksta nesėkmingai. Tačiau, įgyvendant tik bazinę kokybę, sėkmės taip pat nėra ko tikėtis, nors suinteresuotųjų šalių nepasitenkinimas visą laiką tarsi lyg ir mažėja, dėl ko gali susidaryti iliuzija, kad projektas vyksta sėkmingai. Minimalios sėkmės galima tikėtis tik laiku pateikiant reikalaujamą kokybę, o tikroji sėkmė ateina tiktais pateikiant išskirtinę kokybę. Svarbiausioji iš nuostatų, kuriomis yra grindžiama KFS metodika, yra nuostata, jog reikalavimų įgyvendinimą reikia planuoti remiantis būtent šiais pastebėjimais. Kokiai gi būdais metodika padeda šitaip prioretizuoti reikalavimus ir planuoti sistemos kūrimo projektą?

Visų pirmą KFS padeda apibrėžti, vertinti, prioretizuoti reikalavimus, suprantant juos kaip išsakytus ir neišsakytus suinteresuotųjų šalių norus, pageidavimus bei poreikius. Paprastai, aiškinant KFS metodiką, kalbama tik apie dviejų lygmenų reikalavimus. Sakoma, kad ši metodika padeda išreikšti suinteresuotųjų šalių norus bei pageidavimus kuriamosios sistemos funkciniais reikalavimais ir jos techninėmis charakteristikomis, integruoti reikalavimus ne tik į

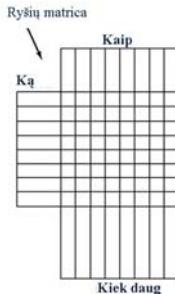
sistemos projektavimo ir konstravimo veiklas, bet ir susieti juos su sistemos diegimo, naudojimo ir netgi demontavimo poreikiais. Be abejo, tai tiesa, tačiau čia yra nutyliimi reikalavimų lygmenys ir gali susidaryti įspūdis, kad yra dirbama tik su dviejų lygmenų reikalavimais: pradiniais dalykinės srities specialistų formuluojamais reikalavimais ir kuriamosios programų sistemos techninio lygmens reikalavimais. Turint omenyje tai, kad metodika buvo sukurta prieš amžiaus septintame dešimtmetyje ir kad ji visų pirma yra skirta ne programų sistemoms, o pačios įvairiausios paskirties produktams, daugeliui iš kurių iš tiesų galima suformuluoti tik dviejų lygmenų reikalavimus, planuoti, projektuoti ir kurti, tokia metodikos pateiktis yra visiškai natūrali. Tačiau iš tiesų nėra jokių kliūčių naudoti metodiką turint daugelio lygmenų reikalavimus ir pritaikyti ją bet kurio lygmens reikalavimams prioretizuoti ir analizuoti. Dar daugiau, kaip bus parodyta 5.14 poskyryje, metodika tinka dirbant ne tik su produkto, bet ir su proceso reikalavimais. Be to, procesą čia galima skaidyti pagal pasirinktą gyvavimo ciklo modelį ir atskirai dirbti su kiekvienos gyvavimo ciklo stadijos reikalavimais. Kitaip tariant, su reikalavimais čia yra dirbama imant dėmesin visą sistemos gyvavimo ciklą ir visose to ciklo stadijose. Sutelkdama visų projekto dalyvių dėmesį į tas sistemos funkcijas ir charakteristikas, kurios suinteresuotosioms šalims turi didžiausią vertę, metodika padeda pagerinti kuriamos sistemos kokybę. Naudojant šią metodiką, visas sistemos kūrimo, diegimo ir platinimo darbuose dalyvaujančios tarnybos yra įtraukiamos į darbą jau pačiose pradinėse projekto stadijose. Pagaliau, metodika padeda dalykinės srities specialistų terminais suformuluotus reikalavimus išversti į inžinieriniam personalui suprantamą kalbą, projektuotojai yra orientuojami į suinteresuotujų šalių lūkesčių įgyvendinimą, palengvinamas visų projekto dalyvių tarpusavio bendravimas.

Kita vertus, KFS nėra sistemos kūrimo proceso valdymo strategija. Ji nepadeda optimizuoti kokybės inžinerijos procedūrų ir tiesiogiai neprisideda prie kuriamos sistemos patikimumo didinimo. Ji taip pat nepadeda rinkti kuriamos sistemos reikalavimų. Metodika tikta padeda įvertinti realią analizuojamo lygmens reikalavimų svarbą užsakovui ir tai, kokių mastu tuos reikalavimus tenkina tarpusavyje konkuruojantys tiekėjai, kurie gali reikiama sistemą sukurti, arba pateikti ją gatavą.

KFS metodikoje vartojama savita terminija. Visos suinteresuotosios šalys čia vadinamos *užsakovu*, bet kurio analizuojamo lygmens reikalavimai – *užsakovo reikalaujama kokybe* arba tiesiog *kokybės reikalavimais*, sistemos charakteristikos, matuojant kurias galima nustatyti, ar pateiktoji sistema tenkina užsakovo reikalaujamą kokybę, – sistemos *kokybės charakteristikomis*, funkcionalumas, kuris turi būti įgyvendintas sistemoje, kad ji turėtų reikiamas kokybės charakteristikas, – *sistemos funkcijomis*. Užsakovas čia visuomet yra suprantamas kaip daugiafunkcinė grupė. Analizuojant verslo lygmens reikalavimus, tokioje grupėje turi dirbti sistemos kūrimą finansuojančiu struktūru atstovai, verslo inžinieriai ir verslo konsultantai, visų organizacijos, kuri naudosis sistema, išorėje esančių suinteresuotujų šalių atstovai. Analizuojant verslo lygmens reikalavimus, daugiafunkcinėje grupėje turi dirbti visų kompiuterizuojamos organizacijos funkcinių padalinių atstovai. Be to, turi būti atstovaujami ir klientų, tiekėjų ir kitais verslo ryšiais su kompiuterizuojama organizacija siejamų šalių interesai. Analogiskai, t. y. vadovaujantis visų to lygmens interesų apėmimo principu, turi būti formuojamos ir kituose lygmenyse užsakovo vaidmenį vaidinančios daugiafunkcinės grupės.

Toliau šiame poskyryje bus prisilaikoma KFS metodikoje priimtos terminijos. Ši metodika susideda iš šešių bazinių žingsnių:

- atitinkamo lygmens terminais aprašoma užsakovo reikalaujama kuriamosios sistemos kokybė arba, kitaip tariant, formuluojamai kuriamosios sistemos atitinkamo lygmens kokybės reikalavimai;
- apibrėžiami ryšiai, susiejantys kokybės reikalavimus su kuriamosios sistemos atitinkamo lygmens kokybės charakteristikomis;
- kokybės reikalavimai pertvarkomi į kuriamosios sistemos atitinkamo lygmens techninius reikalavimus (žemesniame lygmenyje tie reikalavimai jau yra traktuojami kaip to lygmens kokybės reikalavimai);
- parenkami reikalavimų vertinimo metodai ir atliekamas reikalavimų vertinimas;
- vertinimo rezultatai, aprašyti atitinkamo lygmens terminais, pateikiami to lygmens užsakovui (tai vadinama *grįžtamuoju ryšiu*).



87 pav. KFS ryšių matricos struktūra.

Kaip buvo minėta, rinkti ir aiškintis pradinių reikalavimų KFS metodika nepadeda. Tai turi būti daroma kitais būdais, pavyzdžiui, naudojant „smegenų šturmą“ ar interviu, atliekant rinkos tyrimus, nagrinėjant ankstesnių užsakovų nusiskundimus bei jiems duotų garantinių įsipareigojimų vykdymą, analizuojant anksčiau kurtų sistemų baigiamųjų bandymų protokolus ar susipažistant su kitų gamintojų tiekiamomis panašios paskirties sistemomis.

Taigi iš tiesų metodika pradedama naudoti tik antrame žingsnyje. Tiesa, skirtingai nuo kitų metodikų, KFS metodika nereikalauja, kad pradiniai reikalavimai būtų tikslūs. Jie traktuojami kaip to lygmens užsakovo norai ar pageidavimai, kurie įvardijami terminu *užsakovo balsas*. Šie reikalavimai yra tikslinami, išreiškiant juos užsakovui priimtinomis sistemos kokybės charakteristikų reikšmėmis, tiksliau, išreiškiant juos projektavimo reikalavimais. Tam yra naudojama vadinamoji ryšių matrica (87 pav.). Aišku, pereinant iš vieno lygmens į kitą, reikalavimai tampa vis tikslesniais ir griežtesniais, tačiau visiškai tiksliais jie tampa tik pačiamė žemiausiai lygmenyje.

Ryšių matricos „Ką“ (ką reikia sukurti) stulpelyje išrašomi kokybės reikalavimai, „Kaip“ (kaip kurti) eilutėje išrašomi sistemos techniniai reikalavimai, „Kiek daug“ (kokios minimalios kokybės charakteristikų reikšmės) eilutėje – reikalaujamos kokybės charakteristikų reikšmės. Beje, nereikėtų painioti KFS metodikoje ir Zachmano metodiniame karkase vartojamų žodelių ką ir kaip. Nors jie skamba taip pat, bet yra vartojami skirtingomis prasmėmis.

KFS metodikoje nėra skirti funkciniai ir nefunkciniai užsakovo lygmens reikalavimai. Visi kokybės reikalavimai išrašomi „Ką“ stulpelyje. Metodika priverčia nurodyti tų reikalavimų įgyvendinimo laipsnį apibūdinančių sistemos charakteristikų matavimo būdus ir mažiausias priimtinas reikšmes, ko paprastai nėra daroma naudojant kitas metodikas. Kita vertus, reikalavimas matuoti funkcinių reikalavimų įgyvendinimo laipsnį, sukelia rimtų sunkumų. Dažniausiai sunku sugalvoti skaitinius tokio pobūdžio matus ir yra naudojamas paprastas kokybinis matas *Taip/Ne*.

Matricos langleliuose rašomi įverčiai, parodantys kiek stipriai yra priklausomi tame langelyje susikertantys aukštesniojo ir žemesniojo lygmenų reikalavimai. Tam vartojama keturių balų (stipriaus, vidutiniškai, silpnai, visai nesusiję) sistema, kurioje balai žymimi skaitinėmis reikšmėmis 9, 3, 1 ir 0 atitinkamai. Gali būti vartojami ir specialūs grafiniai žymenys (90 pav.).

Būtina pažymeti, kad ryšių matrica apima ne visus, o tiktais lokalizuojamus reikalavimus (žr. 5.14.1 skyrelį). Be to, reikalavimų lokalizavimas čia irgi suprantamas gana savitai. Laikoma, kad reikalavimas yra lokalizuojamas, jei yra susietas ryšiu „stiprus“ su vienu ir tiktais vienu, išimtinais atvejais, su keliais, žemesniojo lygmens reikalavimais. Nelokalizuojamus reikalavimus paprastai įgyvendina visa sistema, o ne kokios nors atskiros jos dalys. Todėl nelokalizuojamą reikalavimą „stiprūs“ ryšiai sieja su visais arba bent jau su dauguma žemesniojo lygmens reikalavimų. Dažniausiai tai esti kuriamos sistemos teisiniai, patikimumo, saugos, aptarnavimo ar priežiūros reikalavimai. KFS metodikoje prie tokio reikalavimų dar yra priskiriami ir kuriamosios sistemos kainos reikalavimai, kurie čia yra traktuojami kaip produkto, o ne kaip proceso reikalavimai (žr. 2.2 poskyri). Nelokalizuojami reikalavimai yra išrašomi kitose, specialiai tam skirtose matricose. Kita vertus, ne visi teisiniai, patikimumo, saugos, aptarnavimo ar priežiūros reikalavimai yra nelokalizuojami. Tie iš jų, kuriuos galima lokalizuoti, yra išrašomi ryšių matricoje.

K _q	
1	Nelokmas
2	Patikimumas
3	Komfortabilus interakcijos
4	Paprastas aplankavimas
5	Gera dokumentacija
6	Darbo paprastumas
7	Miesteliniavimas
8	Gera sprendimų
9	Paprastas instaliavimas

Svarba užsakovui
(nustato užsakovas)

88 pav. Kokybės reikalavimai ir jų svarba užsakovui.

Matrica pradedama pildyti išrašant kokybės reikalavimus ir suderinant su užsakovų tiems reikalavimams teikiamus prioritetus (88 pav.). Kol kas dėmesin yra imami tik užsakovo interesai. KFS pripažista, kad aukštesniojo ir žemesniojo lygmenų reikalavimai, tarkime, verslo lygmens ir vartotojo lygmens reikalavimai, yra skirtingos svarbos. Pirmenybė visuomet yra teikiama aukštesniojo lygmens reikalavimams. Reikalavimų svarba dažniausiai yra vertinama šešių balų (nuo 0 iki 5) sistemoje. Tačiau netgi tais atvejais, kuomet vertinimą atlieka daugiafunkcinė grupė, toks vertinimas, yra gana subjektyvus. Todėl yra naudojami ir formalesni vertinimo metodai, pavyzdžiui, analitinio hierarchijos proceso metodas [98]. Naudojant šį metodą, reikalavimai lyginami poromis ir skaičiuojamas prioritetų pasiskirstymas.

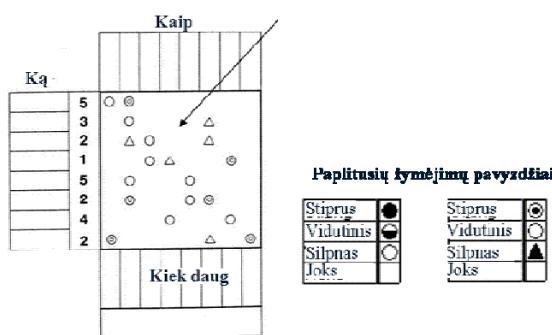
K _q	
1	Nelokmas
2	Patikimumas
3	Komfortabilus interakcijos
4	Paprastas aplankavimas
5	Gera dokumentacija
6	Darbo paprastumas
7	Miesteliniavimas
8	Gera sprendimų
9	Paprastas instaliavimas

Svarba užsakovui
(nustato užsakovas)

- Nustatantys konkretus prioritetas:
- Išskiriantys "K_q" parametrus
- Neigiamas klausius aplankavimui bei instaliavimui
- Galvijamei obiectuose išlikę reikšmės, nesavyje būsi vienai jų pertvarkinti
- Sisteminiuose matavimo būdų

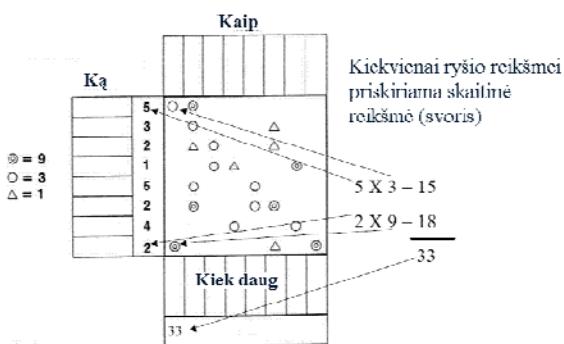
89 pav. Kokybės reikalavimų pertvarkymas į kuriamosios sistemos kokybės charakteristikas.

Suformavus matricos „Ką“ stulpelį ir suteikus prioritetus tame stulpelyje surašytiems reikalavimams, pereinama prie „Kaip“ eilutės formavimo. Šioje eilutėje rašomi žemesniojo lygmens reikalavimai, reikalingi „Ką“ stulpelyje surašytiems reikalavimams įgyvendinti. Vartojant KFS metodikos terminija, yra sakoma, kad „Kaip“ eilutėje rašomi sistemos reikalavimai, reikalingi „Ką“ stulpelyje pateikiems užsakovo reikalavimams įgyvendinti. Kiekvienam užsakovo pateiktam kokybės reikalavimui reikia nuspręsti matuojant kokią sistemos kokybės charakteristiką ar kokią tokią charakteristiką rinkinį bus galima spręsti, kad tas reikalavimas yra įgyvendintas. Kiekvienai tokiai charakteristikai reikia parinkti bent vieną jos matavimo būdą (89 pav.). Rekomenduojama, kad matai būtų kiekybiniai, bet išimtiniais atvejais leidžiama naudoti ir kokybinius matus, pavyzdžiui, matą Taip/Ne. Kiekvieną sistemos reikalavimą turi atitikti viena ir tik viena sistemos kokybės charakteristika. Taigi galima sakyti, kad KFS metodika traktuoją sistemos reikalavimus kaip matuojamus projekto tikslus ir kad panaudojant ryšių matricą aukštesnio lygmens tikslai yra dekomponuojami į žemesnio lygmens tikslus.



90 pav. Ryšių matricos pildymas.

Suformavus „Kaip“ eilutę, pradedamas pačios ryšių matricos pildymas, t. y. įvertinamos užsakovo ir sistemos lygmenų reikalavimų priklausomybės ir į matricos langelius surašomi jų įverčiai (90 pav.). Tai pats sudėtingiausias ir pats atsakingiausias ryšių matricos formavimo etapas. Jį turi daryti patyrę ekspertai, nes blogai įvertinus priklausomybes, bus neteisingai įvertinti projekto prioritetai ir projektas greičiausiai baigsis nesėkmė.



91 pav. Kokybės charakteristikų techninės svarbos vertinimas.

Baigus pildyti matricos langelius, pereinama prie „Kiek daug“ eilutės formavimo arba, kitaip tariant, prie kokybės charakteristikų techninės svarbos vertinimo. Įverčiai skaičiuojami dauginant kiekviename langelyje duotą ryšio stiprumo įvertį iš užsakovo teikiamo prioriteto ir susumuojant gautas sandaugas pagal stulpelius (91 pav.). Šitaip yra įvertinama, kiek svarbi yra viena ar kita kuriamosios

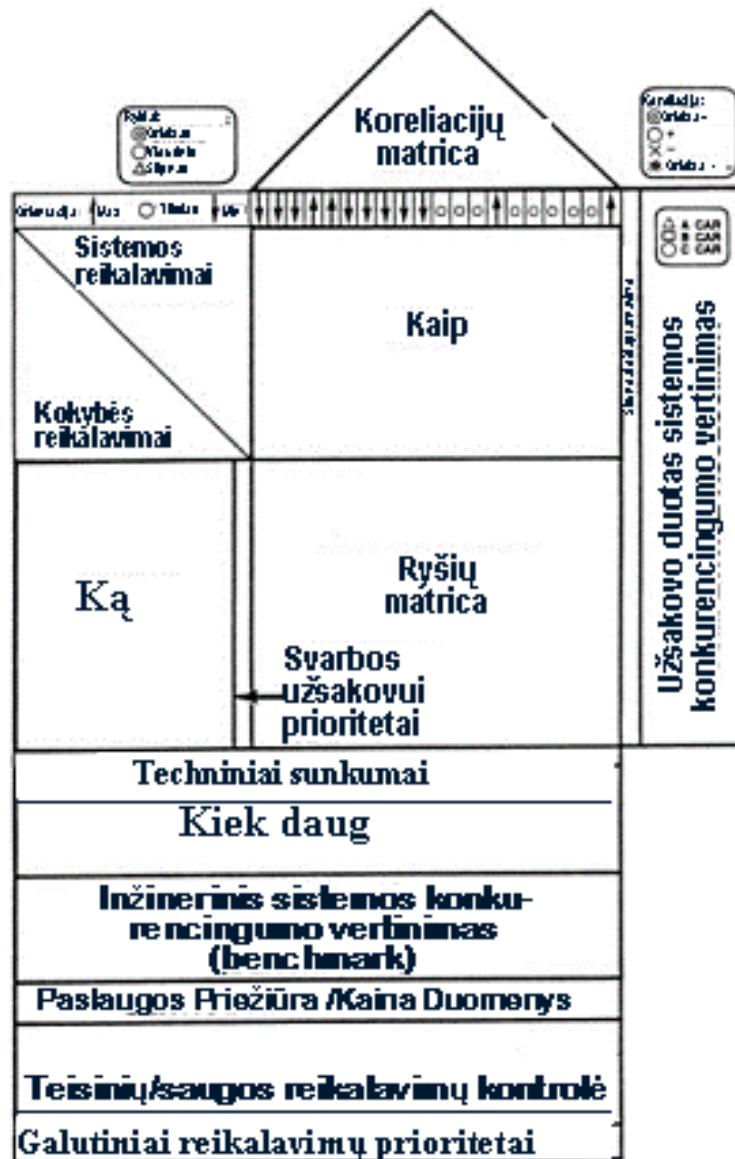
sistemos kokybės charakteristika (žemesnio lygmens reikalavimas) ir kiek dėmesio jai verta skirti projekte.

Užsakovo reikalavimai												
Sistemos reikalavimai												
Nepataisomų klaidų skaičius (%)												
Darbas tinkle (Taip/Net)												
Prieinamumas (% per dieną)												
Duomenų išsiminimo dažnis per val.												
Transakcijos apimtis (Kbaitai)												
Per val. atliekanų transakcijų skaičius												
Aptarnaujamų darbo vietų skaičius												
Reakcijos laikas (sek.)												
Laikas duomenims surinkti (sek./ekr.)												
Leistinas kladų skaičius ekranui												
Mokymosi laikas (val.)												
Užsakovo teikiami prioritetai												
<i>Patogumas</i>												
Galima greitai išmokti	4	9	9	3	3							
Gera pagalba	3	9	9	3								
Nereikia rinkti daug informacijos	3	3	9	3								
<i>Funkcionalumas, našumas</i>												
Padėti dirbtį mūsų buhalterijai	5					9	9	9				
Didelis reaktyvumas	4	3		9	9	9	9					
<i>Prieinamumas ir patikimumas</i>												
Niekad nepraranda duomenų	5								9		3	9
Galima naudotis, kada reikia	3								3	9	3	
Galima naudotis kada panori	4									3	9	
Apsaugo nuo klaidų darymo	4	3							3			9
Techninė svarba (užsakovui)		96	90	66	48	81	81	45	66	39	60	81
Mato reikšmė		20	1	30	2	8	6	500	60	98	+	3
<i>Igyvendinimo sunkumas</i>												
Inžinerinis konkurencingumo vertinimas												
<i>Priežiūros reikalavimai</i>												
<i>Atliktų darbų kaina</i>												
Kontroluojamai reikalavimai	Patiki-mummo	1 reikalavimas	3	0.1	0.3	0.8	0.8	0.3	0.5	1	0.1	1
		2 reikalavimas	5
		3 reikalavimas	1
	Saugos
<i>Techninė svarba</i>												
Kaina	Praradimo		2	3	5	5	5	5	4	3	5	3
	Igyvendinimo		1	4	3	5	4	3	3	2	4	3
<i>Galutiniai reikalavimų prioritetai</i>												
		5	3	5	4	5		5		5	3	3

92 pav. Paprastos KFS ryšių matricos pavyzdys.

Pažymėtina, kad su ryšių matrica patogu dirbtį tiktai tuomet, kuomet ji yra nelabai didelė, nedidesnė nei 30x30 langelių. Jei matrica yra didesnė, reikalavimus reikia skaidyti į kelis nepriklausomų reikalavimų blokus ir kiekvienam tokiam blokui rengti savą ryšių matricą. Deja, ne visuomet pavyksta šitaip išskaidyti reikalavimus.

Suformavus „Kiek daug“ eilutę yra baigiamas visos ryšių matricos formavimas. Paprastos ryšių matricos pavyzdys pateiktas 92 pav. Jame grafiniai žymenys nevartojami, iš karto surašyti užsakovo ir sistemos lygmenų reikalavimų priklausomybių skaitiniai įverčiai. Pavyzdžiui, matome, kad užsakovo reikalavimas „Galima greitai išmokti“ ir sistemos reikalavimas arba, kitaip tariant, sistemos kokybės charakteristika „Mokymosi laikas“ yra tarpusavyje susiję stipriai (įvertus 9). Grafiniai įverčiai yra akivaizdesni, tačiau jie yra patogūs tik tuomet, kuomet naudojamas kokia nors KFS metodiką palaikančia instrumentine sistema.



93 pav. KFS kokybės korta.

Ryšių matrica yra tik vienas iš vadinamosios kokybės kortos (93 pav.) elementų. Pagal savo formą ši korta primena namą ir todėl dažnai dar yra vadinama KFS *kokybės nameliu*. Reikia pasakyti, jog gana dažnai kokybės korta klaidingai yra tapatinama su pačia KFS metodika. Iš tiesų tai tik vienas iš būdų patogiai pateikti įvairias matricas ir kitą informaciją, kurios prisireikia naudojant KFS metodiką. Tačiau visą tą informaciją galima pateikti ir įvairiais kitais būdais. Be to, kadangi KFS metodika yra bendrosios reikalavimų inžinerijos lygmens metodika, tai, pritaikant ją konkrečiai dalykinei sričiai, tarkime, programų sistemoms kurti, šią informaciją taip

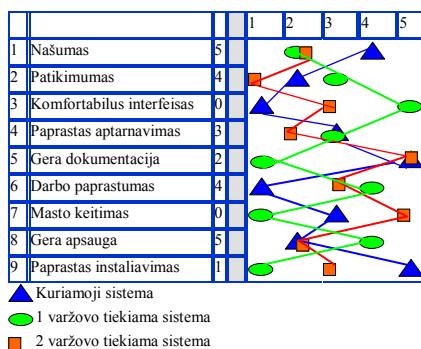
pat reikia sukonkretinti ir detalizuoti. Tą galima padaryti daugeliu būdų. Todėl galima rasti daug darbų, taip pat ir skirtų KFS naudojimo programų sistemų reikalavimų inžinerijoje klausimams, kuriuose ne tik nieko nekalbama apie kokybės kortą ir jos panaudojama, bet ir yra naudojamos gana skirtinges matricos bei skirtinges kuriamų sistemų savybių hierarchijos.

Ryšių matrica yra, centrinis KFS metodikos elementas, tačiau kiti jos elementai taip pat yra labai svarbūs. Kokybės namelio stoge talpinama vadinamoji koreliacijų matrica (94 pav.). Tai trikampė matrica, aprašanti skirtinges sistemos kokybės charakteristikų arba, kitaip tariant, sistemos techninių reikalavimų tarpusavio sąveiką.



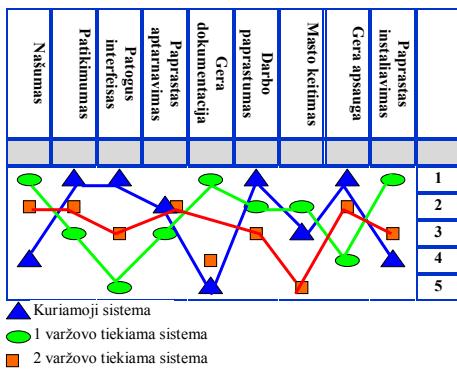
94 pav. Koreliacijų matrica.

Panašiai kaip ir nefunkcinių reikalavimų atveju (žr. 4.4 poskyrį), galimos teigiamos ir neigiamos sistemos kokybės charakteristikų koreliacijos. Tiksliau kalbant, yra skiriamos griežtai teigiamos, teigiamos, neigiamos ir griežtai neigiamos koreliacijos. Griežtai neigiamos ir neigiamos koreliacijos rodo, kad dvi kokybės charakteristikos konfliktuoja arba, kitaip tariant, yra prieštaringos. Tuos konfliktus reikia išspręsti.



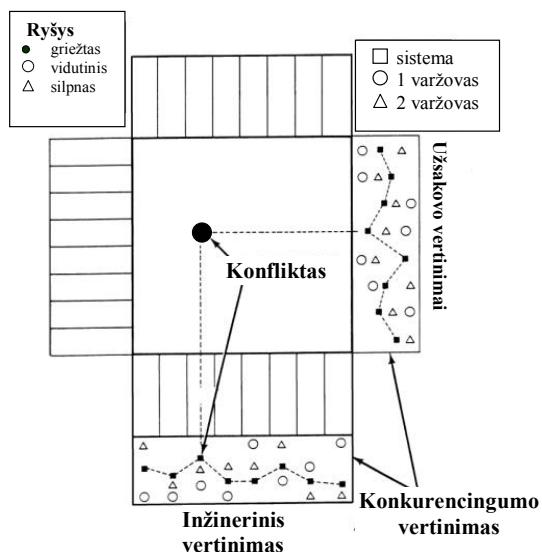
95 pav. Užsakovo duotų varžovų tiekiamų sistemų vertinimo matrica.

Koreliacijų matrica kokybės kortoje išplečia ryšių matricą aukštin. Tačiau ši matrica yra plečiama ir į kitas kokybės kortos puses arba, kalbant tiksliau, turi būti susieta su įvairiomis kitomis matricomis. Kokybės kortos dešinėje talpinama varžovų tiekiamų sistemų vertinimo matricą (95 pav.). Šioje matricijoje užsakovas penkių balų sistemoje vertina varžovų tiekiamas analogiškos paskirties sistemas. Užsakovo vertinimai renkami kartu su jo formuluojamų reikalavimų svarbos vertinimais. Tuo pat metu užsakovas duoda ir kuriamos sistemas vertinimus, t. y. išsako nuomonę, kaip turėtų atrodyti kuriamoji sistema, lyginant ją su varžovų tiekiamomis sistemomis. Taigi ši matrica palygina kuriamąją ir rinkoje esamas sistemas, žvelgiant į jas iš užsakovo požiūrio taško. Aišku, visa tai galima padaryti tiktais tuomet, kuomet užsakovas yra bent šiek tiek susipažinęs su rinkoje parduodamomis analogiškos paskirties sistemomis ir yra pajėgus jas vertinti.



96 pav. Varžovų tiekiamų sistemų inžinerinio vertinimo matrica.

KFS metodikoje nėra ribojamasi vien tik užsakovo atliktu konkuruojančių sistemų vertinimais. Baigus kurti sistemą, atliekamas inžinerinis tos sistemos vertinimas, lyginant ją su varžovų tiekiamomis sistemomis. Tam naudojami testavimo ir kiti programų sistemų vertinimo metodai. Šiuo atvejai įverčiai taip pat yra duodami penkių balų sistemoje. Vertinimo rezultatai pateikiami inžinerinio vertinimo matricoje (96 pav.), kuri kokybės kortoje yra talpinama žemiau ryšių matricos. Pažymėtina, kad inžineriniai vertinimai gali gerokai skirtis nuo užsakovo duotų vertinimų.



97 pav. Konfliktų vertinimas.

Jeigu užsakovas teigia, kad koks nors jo suformuluotas kokybės reikalavimas yra prastai tenkinamas, tai yra atliekamas užsakovo duotų vertinimų ir inžinerinio vertinimo rezultatų lyginimas. Be to, tuo reikalavimu nusakoma sistemos savybė yra palyginama su atitinkama varžovų tiekiamų sistemų savybe. Jei paaiškėja, jog užsakovas iš tiesų yra teius ir reikalavimas nebuvo taip įgyvendintas, kaip buvo nurodės užsakovas (97 pav.), sistemą tenka tobulinti.

KFS metodikoje dirbama ir su kitomis informacinėmis struktūromis (93 pav.). Kokybės kortos eilutė „Techniniai sunkumai“ skirta vykdymui vertinti, kiek sunku yra įgyvendinti kiekvieną iš ryšių matricoje pateiktų sistemos reikalavimų. Sprendžiant apie galutinius reikalavimų prioritetus, apsiriboti tik sistemos reikalavimų įverčiais, išskaičiuotais remiantis užsakovo reikalavimų prioritetais, būtų neprotinga. Taip elgiantis gali, pavyzdžiu, paaiškėti, kad nors kuris nors reikalavimas ir yra labai svarbus užsakovui, bet dėl nepakankamos vykdymo kvalifikacijos ar dėl kokių nors

kitų priežasčių tą reikalavimą įgyvendinti yra labai sunku ir, apsiėmus tai padaryti, gali būti viršytas projekto biudžetas, nespėjama sukurti sistemos numatytu laiku arba galbūt projektas gali ir apskritai sužlugti. Taigi reikia įvertinti techninį kiekvieno reikalavimo įgyvendinamumą – tai gali būti daroma penkių balų skaleje arba vertinant žmogaus darbo valandomis – ir tuos įverčius kartu su sistemos reikalavimų įverčiais panaudoti nustatant galutinius reikalavimų prioritetus, surašomus į kokybės kortos eilutę „Galutiniai reikalavimų prioritetai“ (93 pav.). Remiantis šiais prioritetais yra sudaromas projekto vykdymo planas, projektuojamasis tolimesnis programų sistemos inžinerijos procesas ir sprendžiami visi kiti projekto vykdymo klausimai.

Dar dvi kokybės kortos matricos skirtos darbui su nelokalizuojamais reikalavimais. Kalbant apie programų sistemų reikalavimus, tokius reikalavimus tikslina skirstyti į dvi grupes: sistemos priežiūros reikalavimus ir reikalavimus susijusius su įvairiomis grėsmėmis arba, kitaip tariant, sistemos (ne jos kūrimo) rizikos veiksnius.

Ką	Kaip					
	R1	R2	R3	Rn	
.....
Techninė svarba
Mato reikšmė
Įgyvendinimo sunkumas
Inžinerinis konkurencingumo vertinimas					
Keliamumas į kitas platformas	0					
Plečiamumas	5	0	0.7	1	0.2
Keičiamumas	5	1	0	0.5	0.8
Tobulinimas	3	0	0.6	0.8	...	0
Greitas trykių priežasčių nustatymas	4	0	1	0.6	1
Techninė svarba	5	9.3	12.3	...	9	
Kaina	80	160	120		200	
Igyvendinti prižiūrimumą	40	350	120		400	
Perrašyti kodą iš naujo						

Priežiūros
reikalavimai

Žmogaus darbo
valandos

Tikimybės

98 pav. Prižiūrimumo matrica.

Norint sistemą padaryti lengviau prižiūrimą, t. y. lengviau keičiamą, plečiamą, tobulinamą bei perkeliamą į kitas kompiuterines platformas, reikia naudoti specialias projektavimo ir programavimo technologijas. Kadangi tokios technologijos turėtų būti panaudotos visoje sistemoje, tai atitinkami reikalavimai negali būti lokalizuoti, jie „sukimba“ praktiškai su visais žemesnio lygmens reikalavimais ir ta sankiba visuomet yra aprašoma ryšiu „stiprus“. Tačiau užsakovas mano, kad kai kurie prižiūrimumo reikalavimai yra svarbesni, negu kiti ir gali sudėti atitinkamus prioritetus. Pavyzdžiui, gali būti žinoma, kad kuriamosios sistemos apskritai neplanuojama kelti į kitas kompiuterines platformas ir tokiam reikalavimui galima priskirti prioritetą 0. Be to, matricoje, susiejančioje prižiūrimumo reikalavimus su žemesniojo lygmens reikalavimais, tikslina rašyti ne jų tarpusavio ryšio stiprumo įverčius, bet tikimybes, kad atitinkamus žemesniojo lygmens reikalavimus teks keisti. Dar vienas parametras, kurį verta imti dėmesin, sprendžiant ar atitinkamą reikalavimą verta daryti prižiūrimu, yra kaina. Jei reikalavimo kitimo tikimybė yra nedidelė, gali būti pigiau kartas nuo karto perrašyti atitinkamas programų sistemos dalis, negu visą sistemą programuoti taip, kad joje bet ką būtų galima lengvai keisti. Prižiūrimumo matrica (98 pav.) yra talpinama kokybės kortos dalyje „Paslaugos, priežiūra/Kaina, duomenys“ (93 pav.). Kaip parodyta 98 pav. pateiktame pavyzdyme, užsakovo nedomina sistemos

keliamumas į kitas kompiuterines platformas. Didžiausią svarbą jis teikia sistemos plečiamumui ir keičiamumui, po to – galimybei greitai nustatyti sistemos trikčių priežastis. Iš matricos matome, kad bendru užsakovo ir vykdytojų vertinimu niekuomet nereikės plėsti reikalavimą R1 įgyvendinantį kodą, bet tikrai reikės jį perrašyti. Tai reiškia, kad reikalavimas R1 yra laikinas ir artimiausioje ateityje bus pakeistas kitu reikalavimu. Kita vertus matome, jog realizuoti mechanizmus, leidžiančius tą reikalavimą lengvai prižiūrėti, vykdytojo vertinimu, kainuotų maždaug 80 žmogaus darbo valandų, o visam tą reikalavimą realizuojančiam kodui iš naujo perrašyti prireiktų tik 40 žmogaus darbo valandų. Taigi yra aišku, kad programuojant tą reikalavimą nėra jokios prasmės atsižvelgti į priežiūros reikalavimus. Panašiai yra analizuojami ir kiti bet kurio lygmens reikalavimai. Matome, jog KFS metodika numato gana patogias programų sistemų priežiūros reikalavimų analizės priemones, tačiau vis dėlto reikia pastebeti, kad darbo su šiais reikalavimais klausimai kol kas KFS metodikoje yra ištobulinti nepakankamai.

Labai panašiai yra dirbama ir su kitais nelokalizuojamais reikalavimais. Pagrindinis skirtumas yra tokis, kad visi šie reikalavimai – patikimumo, saugos, apsaugos, teisiniai ir pan. – gali būti pažeidžiami tuomet, kuomet realizuoja vienos ar kitos grėsmės arba, kitaip tariant, kuomet suveikia vieni ar kiti rizikos veiksniai. Analizuojant tokius reikalavimus reikia nuspresti, kokiui mastui yra tikslinga kontroliuoti tuos rizikos veiksnius. Todėl visi tokio pobūdžio reikalavimai yra vadinami kontroliuojamais reikalavimais. Tokių reikalavimų analizė iš esmės yra sistemos naudojimo rizikos veiksnų analizė. Nereikėtų jos painioti su projekto rizikos veiksnų analize. Tai visai skirtinių dalykai.

Be abejo, yra labai daug grėsmių, galinčių vienaip ar kitaip pažeisti kuriamą sistemą. Tai ir nesankcionuotas sistemos panaudojimas, ir klaidos, galinčios sukelti nepataisomas triktis, ir materialinė ar kitokia žala, kurią gali padaryti vienu ar kitu požiūriu nesaugi sistema, ir galimi daugelij metų kaupėti duomenų praradimai, ir potenciali galimybė pažeisti kokius nors galiojančius teisinius aktus ir daugelis kitų atvejų. Visų grėsmių neįmanoma netgi išvardinti, o ką jau bekalbetti apie apsaugos nuo visų jų įgyvendinimo sistemoje galimybes. Todėl apie tai, ar apskritai yra verta saugotis nuo kurios nors konkrečios grėsmės ir, jeigu tai yra verta daryti, tai kokiu mastu, yra sprendžiama vadovaujantis dviem kriterijais: tikimybė, kad grėsmė realiuosis; nuostoliais, kurie bus padaryti, jeigu ji realiuosis; ir apsaugos mechanizmų įgyvendinimo kaina. Savaime aišku, būtų visiškai neprotina kurti tokius apsaugos mechanizmus, kurių sukūrimas kainuos daug brangiau, negu nuostoliai, kurie bus padaryti tuos mechanizmus pralaužus.

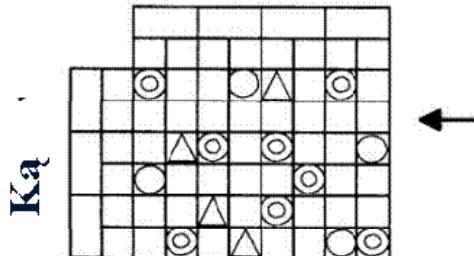
Kontroliuojamų reikalavimų matricos struktūra parodyta 92 pav. Kiekvienai grėsmei užsakovas nurodo apsaugos nuo jos svarbą. Matricos langeliuose kiekvienam žemesniojo lygmens reikalavimui nurodoma tikimybė, kad atitinkama grėsmė gali paveikti tą reikalavimą. Matricos apačioje kiekvienam reikalavimui nurodoma jo apsaugos nuo grėsmių techninė svarba ir vertinimai (nuo 0 iki 5), kokią žalą patirs užsakovas, jei tos grėsmės realiuosis, ir kiek techniskai yra sunku apsaugoti tą reikalavimą nuo grėsmių poveikio. Kai kurie autorai yra pasiūlę, kaip būtų galima atlikti išsamesnę rizikos veiksnų analizę. Tačiau, kaip ir priežiūros reikalavimų atveju, programų sistemų rizikos veiksnų analizės ir vertinimo klausimai KFS metodikoje kol kas nėra labai ištobulinti.

Dabar pakalbėkime apie tai, kaip galima panaudoti KFS kokybės kortą, tiksliau, ryšių matricą, reikalavimų trūkumams nustatyti. Daugeliui reikalavimų trūkumų nustatyti pakanka atlikti vien tik formalią tos matricos analizę, nesigilinant į joje surašytų reikalavimų pobūdį. Aišku, norint pašalinti aptiktus trūkumus, jau tenka

gilintis į reikalavimų prasmę. Taikant tiktai vienus ar kitus formalius metodus ar procedūras, analizės metu aptiktų reikalavimų trūkumų pašalinti yra neįmanoma.

Formaliai analizuojant ryšių matricą, galima aptikti kai kuriuos skirtingų lygmenų ar to pačio lygmens reikalavimų darnos pažeidimus ir jų ryšio matricą įrašytus nelokalizuojamus reikalavimus.

Kaip

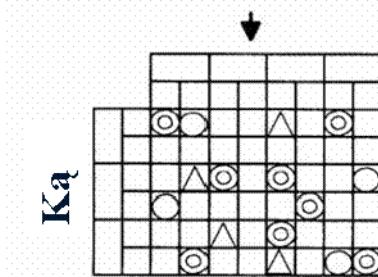


99 pav. Tuščios ryšių matricos eilutės.

Ryšių matricą padeda nustatyti penkias skirtingų lygmenų reikalavimų darnos pažeidimų rūšis. Visų pirmą tokius pažeidimus parodo tuščios ryšių matricos eilutės (99 pav.) ir tušti jos stulpeliai (100 pav.). Tuščia ryšių matricos eilutė parodo, kad atitinkamas aukštesnio lygmens reikalavimas žemesniame lygmenyje neturi jokių atitikmenų arba, kitaip tariant, nėra lokalizuotas. Faktiškai jis yra ignoruojamas ir kuriamoje sistemoje apskritai nebus įgyvendintas. Taigi, radus tuščią matricos eilutę, reikia grįžti atgal, iš naujo peržiūrėti visą ryšių matricą ir nuspręsti, kaip pašalinti ši reikalavimų darnos pažeidimą. Galima priimti vieną iš trijų sprendimų:

- nuspręsti, kad aukštesniojo lygmens reikalavimas yra perteklinis ir jį, kaip nereikalingą, išbraukti iš matricos;
- nuspręsti, kad aukštesniojo lygmens reikalavimas yra netiksliai suformuluotas ir jį taip performuluoti, kad jis būtų galima susieti su bent vienu žemesniojo lygmens reikalavimu;
- nuspręsti, kad yra praleisti kokie nors žemesniojo lygmens reikalavimai, nusakantys aukštesniojo lygmens reikalavimo įgyvendinimo būdą, ir žemesniojo lygmens reikalavimų sąrašą papildyti atitinkamais reikalavimais.

Kaip



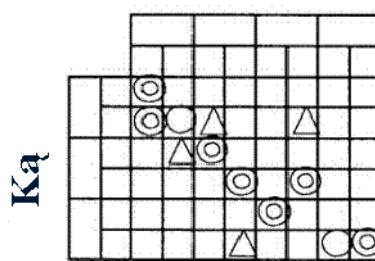
100 pav. Tušti ryšių matricos stulpeliai.

Tušti ryšių matricos stulpeliai (100 pav.) tiktai signalizuoją apie galimus skirtingu lygmenų reikalavimų darnos pažeidimus, bet dar nereiškia, kad tokie pažeidimai iš tiesų yra padaryti. Tuščias stulpelis rodo, kad atitinkamas žemesniojo lygmens reikalavimas neprisideda nei prie vieno iš aukštesniojo lygmens reikalavimų

įgyvendinimo ir galbūt yra perteklinis. Kadangi žemesniojo lygmens reikalavimai yra gaunami iš aukštesniojo lygmens reikalavimų, tai, bendruoju atveju, kiekvienas žemesniojo lygmens reikalavimas turėtų būti susietas su bent vienu aukštesniojo lygmens reikalavimu ir susietas ne bet kokiui ryšiu, o būtent ryšiu „stiprus“. Aišku, dažniausiai jis esti susietas su keliais aukštesniojo lygmens reikalavimais. Tačiau gali būti ir taip, kad kai kurie žemesniojo lygmens reikalavimai yra reikalingi ne tam, kad apskritai būtų įgyvendinti aukštesniojo lygmens reikalavimus, o tam, kad jie būtų įgyvendinti tam tikru konkrečiu būdu. Kitaip tariant, žemesniame lygmenyje gali atsirasti papildomi, pridėtiniai reikalavimai, atspindintys tame lygmenyje priimtus sistemos projektavimo sprendimus. Išsamiau tai aptarsime 5.14 poskyryje.

Tuščias ryšių matricos stulpelis dar gali reikšti ir tai, kad atitinkamas žemesniojo lygmens reikalavimas yra susijęs ne su pačia kuriamaja sistema, o su jos aptarnavimu ar priežiūra. Tokiu atveju ši reikalavimą reikia perkelti iš ryšių matricos į priežiūros reikalavimų matricą (93 pav.).

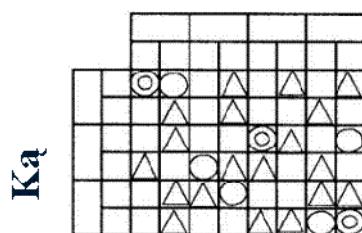
Kaip



101 pav. Dominuojančios diagonalės ryšių matrica.

Dar vienas skirtingų lygmenų reikalavimų darnos pažeidimo požymis yra vadinamoji dominuojančios diagonalės ryšių matrica (101 pav.), t. y. tokia matrica, kurios diagonalėje dominuoja žymuo „stiprus“, o visa kita matricos dalis yra beveik neužpildyta. Tai reiškia, kad beveik visi aukštesniojo lygmens reikalavimai žemesnijame lygmenyje buvo tiesiog pakartoti. Taip gali atsitikti tik tuo atveju, kuomet vieno iš lygmenų reikalavimus formulavo ne tie specialistas, kuriu požiūri į kuriamają sistemą turėtų atspindėti to lygmens reikalavimai. Pavyzdžiui, taip gali atsitikti tuomet, kuomet verslo lygmens reikalavimus formuluoja ne verslo konsultantai ar verslo inžinieriai, o dalykinės srities specialistai, t. y. žmonės, kurie patys naudosis ta sistema. Šitaip yra prarandamas vadinamasis užsakovo balsas, pakeičiant jį naudotojų balsu. Panašūs dalykai gali atsitikti ir formuluojant žemesnių lygmenų reikalavimus.

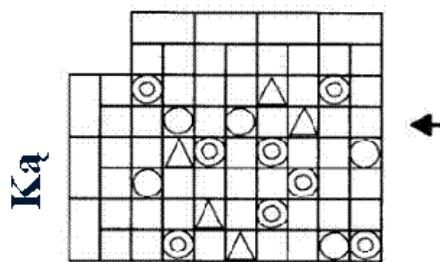
Kaip



102 pav. Ryšių matrica, kurioje dominuoja ryšiai „silpnas“.

Ryšių matrica, kurioje dominuoja ryšiai „silpnas“ (102 pav.), taip pat rodo, kad aukštesniojo ir žemesniojo lygmenų reikalavimų darna yra pažeista. Kaip jau buvo minėta, kiekvienas išvestinis žemesniojo lygmens reikalavimas turėtų būti bent su vienu aukštesniojo lygmens reikalavimu susietas ryšiu „stiprus“, nes kiekvieno aukštesniojo lygmens reikalavimo esmė turi būti išreikšta bent vienu žemesniojo lygmens reikalavimu. Tokioje matricoje šitaip nėra. Taigi žemesniojo lygmens reikalavimai yra neadekvatūs aukštesniojo lygmens reikalavimams ir juos reikia tikslinti.

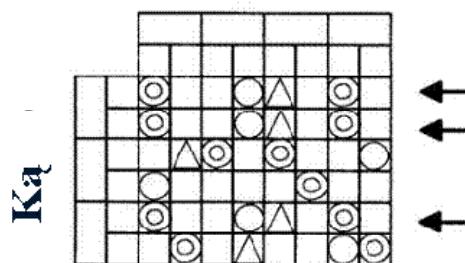
Kaip



103 pav. Ryšių matrica, turinti eilučių, kuriose nėra nei vieno ryšio „stiprus“.

Pagaliau, ryšių matrica, turinti eilučių, kuriose nėra nei vieno ryšio „stiprus“, rodo, kad atitinkami aukštesnio lygmens reikalavimai bus ne iki galo įgyvendinti, nes nei vienas iš žemesnio lygmens reikalavimų neatspindi jų esmęs. Tokiais atvejais reikia peržiūrėti žemesnio lygmens reikalavimus ir pabandyti taip juos formuliuoti, kad atsirastų reikalavimų geriau atspindinčių atitinkamų aukštesnio lygmens reikalavimų esmę.

Kaip

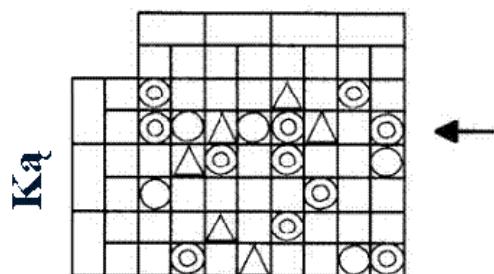


104 pav. Ryšių matrica su pasikartojančiomis eilutėmis.

Ieškant to paties lygmens reikalavimų darnos pažeidimų, formalii ryšių matricos analizė yra daug mažiau naudinga, negu ieškant skirtingų lygmenų reikalavimų darnos pažeidimų. Ji padeda ieškoti tik aukštesniojo lygmens reikalavimų darnos pažeidimų. Analizuoti pačio žemiausio lygmens reikalavimų ryšių matrica apskritai niekaip nepadeda. Be to, naudojant ryšių matricą, aukštesniojo lygmens reikalavimuose galima rasti tik vienos rūšies darnos pažeidimus – reikalavimų abstrakcijos lygmenų darnos pažeidimus. Tokių pažeidimų požymis yra matrica su pasikartojančiomis eilutėmis (104 pav.). Tų pačių ryšių pasikartojimas skirtingose eilutėse kalba apie tai, kad atitinkami reikalavimai greičiausiai priklauso skirtiniems abstrakcijos lygmenims arba, kitaip tariant, kai kurie reikalavimai yra daliniai kitų reikalavimų atvejai. Susidarius tokiai situacijai, reikia grįžti prie reikalavimų

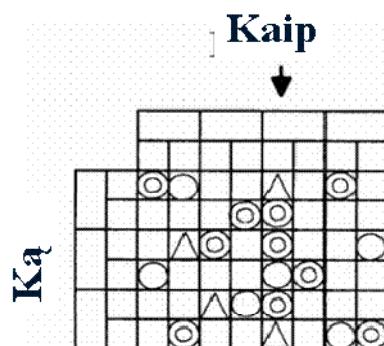
konkretizavimo procedūrų (žr. 5.4 poskyrį) ir patikrinti, ar reikalavimų konkretizavimas buvo atliktas korekiškai. Nesuvienodinus reikalavimų abstrakcijos lygmenų, vėliau paprastai kyla rimtos kokybės valdymo problemos.

Kaip Ką



105 pav. Ryšių matrica, turinti per daug pilnų eilučių.

Paskutinis reikalavimų trūkumas, kurį galima rasti formaliai analizuojant ryšių matricą, yra neteisingas reikalavimų skirstymas į lokalizuojamus ir nelokalizuojamus reikalavimus. Jei turime matricą, kurioje yra eilučių beveik neturinčių tuščių langelių (105 pav.), tai reiškia, kad atitinkami aukštesniojo lygmens reikalavimai yra „sukibę“ beveik su visais žemesniojo lygmens reikalavimais. Nei su funkciniais, nei su įprastais kokybės reikalavimais taip atsitiktį negali. Todėl, matyt, tie reikalavimai yra kainos, saugumo, patikimumo arba juridiniai reikalavimai ir iš ryšių matricos turi būti perkelti į kainos ar juridinių ir saugos reikalavimų kontrolės sritį (93 pav.).



106 pav. Ryšių matrica, turinti per daug pilnų stulpelių.

Panašiai yra ir tais atvejais, kuomet ryšių matrica, turi stulpelių beveik neturinčių tuščių langelių (106 pav.). Tiktai šiuo atveju reikalą turime ne su aukštesniojo, o su žemesniojo lygmens reikalavimais. Tokius reikalavimus taip pat reikia perkelti į kainos ar juridinių ir saugos reikalavimų kontrolės sritį (93 pav.).

Baigiant kalbėti apie KSF metodiką pasakytina, kad mes aptarėme tik vieną iš jos variantų. Šiuo metu mokslinėje literatūroje galima rasti dar bent keletą panašių variantų. Jie skiriasi vienas nuo kito naudojamų matricų skaičiumi (didžiausias matricų skaičius, net 30, naudojamas vadinamoje KFS matricų matricos metodikoje) ir įvairiomis kitomis detaliemis, tačiau iš esmės visi jie yra grindžiami šiame poskyryje išdėstyтомis idėjomis.

5.14 Reikalavimų lokalizavimas, nuleidimas žemyn ir operacionalizavimas

5.14.1 Reikalavimų lokalizavimo ir nuleidimo žemyn metodų svarba

Reikalavimų lokalizavimas <Trm50> ir nuleidimas žemyn <Trm53> yra vieni iš svarbiausių, o gal ir patys svarbiausi reikalavimų inžinerijos metodai. Šie metodai naudojami praktiškai visą projekto vykdymo laiką. Analizuojant verslo sistemas verslo problemos ir grėsmės yra lokalizuojamos verslo funkcijose ir nuleidžiamos žemyn į tą funkciją lygmenį. Verslo funkcijų lygmens problemos lokalizuojamos verslo procesuose ir nuleidžiamos į tą procesą lygmenį, po to – procesų lygmens problemos lokalizuojamos veiklose ir nuleidžiamos į veiklą lygmenį. Analizuojant verslo užduotis tos problemos lokalizuojamos analizuojamose užduotyse ir nuleidžiamos į jų lygmenį. Šitaip yra išsiaiškinamos verslo problemų ir grėsmių priežastys. Produkto galimybės yra lokalizuojamos vykdytojų, atsakingų už galimybių realizavimą, grupėse ir atitinkamuose projekto valdymo lygmenyse, pavedant tuose lygmenyse veikiantiems vadovams kontroliuoti tą galimybę realizavimo eiga. Po to jos yra nuleidžiamos į tuos valdymo lygmenis. Nustačius kokias paslaugas teiks kuriamoji informacinė sistema, jos yra lokalizuojamos dalykinės srities specialistų pareigybiniėse darbo vietose, klientų aptarnavimo terminaluose bei kitiems paslaugų gavėjams (verslo partneriams, kompiuteriniams procesams, įrenginiams) skirtuose prieigos prie sistemos mazguose ir nuleidžiamos žemyn į paslaugą gavėjų lygmenį. Kita vertus, informacinės sistemos galimybės yra lokalizuojamos tos sistemos komponentuose ir nuleidžiamos žemyn į tą komponentą lygmenį. Informacinės sistemos teikiamų paslaugų ribojimai taip pat yra lokalizuojami informacinės sistemos komponentuose ir nuleidžiami į tą komponentą lygmenį. Vartotojo lygmenyje suformuluoti konцепciniai verslo objektus modeliuojančią informacinių objektų reikalavimai yra lokalizuojami tose informacinėje sistemoje numatytose informacijos saugyklose ir nuleidžiami į tą saugyklą lygmenį. Informacinės sistemos komponentų reikalavimai yra lokalizuojami tuos komponentus palaikančiose programų sistemose ir nuleidžiami į programų sistemų lygmenį. Pagaliau, programų sistemų reikalavimai yra lokalizuojami tą sistemą komponentuose ir nuleidžiami į tą komponentą lygmenį. Taigi, kaip matome, reikalavimų lokalizavimo ir nuleidimo žemyn metodai yra naudojami visuose reikalavimų formulavimo ir netgi jos projektavimo etapuose. Be abejo tai, kur yra lokalizuojami reikalavimai ir kaip jie nuleidžiami žemyn, priklauso nuo projekte naudojamo programų sistemų inžinerijos procesų ir darbo su reikalavimais metodiką. Pavyzdžiui, naudojant kokybės funkcijų sklaidos metodiką, viskas vyksta šiek tiek kitaip negu buvo aptarta aukščiau. Tačiau, nepriklausomai nuo to, kokios metodikos yra naudojamos, vis vien kiekviename žingsnyje prieikia kur nors lokalizuoti reikalavimus ir juos nuleisti žemyn. Taigi aptarkime, kas yra vadinama reikalavimų lokalizavimu ir nuleidimu žemyn.

5.14.2 Reikalavimų lokalizavimo ir nuleidimo žemyn samprata

Reikalavimų (problemų, grėsmių, galimybių, tikslų ir kt.) lokalizavimui vadinamas jų skaidymo į tam tikras prasminių požiūrių susietų elementų grupes, galbūt turinčias pasikartojančius elementus, ir tą grupių susiejimas su atitinkamais tuos reikalavimus, galimybes ar tikslus realizuojančiais arba tas problemas ar grėsmes gimdančiais sistemos komponentais. Skaidymą stengiamasi taip atlikti, kad kiekvienas lokalizuojamo rinkinio elementas paklūtų bent į vieną tokią grupę. Kitaip tariant, atlikus lokalizavimą turėtų paaiškėti, kokie konkretūs sistemos komponentai

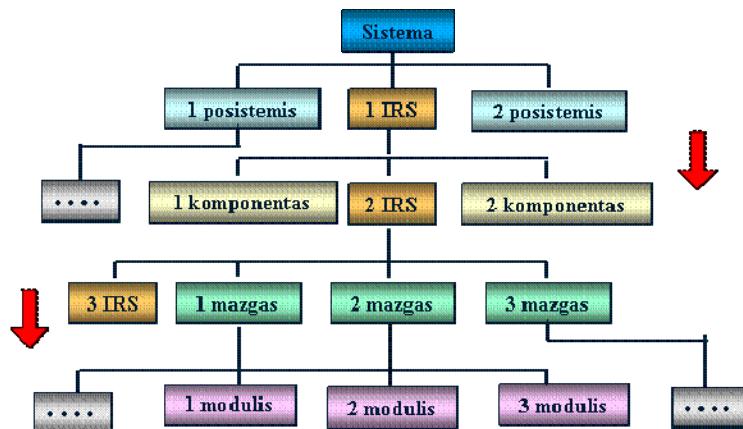
įgyvendina kiekvieną iš reikalavimų, galimybių ar tikslų arba kokiuose sistemos komponentuose slypi sistemos problemų ar jai kylančių grėsmių priežastys. Tačiau tai ne visuomet pavyksta. Reikalavimai, kuriuos pavyksta priskirti bent vienai tokiae grupėi, vadiname lokalizuojamais, o tuos, su kuriais to nepavyksta padaryti, – nelokalizuojamais.

Reikalavimų (problemų, grėsmių, galimybių, tikslų ir kt.) nuleidimu žemyn yra vadinamas jų formulavimo to lygmens, į kurį jie yra nuleidžiami, terminais. Nuleidžiant reikalavimus ar tikslus žemyn, gali tekti juos papildyti, nes gali tekti priimti tam tikrus sprendimus apie jų įgyvendinimo būdą. Kita vertus, tikslus ir nefunkcinius reikalavimus gali tekti operacionalizuoti, t. y. išreikšti juos funkcinių reikalavimų terminais.

Toliau šiame poskyryje vartosime tik terminą *reikalavimas*, turēdami omenyje, kad tai gali būti ne tik reikalavimas, bet ir problema, grėsmė, galimybė, tikslas ar dar kas nors.

5.14.3 Sistemos reikalavimų lokalizavimas ir nuleidimas žemyn

Prieš pradedant lokalizuoti verslo, informacinių, programų ar kokios nors kitos sistemos reikalavimus, jau turi būti žinomi tos sistemos komponentai, su kuriais norima susieti lokalizuojamus reikalavimus. Kitaip tariant, jau turi būti parinkta tos sistemos architektūra (žr. 5.11 poskyrį). Lokalizavimo procesas yra iteratyvus. Iš pradžių reikalavimai yra lokalizuojami aukščiausio lygmens komponentuose, po to lygmuo po lygmens leidžiamasi žemyn ir pagaliau procesas yra baigiamas lokalizavus sistemos reikalavimus tos sistemos žemiausio lygmens komponentuose. Kiekvienam lygmenyje kiekvienas aukštesnio lygmens reikalavimas turėtų būti lokalizuotas bent viename žemesnio lygmens komponente. Tačiau, kaip buvo minėta, lokalizuoti pavyksta ne visus reikalavimus. Kai kurių reikalavimų lokalizuoti negalima, nes juos realizuoja ne koks nors konkretus komponentas, o to lygmens komponentų visuma.

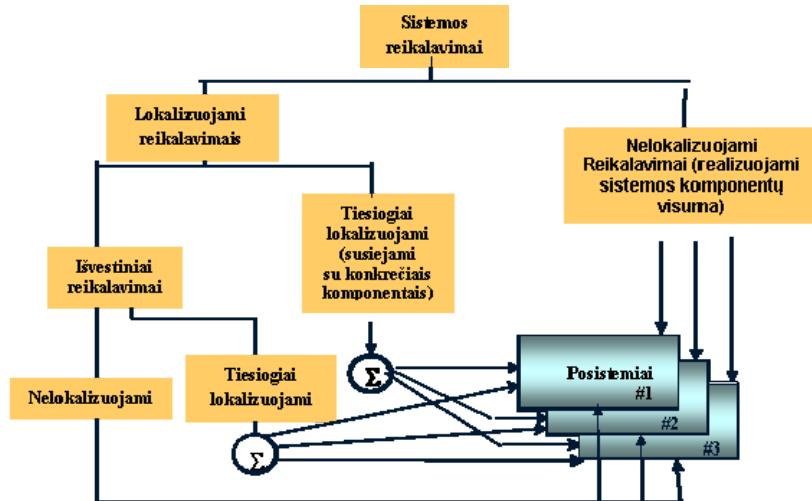


107 pav. Sistemos reikalavimų lokalizavimas ir nuleidimas žemyn (IRS – interfeisių reikalavimų specifikacija).

Tačiau susidūrus su nelokalizuojamais reikalavimais visų pirma juos reikia kruopščiai išanalizuoti. Gali būti, ir dažniausiai taip esti, kad reikalavimų nepavyko lokalizuoti ne todėl, kad jie iš tiesų yra nelokalizuojami, o todėl, kad jie buvo netinkamai suformuluoti. Tokiai atvejais reikia grįžti atgal ir patikslinti reikalavimus.

Reikalavimų lokalizavimas ir jų nuleidimas žemyn yra susipyne procesai (107 pav.). Be to, abu šie procesai yra susipyne dar ir su sistemos dekomponavimo procesu (žr. 5.11 poskyrį). Dekomponavus eilinių sistemos lygmenų į žemesnio lygmens komponentus, juose yra lokalizuojami aukštesnio lygmens reikalavimai, tačiau pradėti jų lokalizavimą dar žemesnio lygmens komponentuose galima tiktais po to, kai tie

reikalavimai jau yra nuleisti į tuos komponentus, kuriuose jie buvo lokalizuoti, ir yra suprojektuotas žemesnis sistemos lygmuo. Taigi reikalavimai yra lokalizuojami atitinkamo lygmens komponentuose, nuleidžiami žemyn, į tuos komponentus, ir tik po to pradedami lokalizuoti kito, žemesnio lygmens komponentuose.



108 pav. Sistemos reikalavimų lokalizavimas jos komponentuose

Reikalavimai gali būti lokalizuojami tiesiogiai arba netiesiogiai (108 pav.). Tiesiogiai lokalizuojamais reikalavimais vadinami tokie reikalavimai, kurie, nuleidžiant juos žemyn, iš esmės nėra keičiami. Jie tik išreiškiami konkretaus komponento terminais. Kaip jau buvo sakyta kalbant apie kokybės funkcijų sklaidos metodiką (žr. 5.13 poskyrį), tai yra požymis, jog aukštesnio lygmens reikalavimai buvo suformuluoti netinkamai ir tikriausiai atspindi žemesnio lygmens požiūri į kuriamąjį sistemą, negu iš tiesų turėtų atspindėti.

Sistemos reikalavimai

PR S01. Sistemoje turi būti numatytos priemonės prieigai prie kitų sistemų (Navision, RP/3, etc.) sukurtų failų.

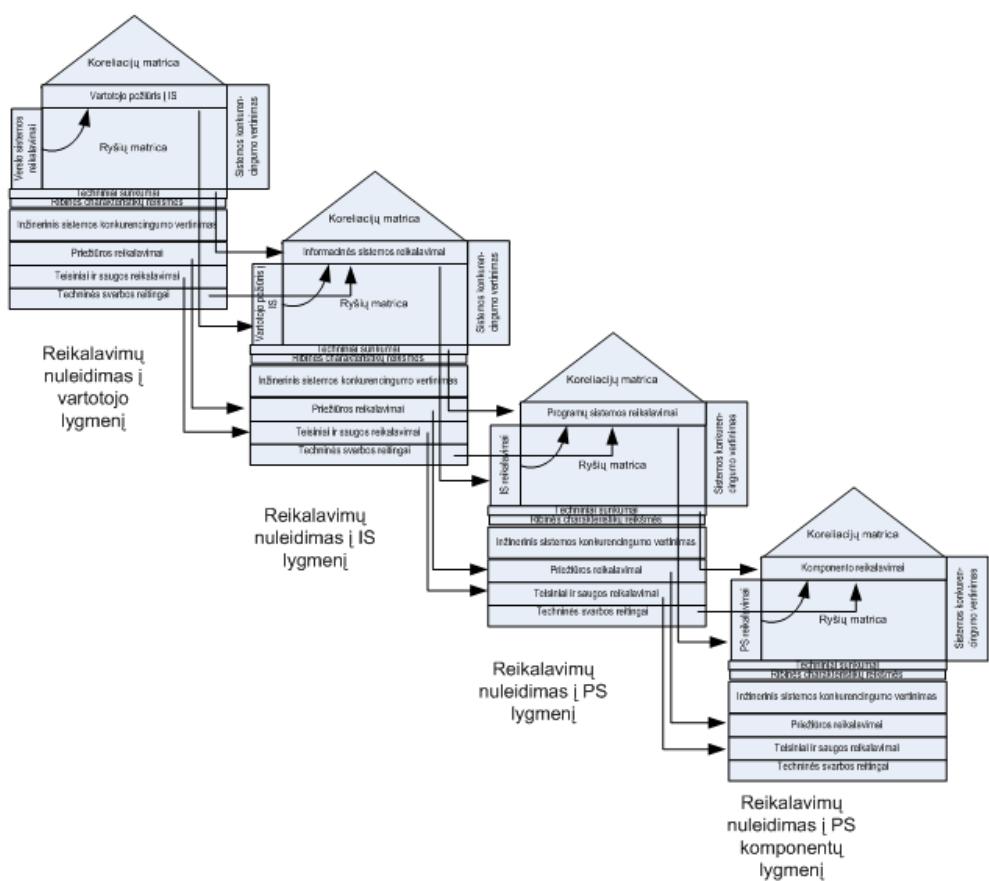
Posistemio reikalavimai

- 1PS F001. Vartotojas turi turėti galimybę nurodyti išorinio failo tipą.
- 1PS F002. Su kiekvieno tipo failu turi būti susieta jam apdoroti skirta priemonė.
- 1PS F003. Kiekvienas leistinas tipas ekrane turi būti pavaizduotas specialiai piktograma.
- 1PS F004. Vartotojas turi turėti galimybę apibrežti naujus failų tipus ir su jais susieti naujas piktogramas.
- 1PS F005. Vartotojui parinkus failą vaizduojančią piktogramą ir nurodžius failo pavadinimą, turi būti iškiesta tam failo tipui apdoroti skirta priemonė ir jai kaip parametras turi būti perduotas apdorojamo failo pavadinimas.

109 pav. Programų sistemos reikalavimų nuleidimo žemyn pavyzdys

Nuleidžiant žemyn netiesiogiai lokalizuojamus reikalavimus, jie yra keičiami iš esmės. Iš kelių aukštesniojo lygmens reikalavimų gali būti gautas vienas žemesniojo lygmens reikalavimas arba, atvirkščiai, iš vieno aukštesniojo lygmens reikalavimo gali būti gauti keli žemesniojo lygmens reikalavimai (109 pav.). Iš aukštesniojo lygmens reikalavimų gauti žemesniojo lygmens reikalavimai vadinami išvestiniiais. Gali paaiškėti, kad kai kurie iš išvestinių reikalavimų yra nelokalizuojami arba, tiksliau tariant, įgyvendinami to lygmens komponentų visumos. Kadangi nuleidžiant reikalavimus žemyn visuomet yra priimami vieni ar kiti projektavimo sprendimai, nusakantys tų reikalavimų įgyvendinimo būdą, tai greta išvestinių

reikalavimų beveik visuomet atsiranda dar *pridėtiniai*, papildomi reikalavimai, atspindintys tuos sprendimus. Pridėtiniai reikalavimai visų pirma susiję su komponentų interfeisų specifikavimu. Kaip ir visi kiti reikalavimai, sistemos interfeisų reikalavimai yra nuleidžiami į kiekvieno lygmens komponentus. Tačiau vien tik šito nepakanka, nes tuos reikalavimus dar tenka papildyti reikalavimais, kurie aprašo tas komponentų interfeisų savybes, kurios reikalingos to lygmens komponentų tarpusavio sąveikai ir sąveikai su aukštesniojo lygmens komponentais. Nors dažniausiai leidžiama tik dviejų gretimų lygmenų komponentų sąveika, bet gali būti ir kitaip. Tai priklauso nuo pasirinktos sistemos architektūros stiliaus. Tačiau bet kuriuo atveju komponentų interfeisus reikia specifikuoti ir todėl kiekviename lygmenyje greta to lygmens komponentų atsiranda dar ir komponentų interfeisų specifikacija (107 pav.).



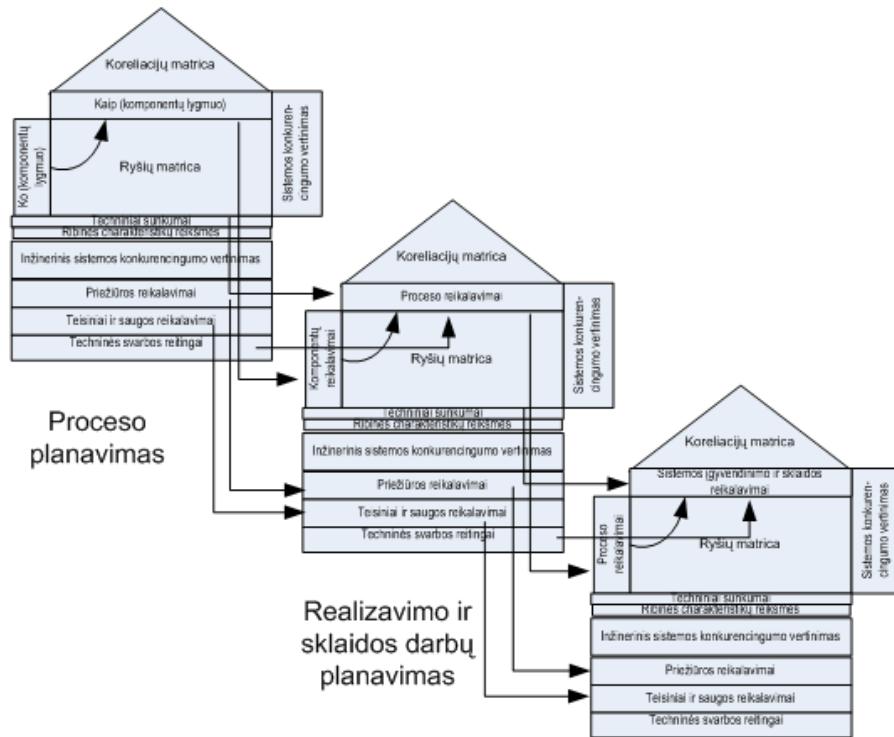
110 pav. Reikalavimų nuleidimas žemyn, naudojant KFS metodiką.

Judant sistemos hierarchijos lygmenimis žemyn, gaunami vis detalesni ir konkretesni reikalavimai. Sistemos lygmens reikalavimai yra labai bendri, žemesniųjų lygmenų reikalavimai yra vis konkretesni ir konkretesni. Pereinant nuo programų sistemos lygmens prie jos komponentų lygmens, pereinama prie projektavimo reikalavimų arba, kitaip tariant, sistemos reikalavimai virsta konkrečiomis užduotimis projektuotojams ir sistemos testuotojams. Galų gale žemiausiai lygmenyje projektavimo reikalavimai virsta konkrečiomis užduotimis programuotojams ir jos modulių testuotojams. Kaip tai yra daroma naudojant kokybės funkcijų skaidos metodiką, parodyta 110 pav.

Nuleidžiant reikalavimus žemyn kaip pagrindiniai yra naudojami du programų sistemų inžinerijos mechanizmai – dekompozicija ir abstraksija. Naudojant šiuos mechanizmus, viena vertus, sistema yra skaidoma (dekomponuojama) į vis

smulkesnius ir smulkesnius komponentus, kita vertus, abstraktūs, bendro pobūdžio sistemos reikalavimai yra vis labiau ir labiau konkretizuojami ir detalizuojami.

Be kita ko, reikalavimų nuleidimas žemyn gali būti traktuojamas ir reikalavimų analizės procesas. Nuleidžiant reikalavimus žemyn gali būti aptiktos reikalavimų lokalizavimo, sistemos architektūros parinkimo ir pradinių sistemos reikalavimų klaidos. Radus tokias klaidas, tenka grįžti atgal ir jas ištaisyti.



111 pav. Programų sistemos reikalavimų nuleidimas į programų sistemų inžinerijos proceso ir programavimo technologijos lygmenis.

Nuleidus reikalavimus į apatinį sistemos hierarchijos lygmenį, reikalavimų nuleidimo žemyn procesas paprastai dar nepasibaigia. Nuleidžiant žemyn nefunkcinius sistemos reikalavimus, yra stengiamasi juos pertvarkyti į funkcinius. Tačiau net ir pasiekus apatinį sistemos hierarchijos lygmenį ne visus nefunkcinius reikalavimus pavyksta pertvarkyti į funkcinius. Tokius reikalavimus reikia operacionalizuoti kitaip, nuleidžiant juos į programų sistemų inžinerijos proceso arba netgi į programavimo technologijos lygmenį (111 pav.). Tarkime, daugelio patikimumo reikalavimų nepavyksta operacionalizuoti kitaip, kaip papildomai padidinant vieno ar kito komponento testavimo apimtį, t. y. pertvarkant tuos reikalavimus į atitinkamus proceso reikalavimus. Programų sistemos prižiūrimumo reikalavimai dažniausiai negali būti įgyvendinti kitaip, negu tam tikru būdu rašant programų kodą. Kitap tarant, jie yra operacionalizuojami pertvarkant juos į atitinkamus programavimo reikalavimus, t. y. nuleidžiant į programavimo technologijos lygmenį.

Reikalavimai	Iš kokių aukštesnio lygmens reikalavimų gautas	Kur lokalizuotas	Kaip vertintas	Vertinimo rezultatas
P RS 01	Pradinis	1 PS	Derinant su užsakovu	Aprobuotas
...
1PS F001	P RS 01	12 KMP	Maketuojant	Aprobuotas
...
10 KMP I012	Pridėtinis	3IRS	Maketuojant	Atmestas
...

112 pav. Reikalavimų lokalizavimo matrica.

Naudojant kokybės funkcijų sklaidos metodiką, reikalavimų lokalizavimo ir nuleidimo žemyn rezultatai yra dokumentuojami ryšio, priežiūros ir kontroliuojamų reikalavimų matricomis. Kitose metodikose tam tikslui dažniausiai yra naudojama *reikalavimų lokalizavimo matrica* (112 pav.). Literatūroje ši matrica dažnai dar yra vadinama reikalavimų ryšio matrica, tačiau jos nereikėtų painioti su KFS metodikos reikalavimų ryšio matrica. Šios matricos yra visiškai kitokios.

112 pav. pateiktą pavyzdį reikėtų nagrinėti kartu su 107 pav. ir 109 pav. pateiktais pavyzdžiais. Pirmojoje matricos eilutėje parodyta, kad reikalavimas P RS 01 yra vienas iš pradinių sistemos reikalavimų. Jis buvo aprobuotas aptariant jį su užsakovu ir yra lokalizuotas pirmajame sistemos posistemyje. Reikalavimas 1PS F001 yra vienas iš reikalavimų, gautų nuleidžiant reikalavimą P RS 01 į pirmąjį sistemos posistemį (109 pav.). Jis buvo aprobuotas maketuojant šį posistemį. Reikalavimas 10 KMP I012 yra pridėtinis reikalavimas, gautas projektuojant trečiojo sistemos lygmens dešimtojo komponento interfeisą (107 pav.) ir įrašytas į to lygmens komponentų interfeisių reikalavimų specifikaciją 3IRS. Tačiau išbandžius to komponento maketą buvo nuspręsta, kad reikalavimas yra nebūtinės ir jis buvo atmestas.

6 Reikalavimų dokumentavimas

6.1 Reikalavimų specifikavimo problemos

Bent kiek sudėtingesnė programų sistema dažniausiai turi tenkinti kelis šimtus reikalavimų. Žmogus nėra pajėgus juos visus prisiminti ir operuoti jais savo galvoje. Be to, su reikalavimais dažniausiai dirba daugelis žmonių. Nedokumentavus reikalavimų negalima garantuoti, kad visi jie dirba su tuo pačiu reikalavimų rinkiniu. Tačiau ir surašius reikalavimus raštu visų problemų išspręsti nepavyksta.

Pirmoji problema yra informacijos paieškos problema. Reikalavimų aprašas paprastai esti didelis, susideda iš keliai dešimt, o kartais ir iš kelių šimtų puslapių. Susiieškoti tame reikiama informacija būna pakankamai sudėtinga. O įvairios informacijos prireikia gana dažnai, pavyzdžiui, norint pasitikslinti kurio nors reikalavimo formuluotę arba norint tą formuluotę keisti. Būtų neprotinga reikalauti, kad kiekvienas, norintis rasti kokią nors informaciją turi jos ieškoti skaitydamas visą dokumento tekštą nuo pradžios iki galo. Todėl dokumentą reikia struktūrizuoti pagal tokias taisykles, kad su juo būtų galima dirbti kaip su žinynu ir greitai tame susiieškoti bet kokią reikiama informaciją. Kitaip tariant, rengiant tokius dokumentus reikia vadovautis turinių atskyrimo principu.

Kita problema susijusi su tuo, kad dokumentą tenka dažnai taisyti ir keisti. Visada, kuomet tenka iš dokumento pašalinti kokį nors reikalavimą, o kartais ir tuomet, kuomet yra pridedamas naujas reikalavimas, dokumento tekštą arba bent jau atskiras to teksto dalis tenka perrašyti. Tai gali būti nemažas darbas. Saugant reikalavimų specifikaciją kompiuteryje, tarkime, .doc formatu, teksto perrašymo problema atkrinta, nes tekštą galima keisti lokaliai. Tačiau reikiamos informacijos paieškos problema išlieka, nes tekštų redagavimo sistemos turi gana ribotas informacijos paieškos priemones. Be to, daugelis žmonių pageidauja dirbti su išspausdintu tekstu, o ne skaityti jį kompiuterio ekrane. Taigi, pakeitus dokumentą, jį tenka spausdinti iš naujo, o jeigu reikia keliolikos to dokumento kopijų, tai irgi gali būti nemažas darbas.

Yra dar viena problema, susijusi taip pat su reikalavimų specifikacijos keitimu. Tai dokumento identifikavimo ir jo legetimumo problema. Reikala tas, kad po kurio laiko atsiranda kelios reikalavimų specifikacijos versijos ir tenka aiškintis, kuri versija yra naujesnė už kitą ir kuo jos skiriasi viena nuo kitos. Be to, gali atsitikti ir taip, kad kuo nors skiriasi skirtinges tos pačios dokumento versijos ir tada iškyla klausimas, kuri gi iš jų yra tikroji versija. Tiesa, nuo to galima apsaugoti gana paprastai, iš karto paskelbiant kurią nors vieną ir tiktai vieną dokumento kopiją originalu ir kaip nors ją apsaugant nuo klastojimų, pavyzdžiui, kam nors pasirašant kiekviename jos puslapyje.

Beveik visų minėtų problemų, išskyrus legetimumo problemą, galima išvengti laikant reikalavimus repozitorijuje ir dirbant su juo, o ne su popieriniu dokumentu ar jo kompiuterine kopija. Tačiau toks darbo stilus daugeliui žmonių vis dar yra mažai priimtinės. Jie nori dirbti su dokumentais, o ne su informacija. Be to kol kas nėra jokių patikimų būdų užtikrinti informacijos, o ne dokumento legetimumą. Yra ir kitų juridinio pobūdžio kliūčių, dėl kurių negalima visiškai atsisakyti popierinių dokumentų. Todėl, net jeigu visa informacija ir yra saugoma repozitorijuje, vis vien tenka rengti reikalavimų specifikacijas ir iškyla klausimas, kokios jos turėtų būti ir kas

jose turėtų būti parašyta. Tiesa, tokiai atvejais specifikacijos paprastai rengiamos ne rankiniu būdu, o generuoojamos automatiškai.

Reikia pabrėžti, kad paprastai esti ne viena reikalavimų specifikacija. Kaip kalbėjome 2.1 poskyryje, galima kalbėti apie verslo lygmens reikalavimus, vartotojo reikalavimus, IS reikalavimus ir skirtingus programų sistemų reikalavimų lygmenis. Net jeigu visų lygmenų reikalavimai ir nėra rengiami, vis vien reikia aprašyti kuriamą informacinę sistemą ir visas joje veikiančias programų sistemas ir tai reikia taip padaryti, kad reikalavimų aprašai būtų vienodai suprantami tiek analitikams, tiek ir visoms kitoms kuriamaja sistema suinteresuotoms šalims. Tai padaryti nėra paprasta. Be to, pageidautina, kad bet kuri reikalavimų specifikacija aprašytų tik iš išorės stebimą specifikuojamas sistemas elgseną bei kitas, nefunkcines, tos sistemas savybes. Nors šią veiklą vadiname sistemos specifikavimu, iš tiesų tai tam tikra projektavimo atmaina, iš išorės stebimos sistemas elgsenos projektavimas, be kita ko apimantis ir preliminarinį vartotojo interfeisių projektavimą. Taigi, rengiant sistemas reikalavimų specifikacijas, reikia specifikuoti taip pat ir vartotojo interfeisių reikalavimus. Tai dar vienas pavyzdys, parodantis kaip tarpusavyje susipina reikalavimų inžinerijos ir kitos programų sistemas inžinerijos veiklos. Apskritai sistemas reikalavimus galima skaidyti ne tik horizontaliai, pagal lygmenis, bet ir vertikaliai, pagal reikalavimų pobūdį – funkciniai reikalavimai, našumo reikalavimai, interfeiso reikalavimai ir kt. – ir pavesti aiškintis ir specifikuoti skirtingo pobūdžio reikalavimus skirtingoms analitikų grupėms. Dėl to dideliuose projektuose gali atsirasti daug vieno lygmens reikalavimų specifikacijų: funkcinių reikalavimų specifikacija, patikimumo reikalavimų specifikacija, našumo reikalavimų specifikacija, panaudojamumo reikalavimų specifikacija, vartotojo interfeisių reikalavimų specifikacija ir kt. Mažesnuose projektuose kiekvieno pobūdžio reikalavimams skiriama atskiras to paties dokumento skyrius ar poskyris.

Specifikuojant iš išorės stebimą sistemas elgseną arba, kitaip tariant, projektuojant tą sistemą kaip juodają dėžę, yra specifikuojamos iš išorės stebimos sistemas būsenos, aprašomi įeities duomenys, kuriami rezultatai ir duomenų bei rezultatų tarpusavio ryšiai. Tuos ryšius galima aprašyti neprocedūriškai arba procedūriškai. Neprocedūriškai duomenų ir rezultatų ryšiai dažniausiai aprašomi funkcinėmis priklausomybėmis. Kitaip tariant, paprasčiausiai yra pasakoma kokie rezultatai iš kokių duomenų yra gaunami. Aprašant tuos ryšius procedūriškai, aprašomas dar ir rezultato gavimo būdas, galbūt netgi detalus algoritmas. Bendruoju atveju, nepatariama aprašinėti ryšius procedūriškai. Taip yra dėl dviejų priežasčių:

- Procedūrines specifikacijas suprasti nėra paprasta. Jų teisingumą gali įvertinti tik ekspertai. Paprastam vartotojui dažniausiai tai yra per daug sudėtinga ir todėl jis negali tokią specifikaciją aprobuoti.
- Gana dažnai procedūrinės specifikacijos primeta projektuotojams tam tikrą problemos suvokimo būdą ir jie įgyvendina jas paraidžiui, nors tai yra visiškai neprasminka.

Vis dėlto kartais apseiti be procedūrinių specifikacijų yra neįmanoma, nes duomenų ir rezultatų ryšius kitais būdais nepavyksta suprantamai aprašyti. Tam tikra alternatyva yra specifikuoti ryšius pateikiant konkretius pavyzdžius. Tačiau toks ryšių specifikavimo būdas taip pat nėra geras, nes reikalaujama, kad projektuotojas gebėtų apibendrinti pateiktus pavyzdžius ir iš jų „išstruktū“ atitinkamą algoritmą. Jei pavyzdžių sistema nėra išsami, gali būti apskritai neįmanoma tai padaryti. Kita vertus, pavyzdžiais papildytose specifikacijose neabejotinai yra lengviau suprantamos. Todėl

pavyzdžius pateikti verta, nepriklausomai nuo to, kaip yra specifikuojami duomenų ir rezultatų tarpusavio ryšiai, procedūriškai ar neprocedūriškai.

Dar viena reikalavimų specifikavimo problema yra reikalavimų konkretizavimo problema. Tačiau šią problemą mes jau aptarėme 5.4 poskyryje. Kaip kalbėjome tame poskyryje, žemiausias specifikacijų abstrakcijos lygmuo yra vadinamas fiziniu. Kadangi fizinio lygmens specifikacijos aprašo visas iš išorės stebimas kuriamos sistemos detales, jų apimtis yra pati didžiausia. Be to jos yra ir dažniausiai keičiamos. Todėl fizinio lygmens specifikacijoms parengti ir palaikyti reikia didžiausių darbo sąnaudų. Taigi gana dažnai, ypač dideliuose projektuose, tiek darbo idėti yra neįmanoma, nes būtų pažeisti visi priimtini projekto terminų ir kainos ribojimai, ir todėl fizinio lygmens reikalavimų specifikacijos apskritai nėra rašomas, reikalavimai fiksuojami tik projektuotojų ir programuotojų darbiniuose užrašuose bei galvose. Kita vertus, suinteresuotąsiams šalis, ypač būsimuosius sistemas vartotojus, dažniausiai domina ne tik loginio, bet ir fizinio lygmens specifikacijos ir visas detales su jais reikia suderinti bent jau žodžiu. Kyla klausimas, kas turi tai daryti. Analitikai to padaryti negali, nes jie patys detalių nežino. Projektuotojai ir programuotojai taip pat to padaryti negali, nes jie užsiémę visai kuo kitu, o be to jie dažniausiai ir nemoka bendrauti su vartotojais ar užsakovais. Be to, po kurio laiko bet kurią sukurtą sistemą tenka perdirbinėti, tobulinti, keisti. Nėra jokių garantijų, kad iki to laiko dalis projektuotojų bei programuotojų, kurių darbiniuose užrašuose bei galvose buvo „surašytos“ fizinio lygmens reikalavimų specifikacijos tebedirbs sistemą kūrusioje organizacijoje. Todėl sistemą perdirbinėjantiems ar tobulinantiems žmonėms vėl gali tekti tyrinėti pačią sistemą, stengiantis išsiaiškinti tos sistemos fizinio lygmens reikalavimus. Vargu ar tokiais atvejais labai būtų naudinga ir keliolikos tūkstančių puslapių apimties fizinio lygmens reikalavimų specifikacija, netgi jeigu ji būtų buvusi parengta ir išlikusi korekтиška. Ką daryti? Niekas nežino. Gero problemos sprendimo paprasčiausiai nėra ir kiekvienoje organizacijoje ar netgi kiekviename projekte ji yra sprendžiama savaip, priklausomai nuo konkrečių aplinkybių. Tačiau visą laiką reikia turėti omenyje, kad netgi ir geriausiai parengtos reikalavimų specifikacijos dažniausiai negali išsamiai dokumentuoti visų kuriamos sistemos detailių, ir nereikia galvoti, kad parengus reikalavimų specifikacijas galima nustoti bendrauti su užsakovais bei vartotojais ir toliau dirbtis savarankiškai. Jei įmanoma, bendravimas turėtų vykti visą projekto vykdymo laiką.

Paskutinė svarbi reikalavimų specifikavimo problema yra reikalavimų specifikavimo būdo parinkimas. Reikalas tas, kad reikalavimus galima aprašyti labai skirtingais būdais:

- natūraliaja kalba,
- kokia nors labiau ar mažiau griežtą struktūrą turinčia kalba, pavyzdžiui, lentelėmis arba kokiu nors pseudokodu,
- kokiais nors grafiniais modeliais, aprašančiais rezultatų gavimo iš duomenų procesus, sistemos būsenas ir jų kaitą, duomenų struktūras ir jų tarpusavio ryšius, duomenų srautus arba, jei modelis yra objektinis, objektų klasses ir jų tarpusavio ryšius,
- griežtą semantiką turinčiomis matematinėmis arba loginėmis kalbomis.

Griežtą semantiką turinčios matematinės ir loginės reikalavimų specifikavimo kalbos vadinamos formaliosiomis specifikavimo kalbomis. Tokiomis kalbomis galima parašyti griežtesnes ir tikslesnes funkcinės specifikacijas, bet labai sunku specifikuoti

kai kuriuos nefunkcinius reikalavimus. Be to, tiktai nedaugelis programų sistemas kuriančių specialistų ir dar mažiau vartotojų ar užsakovų yra susipažinę su tokiomis kalbomis ir geba jas korektiškai vartoti. Todėl daugumoje projektų pats praktiškiausias reikalavimų aprašymo būdas yra aprašymas vienaip ar kitaip struktūruotomis natūraliosiomis kalbomis papildant tuos aprašus pavyzdžiais ir grafiniais modeliais. Taip yra, nepaisant visų gerai žinomų trūkumų (dviprasmybės, aprašo neekonomiškumas ir kt.), būdingų natūraliosioms kalboms.

Taigi apibendrinant galima pasakyti, kad rengiant reikalavimų specifikacijas tenka spręsti šias pagrindines problemas:

- konkrečiam atvejui labiausiai tinkamo rengiamų specifikacijų rinkinio parinkimas ir tų specifikacijų turinio nustatymas,
- greitai informacijos paieškai pritaikytos dokumento struktūros parinkimas, kiekvienam iš rengiamų dokumentų,
- kiekvieno rengiamo dokumento organizavimas tokiu būdu, kad pataisymus ir pakeitimius tame dokumente būtų galima daryti mažiausiomis darbo sąnaudomis,
- dokumentų identifikavimas ir apsauga nuo nesankcionuotų pakeitimų (legetimumo užtikrinimas),
- sistemos įrenginių duomenų ir jos rezultatų tarpusavio ryšių specifikavimo būdo parinkimas,
- specifikavimo abstrakcijos lygmenų nustatymas, reikalavimų nuleidimas iš vieno lygmens į kitą, fizinio lygmens specifikacijų problema,
- reikalavimų specifikavimo būdo parinkimas.

6.2 Reikalavimų dokumentavimo standartai

Kokio nors vieno visų pripažinto reikalavimų dokumentavimo būdo nėra. Nėra ir visų pripažinto dokumentų rinkinio. Yra priimta keletas IEEE reikalavimų dokumentavimo standartų [ST3, ST9, ST12] ir gana daug nacionalinių standartų, pavyzdžiui, [ST21, ST32, ST36, ST37]. Trumpa svarbiausiųjų standartų suvestinė pateikta 6 lentelėje.

6 lentelė. Svarbiausieji reikalavimų dokumentavimo standartai.

Standartas	Kas Jame aprašyta
IEEE 830-1998 [ST3]	Aprašyta grubi programų sistemos reikalavimų specifikacijos struktūra. Ją galima detalizuoti. Pateikta keletas tos struktūros versijų.
IEEE 1233-1998 [ST9]	Aprašo programinę įrangą naudojančių sistemų reikalavimų specifikacijos struktūrą.
MIL STD 498 [ST21] ¹⁶	Aprašo operacinių poreikių, sistemos reikalavimų, programų sistemos reikalavimų ir interfeiso reikalavimų specifikacijų struktūrą. Tebėra labai populiarūs.
ESA (European Space	Aprašo vartotojo reikalavimų, sistemos reikalavimų,

¹⁶ Standartas nebegalioja. Jis buvo pakeistas standartu [ST9]. Tačiau jame aprašyta dokumentų struktūra tebėra labai populiari ir dažnai naudojama.

Standartas	Kas jame aprašyta
Agency) [ST34]	interfeiso reikalavimų, testo reikalavimų ir kai kurių kitų specifikacijų struktūrą.
Reikalavimų inžinerijos proceso modelio „Volere“ standartas [95]	Aprašo dokumento, jungiančio kelias skirtingas specifikacijas, struktūrą ir reikalavimo aprašo struktūrą.
CEFACT/ICG/005 [ST39]	Aprašo verslo reikalavimų specifikacijos struktūrą.

Tačiau jie yra gana skirtingi, o ir kuriuo nors vienu iš jų toli gražu nevisi vadovaujasi. Beveik kiekvienam reikalavimų inžinerijos vadovėlyje dėstomas savas, be abejo geriausias, dokumentavimo būdas ir siūlomas savas reikalavimams aprašyti skirtų dokumentų rinkinys. Skirtingos organizacijos reikalavimus dokumentuoja taip pat skirtingai.

Vis dėlto visi sutaria, kad dokumentuoti reikia visų lygmenų reikalavimus, t. y.:

- verslo lygmens reikalavimus;
- vartotojo lygmens reikalavimus;
- sistemos lygmens reikalavimus;
- programinės įrangos lygmens reikalavimus.

Kaip vadinti konkretaus lygmens reikalavimus aprašantį dokumentą ir kokia turi būti to dokumento struktūra nesutariama, tačiau daugmaž sutariama dėl to, kokia informacija ten turėtų būti pateikta. Nėra būtina kiekvieno lygmens reikalavimus dėti į atskirą dokumentą. Nedideliuose projektuose visų lygmenų reikalavimams aprašyti gali pakakti ir vieno dokumento, kiekvienam lygmeniui skiriant atskirą dokumento skyrių. Gali būti ir atvirkščiai. Dideliuose projektuose vieno lygmens reikalavimams aprašyti gali prieikti kelių skirtingų dokumentų. Taigi neįmanoma pasakyti, kokius konkrečius dokumentus reikėtų parengti atliekant reikalavimų inžinerijos darbus. Aišku, vadovaujantis turinių atskyrimo principu, galima pasakyti, kokius dokumentus reikėtų parengti ir koks turėtų būti tų dokumentų turinys (žr. 7 lentelę). Tačiau konkrečiame projekte tie dokumentai gali būti įvairiais būdais jungiami vienas su kitu ir turėti pačius įvairiausius pavadinimus.

7 lentelė. Reikalavimų inžinerijos proceso rezultatai.

Specifikacija	Turinys
Verslo poreikių	Vizija, tikslų medis, jo ribojimai, verslo objektai, pareigybinių teisės ir įgaliojimai, darbo vietas, laiko ribojimai.
Operacinių poreikių	Operaciniai poreikiai, vartotojų norai ir pageidavimai, ribojimai.
IS, PS, komponentų reikalavimų	Išorinio projektavimo rezultatai.
Papildomų reikalavimų	Kai kurie procesai (pvz., RUP) reikalauja, kad nefunkciniai (IS, PS, komponento) reikalavimai ir projekto reikalavimai būtų dokumentuojami papildomomis specifikacijomis, siejamomis su atitinkamų funkcinių reikalavimų specifikacijomis.

Specifikacija	Turinys
Interfeiso reikalavimų	Rengiama IS, PS ir komponentams, specifikuojami loginio lygmens reikalavimai.
Testo reikalavimų	Reikalavimų specifikacijos rengiamos ne tik testams, bet ir tikrinimams, atliekamiems verifikujant ir vertinant reikalavimus. Aprašo, kas turi būti patikrinta.
Projekto reikalavimų	Terminai, biudžetas, atsiskaitymas, kontroliniai taškai ir kiti su projekto vadyba susiję reikalavimai.
Komponentų hierarchijos	Aprašo IS, PS arba komponento dekompozicijos rezultatus.
Dalykinės srities terminų žodynas (ontologija)	Terminų apibrėžtys ir sąryšiai.
Prielaidų, kitų su reikalavimais susijusių klausimų, rizikos veiksnių ir su reikalavimais atliekamų darbų plano	Tiek reikalavimai, tiek ir testai gali būti grindžiami tam tikromis prielaidomis, kartais reikia dokumentuoti kitus su jais susijusius klausimus. Visa tai transformuojama į rizikos veiksnius. Jiems pašalinti planuojami reikalavimų verifikavimo ir vertinimo darbai.

6.3 Verslo poreikių specifikacija

Verslo poreikių specifikacija <Trm88> – tai dokumentas, skirtas verslo lygmens reikalavimams aprašyti. Jis turi keletą angliskų pavadinimų ir yra rengiamas pagal keletą skirtingų šablonų. Vadovaujantis 2.1 ir 2.4 poskyriuose išdėstyta metodika ir kuriant sistemą konkrečiam užsakovui, šiame dokumente turėtų būti pateikta tokia informacija:

- verslo misija, nuostatos, taisyklės ir vizija;
- verslo vizijos įgyvendinimo būdą aprašantis tikslų medis ir jo ribojimai;
- verslo tikslams pasiekti naudojamų verslo objektų reikalavimai;
- verslo tikslų susiejimas su vykdytojais, vykdytojų įgaliojimai;
- verslo tikslų susiejimas su darbo vietomis;
- verslo našumo reikalavimai.

Kuriant sistemą tikslu pardavinėti ją rinkoje, dokumente turėtų būti pateikiama šiek tiek kitokia informacija:

- rinkos segmento, kuriam skiriamas kuriamasis produktas, aprašas, to produkto pranašumai ir jo vizija;
- produkto vizijos įgyvendinimo būdą aprašantis produkto galimybų medis ir jo ribojimai;
- realaus pasaulio objektų, su kuriais arba su kurių informaciniais modeliais dirba (t. y. juos valdo, apdoroja, kuria, analizuja ar kt.) kuriamasis produktas, aprašas;
- produkto galimybų susiejimas su potencialiomis vartotojų grupėmis ir tų grupių teisių bei įgaliojimų aprašas;
- produkto galimybų susiejimas su darbo vietomis, kuriose tomis galimybėmis galima pasinaudoti;
- produkto galimybų našumo reikalavimai.

1. Introduction 1.1 Document purpose 1.2 Business objectives and success metrics 1.3 Definitions, acronyms and abbreviations 1.4 References and supporting documentation 2. Overall solution description 2.1 Constraints and limitations 2.2 The business opportunity and value proposition 2.3 The primary customers 2.4 Assumptions, constraints, and risks 3. Use cases 5. Requirements 5.1 Information requirements 5.2 Licensing requirements 5.3 Legal, copyright, and other notices 5.4 Security and confidentiality requirements Appendices	1. Įvadas 1.1 Dokumento paskirtis 1.2 Verslo tikslai ir sekės matai 1.3 Terminų žodynas 1.4 Naudojami dokumentai ir kiti šaltiniai 2. Siūlomos sistemos bendrasis aprašas 2.1 Ribojimai 2.2 Verslo galimybės ir naujos vertės 2.3 Tiesioginiai klientai 2.4 Priešlaidos, ribojimai ir rizikos veiksnių 3. Verslo transakcijos 5. Reikalavimai 5.1 Informacinių reikalavimų 5.2 Licencijavimo reikalavimai 5.3 Juridiniai, autorinių teisių apsaugos ir kiti klausimai 5.4 Apsaugos ir konfidencialumo reikalavimai Priedai
---	--

113 pav. IPTC standarto [ST38] nustatyta verslo reikalavimų specifikacijos struktūra.

1. Preamble 2. References 3. Objective 4. Scope 5. Business requirements 5.1 Business process elaboration 5.2 Information flow definition 5.3 Information model definition 5.4 Business rules 4. Definition of terms Appendices	1. Įvadas 2. Naudojami dokumentai 3. Verslo tikslai 4. Verslo sistemos apimtis 5. Verslo reikalavimai 5.1 Verslo procesų detalizavimas 5.2 Informacijos struktūrai 5.3 Informacinių modelių 5.4 Verslo taisyklės 4. Terminų žodynas Priedai
---	---

114 pav. CEFACT/ICG/005 standarto [ST39] nustatyta verslo reikalavimų specifikacijos struktūra.

Šiuos reikalavimus tenkinančių tarptautinių verslo reikalavimų specifikacijos standartų kol kas parengta nėra. Yra keletas žinybinių standartų, pavyzdžiui, IPTC standartas [ST38] ir CEFACT/ICG/005 standartas [ST39]. Trumpai apžvelgsime kaip šie standartai traktuoją verslo reikalavimų specifikaciją. Pirmojo standarto nustatyta verslo reikalavimų specifikacijos struktūra parodyta 113 pav., antrojo – 114 pav. Abi jos pateiktos anglų ir lietuvių kalbomis. Taip padaryta dėl to, kad dėl lingvistinių sistemų skirtumų, ontologinių skirtumų ir skirtingų dokumentalistikos tradicijų angliskąjį dokumento skyrių ir poskyrių pavadinimų esmę perteikti lietuvių kalba yra labai sudėtinga. Todėl skaitytojui paliekama galimybė pačiam palyginti abu tekstus ir spręsti apie pateikto vertimo kokybę. Taip bus daroma ir toliau, pateikiant kitų specifikacijų struktūrą. Beje, reikia padaryti dar vieną pastabą. Šiame skyriuje daug kalbama apie įvairius dokumentavimo standartus ir yra aiškinama jais nustatomų dokumentų struktūra. Tačiau šių paaiškinimų jokiu būdu nereikėtų traktuoti kaip tikslų ir išsamių minimų standartų turinio aprašų ir tiesiogiai jais remtis. **Nusprendus naudoti kurį nors standartą, su jo turiniu būtina detaliai susipažinti patiemis.**

Grįžtant prie verslo reikalavimų specifikacijų struktūros, galima pasakyti, kad 113 pav. ir 114 pav. pateiktos specifikacijų struktūros turi daug panašumų. Abiejose aprašomi verslo tikslai ir verslo sistemos apimtis (pirmajame dokumente ji aprašoma aprašant verslo transakcijas), formuluojami verslo reikalavimai. Patys reikalavimai jau skiriasi. Pirmasis dokumentas yra daugiau pritaikytas sistemoms kuriamoms tikslu jas parduoti rinkoje. Tačiau nei vienas iš dokumentų neapima visos informacijos, kurių reikštę pateikti, kad būtų galima sklandžiai pereiti prie vartotojo lygmens reikalavimų aiškinimosi, formulavimo ir specifikavimo.

6.4 Operacinių poreikių specifikacija

Operacinių poreikių specifikacija <Trm35> – tai dokumentas, skirtas vartotojo lygmens reikalavimams aprašyti. Kaip ir verslo poreikių specifikacija, jis turi keletą anglų pavadinimų ir yra rengiamas pagal skirtingus šablonus. Dokumentas aprašo pačius bendriausius kuriamos sistemos reikalavimus, suformuluotus vadovaujantis dalykinės srities specialistų požiūriu į tai, kokia ta sistema turėtų būti. Be abejo, jis neturi prieštarauti verslo poreikių specifikacijai, kurioje išdėstytais verslo savininkų ar organizacijos vadovų aprobuotas verslo konsultantų požiūris. Gana dažnai dokumentas yra taip rengiamas, kad baigus kurti sistemą ji galima būtų naudoti kaip bendrajį tos sistemos aprašą, t. y. dokumentą, skirtą susipažinti su sistema žmonėms, kurie ketina tą sistemą įsigyti arba pradeti ja naudotis. Todėl dokumentas turi būti parašytas kalba, kuria kalba dalykinės srities specialistai, t. y. vartojant dalykinės srities terminiją. Šiame dokumente funkciniai sistemos reikalavimai aprašomi kartu su jos paskirtimi, jos naudojimo aplinka ir jos nefunkciniais reikalavimais. Dokumente gali būti pateikti sistemos veikimo kontekstą iliustruojantys konцепcinių modeliai, jos panaudojimo scenarijai, įvesties ir išvesties duomenų, principinių esybių bei darbo srautų aprašai.

Vadovaujantis 2.1 ir 2.4 poskyriuose išdėstyta metodika ir kuriant sistemą konkrečiam užsakovui, šiame dokumente turėtų būti pateikta tokia informacija:

- kokius patobulintus, lyginant su buvusiais, verslo procesus palaiko sistema;
- kokius informacinius, skaičiavimo, komunikavimo ir kitus operacinius dalykinės srities specialistų poreikius tenkina sistema ir kaip ji tuos poreikius riboja;
- kokio kompiuterinio raštingumo dalykinės srities specialistams pritaikyta sistema ir kokius panaudojamumo reikalavimus ji tenkina;
- iš kokių darbo vietų galima naudotis sistema;
- kokį verslo užduočių vykdymo našumą užtikrina sistema.

Kuriant sistemą tikslu pardavinėti ją rinkoje, dokumente aprašyta:

- sistemos vykdomos užduotys;
- sistemos teikiamos informacinių, skaičiavimo, komunikavimo ir kitos paslaugos, naudojamos toms užduotims realizuoti;
- kokius konceptinius informacių objektų, su kuriais dirba produktas, reikalavimus, įskaitant tų objektų apsaugą, tenkina sistema;
- kokio kompiuterinio raštingumo vartotojams yra pritaikytas produktas ir kokius panaudojamumo reikalavimus jis tenkina;
- iš kokių darbo vietų galima naudotis sistema;
- kokius užduočių vykdymo našumo reikalavimus tenkina sistema.

Vienas iš standartų, standartizuojančių šio dokumento struktūrą, yra IEEE Std 1362-1998 standartas [ST12]. Trumpai aptarsime šio standarto esmę.

IEEE Std 1362-1998 standarto požiūriu, operacinių poreikių specifikacija gali būti rengiama ne tik sistemos, bet ir programų sistemos lygmenyje. Vadovaujantis šituo požiūriu, 1 lentelę (žr. 2.1 poskyri) reikėtų papildyti dar vienu lygmeniu – programų sistemos vartotojo reikalavimais. Šis lygmuo būtų įterptas tarp IS reikalavimų ir PS reikalavimų.

Bet kuriuo atveju dokumentas turi aprašyti vartotojų operacinius poreikius, nesigilindamas į technines detales, kurios turi būti analizuojamos sistemos arba programų sistemos reikalavimų specifikacijose (žr. 6.5 ir 6.6 poskyrius). Poreikiai turi būti taip aprašyti, kad patikrinti, ar sukurtoji sistema juos tenkina, galėtų bet kuris

savo darbą išmanantis dalykinės srities specialistas. Be to iš jo neturi būti reikalaujama kokių nors specialių techninių žinių. Dokumente gali būti aprašyti ne tik operaciniai poreikiai, bet ir vartotojų norai, lūkesčiai bei vizijos ir tai neprivalo būti padaryta kiekybiškai, t. y. taip, kad jų įgyvendinimą būtų galima patikrinti konkrečiais testais. Panašiai kaip ir kokybės funkcijų skaidos metodikoje (žr. 5.13 poskyri), vartotojas gali, pavyzdžiu, teigti, kad jam reikalinga *didelio patikimumo* sistema, neaiškindamas ką gi iš tiesų tai reiškia. Tačiau šis poreikis privalo būti motyvuotas ir pagrįstas. Kiekybiškai jis turi būti suformuluotas sistemos arba programų sistemos specifikacijoje arba, skelbiant konkursą sistemai įsigytį, pasiūlyme dalyvauti konkurse <Trm39>.

Operacinių poreikių specifikacijoje vartotojams taip pat leidžiama pateikti savo nuomonės apie tai, kaip turi būti tenkinami vieni ar kiti operaciniai poreikiai. Sistemos reikalavimų specifikacijoje (žr. 6.5 poskyri) ir programinės įrangos reikalavimų specifikacijose (žr. 6.6 poskyri) šitos nuomonės virsta projektavimo bei technologiniai ribojimais.

1. Scope 1.1 Identification 1.2 Document overview 1.3 System overview 2. Referenced documents 3. Current system or situation 3.1 Background, objectives, and scope 3.2 Operational policies and constraints 3.3 Description of the current system or situation 3.4 Modes of operation for the current system or situation 3.5 User classes and other involved personnel 3.6 Support environment 4. Justification for and nature of changes 4.1 Justification of changes 4.2 Description of desired changes 4.3 Priorities among changes 4.4 Changes considered but not included 5. Concepts for the proposed system 5.1 Background, objectives, and scope 5.2 Operational policies and constraints 5.3 Description of the proposed system 5.4 Modes of operation 5.5 User classes and other involved personnel 5.6 Support environment 6. Operational scenarios 7. Summary of impacts 7.1 Operational impacts 7.2 Organizational impacts 7.3 Impact during development 8. Analysis of the proposed system 8.1 Summary of improvements 8.2 Disadvantages and limitations 8.3 Alternatives and trade-offs considered 9. Notes Appendices	1. Įvadas 1.1 Pavadinimas 1.2 Dokumento apžvalga 1.3 Sistemos apžvalga 2. Naudojamų dokumentai 3. Esama sistema 3.1 Pagrindimas, tikslai ir apimtis 3.2 Operacinių nuostatos ir ribojimai 3.3 Esamos sistemos aprašas 3.4 Esamos sistemos naudojimo būdai 3.5 Vartotojai ir kitas su sistema susijęs personalas 3.6 Palankanti aplinka 4. Pokyčių pagrindimas ir jų pobūdis 4.1 Pokyčių poreikis 4.2 Pageidaujamų pokyčių aprašas 4.3 Pokyčių prioritetai 4.4 Nagrinėti, bet neapriboti pokyčiai 5. Siūlomos sistemos koncepcija 5.1 Pagrindimas, tikslai ir apimtis 5.2 Operacinių nuostatos ir ribojimai 5.3 Siūlomos sistemos aprašas 5.4 Siūlomos sistemos naudojimo būdai 5.5 Vartotojai ir kitas su sistema susijęs personalas 5.6 Palankanti aplinka 6. Sistemos naudojimo scenarijai 7. Poveikis 7.1 Operacinis poveikis 7.2 Organizacinis poveikis 7.3 Poveikis per plėtrą 8. Siūlomos sistemos analize 8.1 Patobulinimai 8.2 Ribotumas ir trūkumai 8.3 Alternatyvų ir kompromisių nagrinėjimas 9. Pastabos Priedai Žodynas
--	---

115 pav. IEEE 1362-1998 standarto [ST12] nustatyta operacinių poreikių specifikacijos struktūra.

Operacinių poreikių specifikacijos struktūra pateikta 115 pav. Kaip ir kitų specifikacijų struktūra, ji pateikta anglų ir lietuvių kalbomis (kodėl taip yra daroma žr. 6.3 poskyri).

Dokumento įvade pateikiami sistemos pavadinimas ir jo trumpinys (1.1 poskyris), pateikiamas dokumento santrauka ir parašoma kokiai auditorijai jis yra skirti (1.2 poskyris) ir trumpai pristatoma kuriamoji sistema (1.3 poskyris). Antrajame dokumento skyriuje pateikiami dokumentų, į kuriuos daromas nuorodos duotajame dokumente, bibliografiniai aprašai.

Trečiasis ir penktasis dokumento skyriai yra skirti dabar esamai ir kuriamai informaciniems sistemoms aprašyti. Abu šie skyriai turi tą pačią struktūrą ir abiejuose sistema yra aprašoma iš vartotojo požiūrio taško. Pirmajame poskyryje yra pateikiamas pagrindimas, kodėl sistema yra būtent tokia ir aprašomi tos sistemas palaikomo verslo misija, tikslai ir tų tikslų įgyvendinimo strategijos bei taktikos. Taip pat yra nusakoma sistemos apimtis. Tai daroma trumpai aprašant sistemos veikimo

režimus, jos vartotojų klasses ir sąveikos su aplinka interfeisus. Visa tai turi būti detalizuota kituose šio skyriaus poskyriuose.

Antrajame poskyryje aprašomos operacinės nuostatos, kuriomis grindžiama sistema, ir ją ribojantys verslo ribojimai. Aprašomi sistemos palaikomi verslo procesai, jos tenkinami operacioniai poreikiai, kvalifikacioniai vartotojų reikalavimai, darbo vietas ir sistemos užtikrinamas verslo užduočių vykdymo našumas.

Trečiąjame poskyryje pateikiama išsami informacinės sistemos savybių specifikacija. Ji apima:

- sistemos operacinės aplinkos aprašą;
- sistemos architektūros aprašą;
- sistemos sąveikos interfeisų su išorinėmis sistemomis, įrenginiai ar procesais aprašą;
- sistemos funkcionalumo aprašą;
- įeities ir išeigos duomenų bei valdymo srautų aprašą;
- eksploatavimo kaštų aprašą;
- sistemos naudojimo rizikos veiksnių aprašą;
- sistemos našumo charakteristikų specifikacijas;
- sistemos kokybės charakteristikų (prieinamumas, korektiškumas, efektyvumas, plečiamumas, lankstumas, interoperabilumas, prižiūrimumas, perkeliamumas, patikimumas, tiražuojamumas, aptarnaujamumas, gyvybingumas, panaudojamumas) specifikacijas;
- apsaugos nuo saugos, apsaugos, privatumo, integralumo pažeidimų ir sistemos galimybių testi darbą ekstremaliose situacijose specifikacijas.

Ketvirtajame poskyryje aprašomi galimi sistemos veikimo režimai: normaliose situacijose, susidarius kokioms nors išimtinėms situacijoms, treniravimo režimu, dirbant su sistema jos aptarnavimo tarnyboms ir kt.

Penktajame poskyryje aprašomi sistemos vartotojų ir kitų su ja dirbančių asmenų (pvz., sistemos administratoriai) kvalifikacioniai reikalavimai bei jų vaidmenys. Taip pat yra aprašoma organizacinė sistemos aplinka.

Paskutiniame poskyryje aprašoma sistemą palaikanti aplinka (techninė įranga, sisteminė programinė įranga, sistemos eksploatavimo nuostatai, vartotojo dokumentacija ir pan.).

Ketvirtasis dokumento skyrius skirtas motyvacijai, kodėl seną sistemą reikia keisti nauja. Čia aprašomi senosios sistemos ribojimai ir trūkumai, kuriuos pašalins naujoji sistema, pokyčių poveikis verslo sistemos misijai, tikslams ir jų įgyvendinimo strategijoms bei taktilioms ir naujos verslo darymo galimybės, atsirandančios įdiegus naujaą sistemą (4.1 poskyris). Toliau aprašomi pageidaujami operacioniai, funkcionalumo, galingumo, interfeisų, panaudojamumo, aptarnavimo bei kiti pokyčiai (4.2 poskyris), nurodomi tų pokyčių prioritetai (4.3 poskyris) ir aprašomi nagrinėti, bet dėl vienų ar kitų priežasčių neaprobuoti pokyčiai (4.4 poskyris). Skyrius baigiamas (4.5 poskyris), aprašant prielaidas, kuriomis remiamasi siūlant tokius esamos sistemos pokyčius, ir ribojimus, kurių negalima pažeisti pereinant nuo senos sistemos prie naujos.

Šeštajame dokumento skyriuje aprašomi naujos sistemos naudojimo scenarijai. Šie scenarijai žingsnis po žingsnio aprašo sistemos veikimą ir jos sąveiką su vartotojais bei kitais išoriniais agentais (įrenginiai, procesais ir kt.). Reikalaujama taip aprašyti scenarijus, kad peržiūrinėdamas juos skaitytojas suvoktų atskirų sistemos komponentų veikimą ir sąveiką. Scenarijais galima aprašyti ir tai, ko sistema privalo nedaryti.

Septintasis dokumento skyrius yra skirtas naujos sistemos diegimo pasekmėms aprašyti. Yra aprašoma, kaip sistema pakeis esamus darbo būdus (7.1 poskyris) ir esamas organizacines struktūras (7.2 poskyris) bei ką reikia padaryti prieš diegiant sistemą arba jos kūrimo ir diegimo metu (7.3 poskyris), pavyzdžiu, apmokyti darbuotojus, išrengti kompiuterių tinklą, susitikinėti su analitikais, aiškintis su jais sistemos reikalavimus ir kt.

Aštuntajame dokumento skyriuje aprašoma, kokias naujas galimybes turės nauja sistema, kokios senos sistemos galimybės joje bus patobulintos ir kokių senos sistemos galimybės joje nebebus (8.1 poskyris). Jame taip pat aprašomi naujos sistemos trūkumai, išskaitant galimybes, kurių ji neturės, nors ir būtų naudingos (8.2 poskyris). Skyrius baigiamas poskyriu (8.3 poskyris), kuriamo aptariamos kuriamos sistemos alternatyvos ir pagrindžiama, kodėl buvo nuspresta pasirinkti būtent tokį sistemos variantą.

Devintajame dokumento skyriuje pateikiama visos pastabos apie kuriamąją sistemą, kurias norima pateikti, bet kurioms neatsirado vienos kituose dokumento skyriuje.

Baigiant reikia pasakyti, kad IEEE 1362-1998 standarte jaučiama stipri struktūrinės metodikos, ypač knygos [25], įtaka. Matyt, jis jau šiek tiek paseno ir nebelabai dera su kituose mūsų knygose skyriuose dėstomomis mintimis.

6.5 Sistemos reikalavimų specifikacijos

Programų sistemos paprastai veikia kitose, aukštesnio lygmens sistemoje. Aukštesnio lygmens sistemas sudaro ne tik programos, bet ir kitokie komponentai. Pavyzdžiu, techninėse sistemoje tokiai komponentai yra programinės įrangos valdoma aparatūra arba technologiniai procesai, organizacijų informacinėse sistemoje – rankinio darbo procedūros ir pan. Todėl sistemos reikalavimai <Trm74> paprastai yra atskiriami nuo tos sistemos programinės įrangos reikalavimų. Jie aprašomi skirtinguose dokumentuose. Bendruoju atveju sistemos, mūsų nagrinėjamu atveju – informacinės sistemos, reikalavimų specifikacija <Trm75> aprašo ne tik tos sistemos programinės įrangos, bet ir kitų jos komponentų reikalavimus.

1. Introduction	1. Įvadas
1.1 System purpose	1.1 Sistemos paskirtis
1.2 System scope	1.2 Sistemos apimtis
1.3 Definitions, acronyms and abbreviations	1.3 Terminų žodynas
1.4 References	1.4 Naudojami dokumentai
1.5 System overview	1.5 Sistemos apžvalga
2. General system description	2. Bendrasis sistemos aprašas
2.1 System context	2.1 Sistemos kontekstas
2.2 System modes and states	2.2 Sistemos darbo režimai ir jos būsenos
2.3 Major system capabilities	2.3 Svarbiausios sistemos galimybes
2.4 Major system conditions	2.4 Svarbiausios sistemos sąlygos
2.5 Major system constraints	2.5 Svarbiausi sistemos ribojimai
2.6 User characteristics	2.6 Kvalifikacinių vartotojų reikalavimai
2.7 Assumptions and dependencies	2.7 Prieišo ir priklausomybės
2.8 Operational scenarios	2.8 Sistemos naudojimo scenarijai
3. System capabilities, conditions, and constraints	3. Sistemos galimybes, sąlygos ir ribojimai
3.1 Physical	3.1 Fizinių charakteristikos
3.1.1 Construction	3.1.1 Konstrukcija
3.1.2 Durability	3.1.2 Ilgaamžiškumas
3.1.3 Adaptability	3.1.3 Adaptyuojamumas
3.1.4 Environmental conditions	3.1.4 aplinkos sąlygos
3.2 System performance characteristics	3.2 Sistemos našumo charakteristikos
3.3 System security	3.3 Sistemos apsauga
3.4 Information management	3.4 Informacijos valdymas
3.5 System operations	3.5 Sistemos veikimas
3.5.1 System human factors	3.5.1 Žmogaus veiksniai
3.5.2 System maintainability	3.5.2 Sistemos priziūrimumas
3.5.3 System reliability	3.5.3 Sistemos patikimumas
3.6 Policy and regulation	3.6 Nuostatos ir reguliavimas
3.7 System life cycle sustainment	3.7 Sistemos gyvavimo ciklo palaikymas
4. System interfaces	4. Sistemos interfeisai
Appendices	Priedai

116 pav. IEEE 1233-1998 standarto [ST9] nustatyta sistemos reikalavimų specifikacijos struktūra.

Sakoma, kad sistemos reikalavimai yra formuluojami, o sistemos programinės įrangos reikalavimai yra gaunami iš sistemos reikalavimų, po to yra lokalizuojami tą įrangą sudarančiose programų sistemose (kuriamos sistemos programiniuose komponentuose) ir nuleidžiami į tu sistemų lygmenį (žr. 5.14 poskyrį).

Populiariausias sistemos reikalavimų specifikacijos standartas yra IEEE 1233-1998 [ST9]. Trumpai apžvelgsime, kaip sistemos reikalavimų specifikacija yra traktuojama tame standarte.

Standarto rekomenduojama sistemos specifikacijos struktūra pateikta 116 pav. Kaip ir kitų specifikacijų struktūra, ji pateikta anglų ir lietuvių kalbomis (kodėl taip yra daroma žr. 6.3 poskyrį). Reikia pastebėti, kad dokumentas pritaikytas bet kurioms sistemoms turinčioms programinius komponentus specifikuoti. Taip pat ir įvairiomis techninėmis bei įterptinėmis sistemoms. Todėl, specifikuojant informacines sistemas, prisireikia ne visų specifikacijos struktūrių dalių.

Trumpai aptarsime tuos specifikacijos skyrius ir poskyrius, kurių paskirtis nėra savaime aiški.

Aprašant sistemos apimtį (1.2 poskyris), nurodomas sistemos pavadinimas, trumpai yra apibūdinami tie sistemos vartotojų operaciniai poreikiai, kuriuos privalo tenkinti sistema, aprašomi verslo tikslai, kurių siekti turi padėti sistema ir apibūdinama ta nauda, kurią kaip tikimasi turėtų duoti sistema.

Aprašant sistemos kontekstą (2.1 poskyris), pateikiama ir paaškinama sistemos konteksto diagrama (žr. 5.3 poskyri).

Nefunkciniai sistemos reikalavimai specifikacijoje yra išskaidomi į dvi dalis: sąlygas (2.4 poskyris) ir ribojimus (2.5 poskyris). Sąlygomis yra priskiriamos sistemos našumo, patikimumo ir kiti reikalavimai, kurių įgyvendinimą bent jau iš principo galima tiesiogiai pamatuoti, o ribojimams – prižiūrimumo, aptarnaujamumo, adaptuojamumo ir kiti panašūs reikalavimai, kurie juos operacionalizuojant yra performuluojami į projektavimo, programavimo, testavimo ar kokius nors kitus technologinius reikalavimus.

Aprašant sistemos naudojimo scenarijus (2.8 poskyris) pateikiama ne tik UML ar kokiomis nors kitomis priemonėmis pavaizduoti sistemos naudojimo scenarijų grafiniai modeliai, bet ir konkretūs tų scenarijų pavyzdžiai.

Informacinių sistemų atveju sistemos konstrukcijos aprašas (3.1.1 skyrelis) yra suprantamas kaip reikalavimai kokia techninė įranga gali būti naudojama sistemoje. Kitaip tariant, šis skyrelis aprašo operacinės sistemos techninius komponentus. Techninių sistemų atveju Jame aprašomas mechaninės, cheminės ir kitos aplinkos, kurioje bus instaliuojama sistema, charakteristikos.

Informacinių sistemų specifikacijoje skyreliai „Ilgaamžišumas“ (3.1.2 skyrelis) ir „Aplinkos sąlygos“ (3.1.4 skyrelis) yra nereikalingi.

Aprašant žmogiškuosius veiksnius (3.5.1 skyrelis), aprašomas žmonių vaidmuo sistemoje. Reikia prisiminti, kad žmonės yra vienas iš informacinių sistemos komponentų (žr. 2.6 poskyrį). Iš esmės jie yra traktuojami kaip lygiaverčiai procesoriai, realizuojantys dalį sistemos galimybių. Šiame skyrellyje ir yra aprašoma, kokias sistemos galimybes realizuos žmonės arba, kitaip tariant, kokios informacijos apdorojimo procedūros sistemoje bus rankinės.

Aprašant nuostatas ir teisinį reguliavimą (3.6 skyrelis), aprašoma kokiomis organizacijos politikos nuostatomis yra grindžiama kuriamoji sistema, kokių organizacių struktūrų prieiks jai aptarnauti ir kokie turėtų būti tos sistemos naudojimo nuostatai.

Aprašant sistemos gyvavimo ciklo palaikymą, aprašoma, ką turėtų daryti sistemos aptarnavimo tarnybos be tiesioginio sistemos aptarnavimo, siekdamos gerinti

sistemos veikimą. Čia gali būti numatytos mėnesinės ar metinės sistemos profilaktikos, įvairūs sistemos auditai, vartotojų nusiskundimų rinkimas ir apdorojimas bei kiti panašaus pobūdžio dalykai.

6.6 Programinės įrangos reikalavimų specifikacija

Kaip parodyta 6 lentelėje, daugiausiai reikalavimų dokumentavimo standartų skirta būtent programinės įrangos reikalavimų specifikacijoms standartizuoti. Terminą *programinės įrangos reikalavimų specifikacija* mes vartojaame todėl, kad norime apimti ne tik programų sistemų specifikacijas <*Trm45*>, bet ir jų komponentų, pavyzdžiu, jų posistemių specifikacijas. Visos šios specifikacijos turi turėti panašią struktūrą.

Kiekvienas programinės įrangos reikalavimų specifikacijos standartas nustato kiek kitokią reikalavimų specifikacijos struktūrą, tačiau visi jie reikalauja, kad specifikacijoje būtų pateikti tie patys pagrindiniai dalykai: sistemos vizija, jos interfeisų reikalavimai, funkciniai reikalavimai, našumo (angl. *performance*) reikalavimai, projektavimo ribojimai ir reikalaujančios sistemos kokybės atributų reikšmės. Kokį konkretų standartą geriau pasirinkti, priklauso nuo to, pagal koki programų sistemas gyvavimo ciklo modelį yra dirbama, ir nuo to, koks konkretus programų sistemas inžinerijos procesas yra pasirinktas tam modeliui įgyvendinti. Jeigu, tarkime, yra dirbama su kokia nors reikalavimų duomenų baze (žr. 8.5 poskyri) ir visas dokumentas arba jo dalys yra iš tos bazės automatiškai generuojami, tai ir reikalavimų specifikacijos standartas turi atitinkti generatoriaus galimybes. Be to, reikia turėti omenyje, kad visi standartai yra labai bendro pobūdžio, nusako tik pačią bendriausią dokumento struktūrą. Pasirinkus bet kurį iš tų standartų, jį vis vien reikia detalizuoti ir pritaikyti (angl. *tailor*) konkretaus projekto poreikiams.

Beje, yra ir toks požiūris, kad programų sistemas reikalavimų specifikacijos apskritai nereikia rengti. Panašiai kaip sistemos reikalavimų specifikacijos atveju, programų sistemas reikalavimų specifikaciją galima pakeisti bendruoju programų sistemas aprašu, vartotojo vadovu (angl. *users manual*) arba netgi, kaip siūlo ekstremaliojo programavimo metodika [58], tai sistemai testuoti skirtų testų specifikacijomis. Toks požiūris pateisinamas tais atvejais, kada svarbiau yra kuo greičiau pateikti sistemą rinkai arba užsakovui, negu užtikrinti ilgalaikę tos sistemos priežiūrą.

Panagrinėkime kaip siūlo struktūrizuoti programų sistemas reikalavimų specifikaciją pora svarbesnių standartų.

6.6.1 IEEE 830-1998 standarto nustatyta reikalavimų specifikacijos struktūra

Nors IEEE 830-1998 standartas [ST3] buvo sukurtas gana seniai, 1998 metais, jis vis dar tebéra vienas iš populariausių programų sistemas reikalavimų specifikaciją aprašančių standartų. Sutinkamai su šiuo standartu, programų sistemas reikalavimų specifikacija yra dokumentas, kuris:

- *Dokumentuoja užsakovo ir vykdytojo susitarimus dėl kuriamos programų sistemos funkcionalumo.* Išsamus sistemos vykdomų funkcijų aprašas turi padėti busimiems sistemos vartotojams nuspręsti, ar sistema tenkins jų poreikius ir, jeigu taip nėra, padėti jiems aprašyti, kaip ją reikėtų pakeisti.
- *Sumazina sistemos kūrimo darbų apimtis.* Rengdamos reikalavimų specifikaciją, visos sistema suinteresuotos šalys dar prieš pradedant sistemos projektavimą yra priverstos detaliai išnagrinėti ir aptarti visus tos

sistemos reikalavimus, kas sumažina vėlesnio reikalavimų keitimo riziką, o tuo pačiu ir jos perprojektavimo, perprogramavimo ir pakartotino testavimo darbų apimtis.

- *Yra pagrindas projekto kainai vertinti ir jo vykdymo terminams planuoti.* Išsamus sistemos reikalavimų aprašas padeda įvertinti jos realizavimo darbų apimtis, o tuo pačiu ir projekto kainą bei vykdymo terminus.
- *Sukuria bazinį reikalavimų komplektą, naudojamą projekto eigai sekti ir kuriama sistemai vertinti.* Gerai parengta reikalavimų specifikacija padeda planuoti projekto eigos kontrolės bei sukurtų artefaktų tikrinimo ir vertinimo darbus. Be to, būdama sudėtinge sandorio su užsakovu dalimi, specifikacija padeda matuoti projekto progresą.
- *Palengvina perkelti sistemą į naują verslo aplinką ar į kitą kompiuterinę platformą.* Reikalavimų specifikacija leidžia nustatyti, kokius reikalavimus reikia keisti, keliant sistemą į naują verslo aplinką ar į kitą kompiuterinę platformą. Dėl to yra lengviau nuspresti, ką gi konkrečiai sistemoje reikia perdaryti.
- *Palengvina sistemos tobulinimo darbus.* Keičiant ar tobulinant sistemą, pirmiausiai turi būti pakeista jos reikalavimų specifikacija. Šitaip nustatoma, kokie sistemos reikalavimai turi būti keičiami, o tuo pačiu gali būti nustatyta ir kokias konkrečias sistemos dalis palies tie pakeitimai.

Programų sistemos reikalavimų specifikacija neturi lieсти jokių sistemos projektavimo ar programavimo klausimų. Joje neturi būti rašoma, į kokius komponentus reikia skaidyti programų sistemą, koks turi būti tų komponentų funkcionalumas, kokius interfeisus jie turi turėti, kokiais duomenimis keistis ar pan. Projektavimo sprendimų nereikia aprašyti specifikacijoje net ir tais atvejais, kada tie sprendimai yra jau padaryti ir žinomi. Juos reikia aprašyti projektavimo dokumentuose. Kitaip būtų pažeistas turinių atskyrimo principas. Kitą vertus, specifikacijoje turi būti aprašytos nefunkcinės sistemos charakteristikos, ribojančios projektuotojų priimamus sprendimus. Pavyzdžiui, iš sistemos apsaugos ar jos saugumo reikalavimų gali sekti tiesioginė išvada, kad kai kurias sistemos funkcijas būtina lokalizuoti tame pačiame komponente, kad turi būti ribojamas tam tikrų komponentų komunikavimas ar kad būtina yra tikrinti tam tikrų duomenų kritines reikšmes.

Programų sistemos reikalavimų specifikacija taip pat neturi dubliuoti informacijos, pateikiamas kokybės valdymo plane. Joje neturi būti pateikiami projekto reikalavimai. Jie yra aprašomi kituose dokumentuose.

1. Introduction	1. Įvadas
Purpose	Dokumento paskirtis
Scope	Sistema, jos sprendžiami uždaviniai
Definitions, acronyms, abbreviations	Terminių žodynas
Reference documents	Naudojami dokumentai
Overview	Likusios dokumento dalies apžvalga
2. Overall Description	2. Bendras aprašas
Product perspective	Sistemos kontekstas
Product functions	Sistemos funkcionalumas
User characteristics	Vartotojams keliami reikalavimai
Constraints	Projektavimo ir realizavimo ribojimai
Assumptions and dependencies	Prielaidos ir priklausomybės
3. Specific Requirements	3. Detalūs reikalavimai
Appendices	Priedai
Index	Indeksas

117 pav. IEEE 830-1998 standarto [ST3] nustatyta programų sistemos reikalavimų specifikacijos struktūra.

Rekomenduojama dokumento struktūra pateikta 117 pav. Kaip ir kitų specifikacijų struktūra, ji pateikta anglų ir lietuvių kalbomis (kodėl taip yra daroma žr. 6.3 poskyri).

Įvadinį dokumento skyrių sudaro penki poskyriai:

- dokumento paskirtis,
- sistemos ir jos sprendžiamų uždavinių aprašas,
- terminų ir santrumpų žodynas,
- panaudotų dokumentų sąrašas,
- likusios dokumento dalies apžvalga.

Pirmajame poskyryje aprašoma kokiai skaitytojų auditorijai yra skirta specifikacija ir kokiems tikslams numatoma ją naudoti. Pavyzdžiu, specifikacija gali būti parengta kaip produkto, kurį norima įsigyti aprašas, kaip užduotis sistemą kuriančiam vykdytojui ar kt.

Antrajame poskyryje pateikiama pilna ir sutrumpinta sistemos pavadinimai, aprašoma kokiose veiklos srityse sistemą galima naudoti ir ką ten su ja galima daryti. Taip pat yra aprašoma, kokius operacinus poreikius ji tenkins, kokios naudos verslui yra iš jos tikimasi ir kokių tikslų apskritai yra siekiama ją kuriant. Poskyrio pabaigoje išvardinami tie informacinės ar kitos aukštesnio lygmens sistemos reikalavimai, kuriuos ji turi įgyvendinti.

Terminų ir santrumpų žodyne turi būti pateiktos visų specifikacijoje vartojamų terminų apibrėžtys. Vėliau jos perkeliamas į duomenų žodyną (žr. 5.10 poskyri). Jei apibrėžtys imamos iš kitų dokumentų, turi būti padarytos nuorodos į tuos dokumentus. Reikia pasakyti, kad Lietuvoje į dokumentuose pateikiama terminų žodynus vis dar žiūrima nepakankamai rimtai. Paprastai žodynas, jeigu jis apskritai yra rengiamas, pateikiamas dokumento prieduose, ji sudaro vos keli ar geriausiu atveju keliolika terminų. Vakarų šalių dokumentalistikoje vyrauja visai kita tradicija. Ten žodynas laikomas viena iš svarbiausių dokumento dalių, tame paprastai pateikiama ne mažiau kaip keliasdešimt terminų.

Kadangi mūsų terminija vis dar yra nelabai nusistovėjusi, tai žodyne greta lietuviško termino visuomet reikėtų pateikti ir jo angliską atitikmenį.

Panaudotų dokumentų sąraše pateikiama kiekvieno dokumento, į kurį kur nors specifikacijoje yra padaryta nuoroda, bibliografiniai duomenys. Jie turi atrodyti panašiai, kaip šios knygos gale pateiktame literatūros sąraše. Kiekvienam dokumentui privalu nurodyti jo pavadinimą, kas jį parengė ir jo parengimo datą. Jeigu dokumentas turi kelias versijas, būtinai reikia nurodyti, į kurią dokumento versiją yra daromos nuorodos. Programų sistemos reikalavimų specifikacijoje nuorodos dažniausiai yra daromos į aukštesnio lygmens specifikacijas, sandorio dokumentus, projekto vykdymo planus, kokybės valdymo planus, įvairius standartus ir į terminų žodynus.

Paskutiniame įvado poskyryje trumpai aprašoma, kokia medžiaga yra pateikiama likusiuose dokumento poskyriuose ir jo prieduose.

Antrajame dokumento skyriuje jokie konkretūs sistemos reikalavimai dar nėra aprašomi. Tam skirtas trečiasis dokumento skyrius. Antrajame skyriuje pateikiama medžiaga, reikalinga trečiajame skyriuje pateikiems reikalavimams suprasti. Skyrių sudaro šie poskyriai:

- sistemos kontekstas,
- sistemos funkcionalumas,
- vartotojams keliami reikalavimai,
- projektavimo ir realizavimo ribojimai,
- prielaidos ir priklausomybės.

Pirmajame poskyryje aprašomas specifikuojamos sistemos kontekstas ir ji yra palyginama su kitais žinomais analogiškos paskirties produktais. Jei programų sistema yra informacinės sistemos dalis, tai reikia aprašyti kokias kitas dalis dar turi informacinę sistemą ir per kokius interfeisus vyksta informacinės sistemos ir programų sistemos sąveika. Čia taip pat išsamiai aprašomi vartotojo ir kiti sistemos interfeisai (sąveikai su technine įranga, sąveikai su kitomis programų sistemomis, kompiuterinio ryšio interfeisai ir kt.) bei techninės įrangos ribojimai (operatyvioji atmintis, atmintis išoriniuose įrenginiuose, reikalaujamas procesoriaus greitis ir kt.). Aprašant vartotojo interfeisus yra aprašomi ne tik langai, meniu bei kiti panašūs dalykai, bet ir tų interfeisų konfigūravimo bei optimizavimo galimybės. Kiekvienam interfeisiui aprašoma jo dalykinė paskirtis (t. y. kokias užduotis galima formuliuoti) ir jo įsisavinamumo reikalavimai. Standartas pateikia tokį pavyzdį „4 kvalifikacijos operatorius po 1 valandos mokymo turi išmokti pateikti užduotį X per Z minučių“. Šiame poskyryje taip pat yra aprašomos pagalbinės sistemos funkcijos, sistemos užduočių vykdymo režimai ir būdai, bei ką galima daryti su sistema ją instalijuojant.

Antrajame poskyryje pateikiamas svarbiausių sistemos vykdomų funkcijų anotuotas sąrašas. Jeigu programų sistema yra informacinės ar kokios nors kitokios sistemos dalis, tas sąrašas gali būti paimtas iš aukštesniojo lygmens specifikacijos arba, kitaip tariant, čia gali būti surašytos tos informacinės ar kitos sistemos funkcijos, kurios buvo lokalizuotos duotojoje programų sistemoje (žr. 5.14 poskyri). Žemyn į programų sistemos lygmenį jos dar neturi būti nuleistos.

Trečiajame poskyryje aprašoma kokį išsilavinimą ir kokią darbo patirtį privalo turėti vartotojai, kad jie galėtų dirbti su specifikuojama programų sistema.

Ketvirtajame poskyryje išrašomi visi ribojimai, iš kuriuos turi atsižvelgti projektuotojai ir programuotojai, projektuodami sistemą ir rašydami programas. Šie ribojimai apima:

- teisės aktus, kurių negali pažeisti sistema,
- patikimumo reikalavimus,
- apsaugos ir saugos reikalavimus,
- nurodymus, kokiomis programavimo kalbomis galima rašyti programas,
- audito ir kontrolės procedūras, kurias privalu numatyti sistemoje,
- sistemos kritiškumo aprašą,
- kompiuterinės platformos reikalavimus,
- leidžiamus naudoti protokolus,
- lygiagrečiai vykdomas operacijas.

Be abejo, šis sąrašas nėra išsamus. Čia reikia surašyti viską, iš ką privalo atsižvelgti sistemą kuriantys projektuotojai ir programuotojai.

Paskutiniame šio skyriaus poskyryje aprašomos visos prielaidos, kurios buvo darytos rašant reikalavimus ir kurioms pakitus teks keisti ir reikalavimus. Čia taip pat aprašoma, kurie reikalavimai nuo kurių prielaidų yra priklausomi.

Trečiajame dokumento skyriuje pateikiami detalūs programų sistemos reikalavimai. Čia pateikiami:

- interfeisų reikalavimai,
- funkciniai reikalavimai,
- našumo reikalavimai,
- duomenų bazų reikalavimai,
- projektavimo ribojimai.

Nėra pateikiamos jokios informacijos, kartojančios antrajame skyriuje pateiktą informaciją. Interfeisų reikalavimai detaliai specifikuojami per tuos interfeisus tekančius

duomenų srautus. Funkciniai reikalavimai yra nuleisti į programų sistemos lygmenį, o projektavimo ribojimai detalizuoti iki pamatuojamų charakteristikų lygmens. Be to, čia pridedami prieinamumo, priežiūros ir keliamumo reikalavimai.

6.6.2 Reikalavimų inžinerijos proceso VOLERE nustatyta reikalavimų specifikacijos struktūra

„Volere“ [95] yra vienas iš šių metu populiausiu reikalavimų inžinerijos proceso modelių. Ši modelj aptarėme 3.4.3 skyrelyje. Be kita ko, „Volere“ modelis nustato ir reikalavimams specifikuoti skirtų dokumentų struktūrą. Skirtingai nuo IEEE 830-1998 standarto nustatytos struktūros, šios struktūros neturi jokio oficialaus statuso. Tai *de facto*, o ne *de jure* standartas.

Reikalavimo Nr:	Reikalavimo tipas:	Verslo transakcija:
Reikalavimas: vieno sakinio tekstas		
Pagrindimas: kodėl reikia tokio reikalavimo		
Šaltinis: kas suformulavo šį reikalavimą		
Atitikimo kriterijus: kiekybinis matas arba kokybinė skali		
Užsakovo pasitenkinimas: koks bus užsakovo pasitenkinimas (5 balų skaleje), jei reikalavimas bus sekmingai įgyvendintas	Užsakovo nepasitenkinimas: koks bus užsakovo nepasitenkinimas (5 balų skaleje), jei reikalavimas nebus įgyvendintas	Konfliktai: reikalavimai, kuriuos bus neįmanoma įgyvendinti, jei šis reikalavimas bus įgyvendintas
Prioritas: reikalavimo vertė užsakovui		
Papildoma medžiaga: nuorodos į dokumentus iliustruojančius ar aiškinančius reikalavimą		
Istorija: sukurtas pakeitimai		

118 pav. Reikalavimų inžinerijos proceso modelio „Volere“ nustatyta reikalavimo aprašo kortos struktūra.

„Volere“ modelis nustato ne tik visos programų sistemos reikalavimų specifikacijos struktūrą, bet ir struktūrą dokumento, skirto kiekvienam iš reikalavimų aprašyti (118 pav.). Aprašas daromas atskiroje kortoje. Kortos naudojamos reikalavimams rinkti. Po to iš jų informacija perkeliama į reikalavimų specifikaciją. Jei reikalavimai kaupiami kompiuterinėje bazėje (žr. 8.5 poskyrį), tai 118 pav. pateiktą reikalavimo aprašo kortos struktūrą reikia traktuoti kaip tos bazės įrašo struktūros aprašą.

Pirmoje reikalavimo eilutėje pateikiama reikalavimų identifikuojantys duomenys. Tai reikalavimo numeris, reikalavimo tipas ir nuoroda į verslo transakciją, kuriai įgyvendinti reikalingas duotasis reikalavimas. Kiekvienas reikalavimas privalo turėti unikalų numerį, tiksliau žymenį, nes tame galima vartoti ne tik skaitmenis, bet ir raides, ir netgi specialiuosius ženklus. Vietoje reikalavimo tipo duodama nuoroda į tą reikalavimų specifikacijos poskyrį, kuriame tas reikalavimas yra patektas. Nuoroda gali būti daroma į bet kurį reikalavimų specifikacijos (119 pav.) poskyrį. Tai reiškia, kad „Volere“ modelyje prie reikalavimų priskiriami ir tokie dalykai, kaip, tarkime, projekto tikslai ar rinkoje perkami kuriamosios sistemos komponentai. Pagaliau, „Volere“ modelis grindžiamas prielaida, kad reikalavimai yra analizuojami analizuojant verslo transakcijas <Trm81>. Verslo transakcijos arba tas transakcijas inicijuojantys verslo įvykiai turi būti sunumeruoti. Reikalavimų aprašančioje kortelėje nurodomas atitinkamos verslo transakcijos arba atitinkamo verslo įvykio numeris ir šitaip reikalavimas yra susiejamas su atitinkama verslo transakcija.

<p>1. Project Drivers</p> <ul style="list-style-type: none"> 1.1 The Purpose of the Project 1.2 The Client, the Customer, and Other Stakeholders 1.3 Users of the Product <p>2. Project Constraints</p> <ul style="list-style-type: none"> 2.1 Mandated Constraints 2.2 Naming Conventions and Definitions 2.3 Relevant Facts and Assumptions <p>3. Functional requirements</p> <ul style="list-style-type: none"> 3.1 The Scope of the Work 3.2 The Scope of the Product 3.3 Functional and Data Requirements <p>4. Nonfunctional Requirements</p> <ul style="list-style-type: none"> 4.1 Look and Feel Requirements 4.2 Usability and Humanity Requirements 4.3 Performance Requirements 4.4 Operational and Environmental Requirements 4.5 Maintainability and Support Requirements 4.6 Security Requirements 4.7 Cultural and Political Requirements 4.8 Legal Requirements <p>5. Project Issues</p> <ul style="list-style-type: none"> 5.1 Open Issues 5.2 Off-the-Shelf Solutions 5.3 New Problems 5.4 Tasks 5.5 Migration to the New Product 5.6 Risks 5.7 Costs 5.8 User Documentation and Training 5.9 Waiting Room 5.10 Ideas for Solutions 	<p>1. Projekto varančiosios jėgos</p> <ul style="list-style-type: none"> 1.1 Projekto paskirtis 1.2 Klientas, ižsakovas ir kitos suinteresuotosios šalys 1.3 Produktio vartotojai <p>2. Projekto ribojimai</p> <ul style="list-style-type: none"> 2.1 Privalomi rebojimai 2.2 Susitarimai dėl vardo ir apibrėžtys 2.3 Priešaidos ir kiti reikšmingi faktai <p>3. Funkciniai reikalavimai</p> <ul style="list-style-type: none"> 3.1 Projekto apimtis 3.2 Produktio spėnis 3.3 Funkciniai ir duomenų reikalavimai <p>4. Nefunkciniai reikalavimai</p> <ul style="list-style-type: none"> 4.1 Vartotojų interfeisų reikalavimai 4.2 Panaudojamumo ir ergonominių reikalavimai 4.3 Našumo reikalavimai 4.4 Operacinių ir aplinkos reikalavimai 4.5 Aptarnavimo ir priežiūros reikalavimai 4.6 Apsaugos reikalavimai 4.7 Kultūriniai ir politiniai reikalavimai 4.8 Teisiniai reikalavimai <p>5. Kiti su projektu susiję klausimai</p> <ul style="list-style-type: none"> 5.1 Neatsakyti klausimai 5.2 Rinkoje perkami komponentai 5.3 Problemos, su kuriomis bus susidurta įdiegus projekta 5.4 Išduodys 5.5 Produktais diegimas 5.6 Rizikos veiksniai 5.7 Kaštai 5.8 Vartotojams skirti dokumentai ir vartotojų mokymas 5.9 Atidėti reikalavimai 5.10 Idėjos kaip igyvendinti reikalavimus
---	--

119 pav. Reikalavimų inžinerijos proceso modelio „Volere“ nustatyta reikalavimų specifikacijos struktūra.

Visų reikalavimo aprašymo kortos laukų neaptarinėsime. Daugumos jų paskirtis pakankamai aiškiai aprašyta pačiame 118 pav. Kam reikalingi laukai „Užsakovo pasitenkinimas“, „Užsakovo nepasitenkinimas“ ir „Prioritetas“ jau buvo aiškinta 5.12 ir 5.13 poskyriuose. Todėl čia pakalbėsime tik apie lauką „Atitikimo kriterijus“ (5 kortos eilutė) ir „Istorija“ (9 kortos eilutė) paskirtį.

„Atitikimo kriterijus“ – tai pamatuojamas tikslas, kurį reikia pasiekti, kad sistema galėtų praeiti baigiamuosius bandymus. Kitaip tariant, tai yra matuojama sistemos savybė. Taigi, „Volere“ modelyje taip pat, kaip ir kokybės funkcijų skaidos metodikoje (žr. 5.13 poskyri), yra reikalaujama, kad reikalavimai būtų siejami su išmatuojamomis sistemos savybėmis. Tačiau čia, skirtingai negu kokybės funkcijų skaidos metodikoje, funkcinių reikalavimų igyvendinimo laipsnių siūloma matuoti ne kokybiniu matu Taip/Ne, bet tos funkcijos igyvendinimui patikrinti skirtu testų rinkiniu.

Laukas „Istorija“ skirtas reikalavimo statuso pokyčiams registruoti (žr. 8.6 poskyri). Čia rašoma, kada reikalavimas buvo suformuluotas, keistas, analizuotas, vertintas, įtrauktas į specifikaciją, igyvendintas ir kt. Paskaičius šitą lauką, iš karto sužinoma, kas su reikalavimu jau buvo padaryta, koks yra jo dabartinis statusas ir kas su juo dabar yra daroma.

„Volere“ reikalavimų specifikacijos struktūra parodyta 119 pav. Kaip ir IEEE 830-1998 standarto atveju, ji pateikta dviem kalbom: anglų ir lietuvių. Įvadinės dalies dokumentas neturi, jis pradedamas poskyriu „Projekto paskirtis“. Šiame poskyryje visų pirma aprašoma, kokias verslo problemas padės spręsti ar kokiomis naujomis verslo galimybėmis padės pasinaudoti kuriamoji programų sistema. Bendruoju atveju tai turi būti imama iš verslo lygmens reikalavimų ir čia pateikiama labai trumpai, skiriant problemai ar galimybei aprašyti tik 1-2 sakinius. Kadangi knygoje [95] kaip iliustracinis pavyzdys pateikiama kelių priežiūros tarnybos informacinių sistemos, tai jos problema yra įvardinama tokiu sakiniu:

„Kelių priežiūros tarnyba yra nelaiminga dėl didelio avarių, vykstančių dėl aplėdėjusių kelių, skaičiaus.“ [95].

Toliau yra aprašomi projekto tikslai. Nagrinėjamo pavyzdžio atveju, jie skamba šitaip:

„Sumažinti keliuose vykstančių avarijų skaičių, tiksliau prognozuojant kelių aplėdėjimą ir planuojant jų valymą.“ [95].

Aprašant projekto tikslus, turi būti aprašyta ir tai, kokią naudą verslas gaus juos pasiekus. Rekomenduojama nurodyti vieną iš trijų atvejų: įgys pranašumus rinkoje, sumažins veiklos kaštus, padidins savo klientams teikiamų produktų ar paslaugų vertę. Nagrinėjamo pavyzdžio atveju, kelių priežiūros tarnyba būtent ir pradės teikti kokybėkesnes paslaugas savo klientams. Tikslai turi būti kiekybiniai. Kitaip nebūtų galima įrodyti, kad sistema iš tiesų duos verslui kokią nors apčiuopiamą naudą. Nagrinėjamo pavyzdžio atveju, galima siekti sumažinti avarijų skaičių, tarkime, per pusę. Kitą vertus, jie turi būti dekomponuojami į sistemos funkcinius reikalavimus ir todėl patys yra traktuojami kaip aukščiausiojo lygmens funkciniai reikalavimai.

Trumpai tariant, poskyryje „Projekto paskirtis“ pateikiamą trumpą verslo lygmens reikalavimą (žr. 2.4 poskyri) santrauka. Mažesniuose projektuose gali būti ir taip, kad verslo lygmens specifikacija kaip atskiras dokumentas apskritai nėra rengiama ir tada ji visa pateikiamā šiame poskyryje. „Volere“ modelyje rekomenduojama, kad kiekvienai sprendžiamai problemai būtų pateikta nuoroda į literatūros šaltinį, kuriame aprašytas principinis tos problemos sprendimo būdas. Šitaip pagrindžiamas vienas iš sistemos techninio įgyvendinamumo aspektų.

Poskyryje „Klientai, užsakovai ir kitos suinteresuotosios šalys“ aprašomos visos sistemos kūrimu suinteresuotos šalys ir kiekvienai iš jų nurodoma, kokius tos šalies operacinis poreikius turi tenkinti sistema. Kitaip tariant, šiame poskyryje yra pateikiamā arba vartotojo lygmens reikalavimų specifikacijos santrauka, arba pati ta specifikacija.

Poskyryje „Produkto vartotojai“ išsamiai aprašomi visi tie, kas tiesiogiai dirbs su sistema. Šis poskyris traktuojamas kaip aukščiausio lygmens sistemos panaudojamumo reikalavimai. Kitaip tariant, čia turi būti pateikta visa informacija, į kurią būtina atsižvelgti projektuojant ir konstruojant vartotojo interfeisus. Nagrinėjamo pavyzdžio atveju, ši informacija gali būti tokia:

„Pagrindiniai sistemos vartotojai yra kelių valymo tarnybos inžinieriai. Jie gerai išmano viską apie kelių tipus, apie jų išdėstytmą ir apskritai viską apie kelių tinklą. Jie turi darbo su asmeniniais kompiuteriais patirties, naudojasi tekstu redagavimo sistemomis ir kompiuterinio projektavimo paketais. Visi jie turi bakalauro arba magistro laipsnį ir visi kalba angliskai.“ [95].

Antrasis dokumento skyrius skirtas projekto ribojimams aprašyti. „Volere“ modelyje jie irgi yra traktuojami kaip aukšto lygmens reikalavimai. Pirmajame šio skyriaus poskyryje pateikiami vadinamieji *privalomieji reikalavimai*. Jie paliečia visą sistemą, į juos būtina atsižvelgti kuriant bet kurį tos sistemos komponentą. Jie skirstomi į:

- įgyvendinimo būdo ribojimus (angl. *solution constraints*),
- esamos sistemos veikimo aplinkos ribojimus (angl. *implementation environment of the current system*),
- sąveikos su aplinka ribojimus (angl. *partner or collaborative applications*),
- rinkoje įsigyjamos programinės įrangos ribojimus (angl. *off-the-shelf software*),
- darbo vietų ribojimus (angl. *anticipated workplace environment*),

- vykdymo terminų ribojimus (angl. *schedule constraints*),
- biudžeto ribojimus (angl *budget constraints*).

Igyvendinimo būdo ribojimai iš esmės yra projektavimo ar technologiniai sprendimai, kuriuos dėl vienų ar kitų priežasčių priima užsakovas ir primeta vykdytojams. Jie gali būti susiję su kuriamos sistemos architektūra, projektavimo ar programavimo technologija ir kitais panašaus pobūdžio dalykais. Pavyzdžiui, gali būti suformuluotas tokis ribojimas:

„Visi vartotojo interfeisai sistemoje turi atitikti Microsoft Windows grafinio vartotojo interfeiso standartus“.

Esamos sistemos veikimo aplinkos ribojimai irgi yra panašaus pobūdžio. Jie formuluojami tuomet, kuomet užsakovas nori, kad sukurtoji sistema veiktų toje pačioje aplinkoje, kurioje jau veikia kitos jo naudojamos sistemos. Pavyzdžiui, gali būti suformuluotas tokis ribojimas:

„Sistema turi veikti esamame asmeninių kompiuterių tinkle ir nereikalauti instaliuoti jokios papildomos sisteminės programinės įrango“.

Sąveikos su aplinka ribojimai nurodo su kokiais įrenginiais bei kokiomis kitomis programų sistemomis turi dirbti kuriamoji sistema. Pavyzdžiui, gali būti suformuluotas tokis ribojimas:

„Duomenis apie klientus sistema turi imti iš gyventojų registro“.

Rinkoje įsigyjamos programinės įrango ribojimai aprašo, kokius gatavus produktus reikia įsigyti rinkoje ir panaudoti kaip kuriamos sistemos komponentus. Darbo vietų ribojimai formuluojami tiktais tuomet, kuomet jie daro įtaką kokiems nors kitiems reikalavimams. Šie ribojimai buvo aptarti 2.4.6, 2.5.6, 2.6.6 ir 2.7.6 skyreliuose.

Antrajame antro skyriaus poskyryje yra pateikiamas duomenų žodynas (žr. 5.10 poskyrį) arba, jeigu tai įmanoma, dalykinės srities ontologija. Paskutiniame antro skyriaus poskyryje pateikiami faktai, kuriais reikia remtis kuriant sistemą, ir prielaidos, kuriomis yra grindžiami reikalavimai. Kuriant jau nagrinėtą kelių valymo informacinię sistemą reikia remtis, pavyzdžiui, tokiais faktais:

„Druską (NaCl) ledui ant kelio nutirpinti galima naudoti iki 6°C, kalcio chloridą (CaCl) – iki 15°C.

Vienos tonos ledo tirpinimo medžiagos pakanka ledui nuo trijų mylių vienos krypties kelio juostos nutirpinti.“ [95].

Galima pateikti tokius prielaidų pavyzdžius:

„Nuvalyto kelio pakartotinai valyti nereikia mažiausiai dvi valandas.

Kelių priežiūros tarnyba valo tik savo apskrities kelius.“ [95].

Trečiąjame dokumento skyriuje pateikiami funkciniai sistemos reikalavimai. Pirmajame šio skyriaus poskyryje aprašoma projekto apimtis. Ji aprašoma išvardinant verslo įvykius, kurių inicijuojamas transakcijas turi palaikyti kuriamoji sistema. Kitaip tariant, yra aprašomos kuriamos programų sistemos palaikomos verslo sistemos užduotys <Trm81>. Kiekvienai verslo užduočiai aprašomi jeities ir išeities duomenys ir 1-2 sakiniai apibūdinama atliekamos verslo transakcijos esmė. Antrajame poskyryje aprašoma kuriamos programų sistemos apimtis. Ji aprašoma panašiai kaip ir projekto apimtis, tiktais aprašinėjamos ne verslo, o programų sistemos vykdomos užduotys.

Paskutiniame šio skyriaus poskyryje ir visuose ketvirtojo skyriaus poskyriuose pateikiamos atitinkamų reikalavimų aprašų kortos (118 pav.). Šie reikalavimai jau buvo ne kartą aptarti ir plačiau prie jų neapsistosime.

Ketvirtajame dokumento skyriuje pateikiami projekto reikalavimai ir aprašomi kiti su projektu susiję klausimai. Priminsime, kad kai kurie projekto reikalavimai (biudžetas, vykdymo terminai) traktuojami kaip ribojimai ir aprašinėjami antrajame dokumento skyriuje. Apskritai, ketvirtasis skyrius yra neprivalomas. Su projektu susiję klausimai gali būti aprašomi ir kokiuose nors kituose dokumentuose. Jeigu šis skyrius yra rengiamas, tai pirmajame jo poskyryje aprašomi klausimai, kurių nepavyko galutinai išsiaiškinti reikalavimų inžinerijos stadijos metu ir kuriuos toliau reikia aiškintis projektuojant sistemą. Jau nagrinėtame kelių valymo informacinių sistemos pavyzdyme galėjo likti nebaigtas aiškintis toks klausimas:

„Kol kas liko neaišku, ar vairuotojų darbo pamainos trukmė artimiausioje ateityje nebus trumpinama.“

Antrame poskyryje atliekama preliminarinė rinkoje perkamų produktų analizė ir pateikiami duomenys apie tą produktą kainą, funkcionalumą ir įsigijimo galimybes. Trečiajame poskyryje aprašomi operacinio sistemos įgyvendinamumo slenksčiai arba, kitaip tariant, kokias naujas darbo organizavimo ir operacinės veiklos problemas teks spręsti įdiegus sistemą. Ketvirtajame poskyryje trumpai aprašomos visos projekto užduotys <Trm82>, t. y. aprašoma, ką reikia suprojektuoti, suprogramuoti, kokius dokumentus reikia parengti, kokius testus sukurti ir pan. Penktajame poskyryje pateikiamas planas, kaip bus demontuojama šiuo metu veikianti verslo sistema ir vietoje jos diegiant nauja sistema, naudojanti kuriamąją programą sistemą. Šeštajame poskyryje pateikiama projekto rizikos veiksnių analizė, septintajame – projekto kaštų vertinimas. Jei antrajame skyriuje buvo pateiktas ribojimas, kiek užsakovas gali investuoti į projektą, tai čia yra pateikiami skaičiavimai, kiek vykdytojui kainuos to projekto vykdymas. Septintajame poskyryje aprašoma, kokius sistemos vartotojams skirtus dokumentus, išskaitant pagalbos failus, reikia parengti ir kokius vartotojams skirtus mokymus reikia surengti. Aštuntajame poskyryje nurodoma, kurie sistemos reikalavimai bus įgyvendinti ne pradinėje, o kitose jos versijose. Pagaliau, paskutiniame dokumento poskyryje yra aprašomos preliminarios sistemos reikalavimų įgyvendinimo idėjos.

Kaip matome, IEEE 830-1998 standarto ir „Volere“ modelio rekomenduojamos reikalavimų specifikacijos struktūros yra labai skirtinges. Skiriiasi netgi pačios reikalavimų specifikacijos ir reikalavimo sampratos. „Volere“ reikalavimų inžinerijos proceso modelyje jaučiama stipri kokybės funkcijų sklaidos metodikos (žr. 5.13 poskyri) įtaka. Ji atispindi ir tame, ką ir kaip rekomenduojama pateikti reikalavimų specifikacijoje. Kadangi skirtingu darbo su reikalavimais metodiką yra labai daug, tai ir skirtingu reikalavimų specifikacijos standartu taip pat yra labai daug. Kurį iš jų geriausiai pasirinkti, labiausiai priklauso nuo pasirinkto reikalavimų inžinerijos proceso modelio.

7 Reikalavimų tikrinimas ir vertinimas

7.1 Reikalavimų tikrinimo ir vertinimo problemos

Reikalavimus reikia formuluoti labai kruopščiai ir po to įsitikinti, kad visos suinteresuotosios šalys juos aprobuoja. Juos būtinai dokumentuoti. Dokumentus, kuriuose aprašyti reikalavimai, reikia tikrinti (angl. verification) ir vertinti (angl. validation). Įterpiant į reikalavimų inžinerijos procesą reikalavimų tikrinimo ir vertinimo procedūras, į šį procesą yra įvedami kokybės kontrolės elementai. Reikia pabrėžti, kad reikalavimų analizė ir reikalavimų tikrinimo bei vertinimo procesai, yra skirtini procesai, nors jie ir yra daug kuo panašūs. Reikalavimų analizė atliekama prieš pradedant rengti reikalavimų specifikaciją. Analizuojami visi dalykinės srities analizės metu surinkti reikalavimai. Reikalavimų tikrinimas ir vertinimas atliekami po to, kai reikalavimų specifikacija jau yra parengta. Tikrinami ir vertinami tik tie reikalavimai, kurie buvo įtraukti į specifikaciją. Be to, reikia suprasti, kad reikalavimų tikrinimas ir jų vertinimas taip pat yra skirtini dalykai.

Tikrinant reikalavimus, siekiama atsakyti į klausimą „Ar sistema yra kuriamā teisingai?“. Dažnai reikalavimų tikrinimo metodai, o kartais ir tikrinimo lygmenys yra aprašomi pačioje reikalavimų specifikacijoje. Bendruoju atveju, tikrinimo metodai apima demonstravimą, parodantį, kad specifikacijoje suformuluoti funkcioniniai reikalavimai sistemoje tikrai yra įgyvendinti, testavimo metu surinktų duomenų analizę, realią sistemos veikimo aplinką imituojančius testavimo stendus ir pradinio programų kodo bei sistemos dokumentacijos inspektavimą. Tikrinimo lygmenys priklauso nuo pasirinktojo programų sistemos kūrimo proceso ir pasirinktojo gyvavimo ciklo modelio, nes jie siejami su gyvavimo ciklo stadijomis, kuriose reikia atlikti tikrinimo procedūras. Pavyzdžiu, atitinkamos procedūros gali būti įterptos į modulių testavimo, visos sistemos testavimo ir jos instalavimo procesus. Tačiau, kalbant apie reikalavimų inžinerijos procesą, reikalavimų tikrinimas yra suprantamas siauriau. Yra tikrinama tik tai, ar visi surinkti reikalavimai tikrai pakliuva į atitinkamą reikalavimų specifikaciją.

Vertinant reikalavimus, yra siekiama atsakyti į klausimą „Ar planuojama kurti tikrai tokią sistemą, kokios iš tiesų reikia užsakovams?“. Pagrindinis reikalavimų vertinimo uždavinys yra nustatyti, kokių reikalavimų trūksta ir kokie reikalavimai yra nepagrūsti. Pagal veiklos pobūdį reikalavimų vertinimą galima traktuoti kaip dalį sistemos tinkamumo testavimo proceso (angl. acceptance testing). Tačiau, jei reikalavimų neišsamumas ar nepagrūstumas nustatomi tik sukūrus sistemą ir pradėjus jos testavimą, tai sistemą tenka perprojektuoti ir bent jau iš dalies programuoti iš naujo. Tai kainuoja kelis kartus brangiau, nei tuos trūkumus nustačius prieš pradedant sistemos projektavimą. Todėl reikalavimų vertinimo procedūros yra terpiamos ne į sistemos tinkamumo, bet į reikalavimų inžinerijos procesą ir naudojamos ne jau sukurtai sistemai, bet tos sistemos reikalavimų specifikacijai įvertinti. Pats vertinimas taip pat yra atliekamas ne testavimo, bet kitais metodais.

Adekvaciai įvertinti reikalavimus yra neįmanoma, jei juos vertinant nedalyvauja užsakovo atstovai ir dalykinės srities ekspertai. Tai dar kartą parodo, kad formalūs reikalavimų specifikavimo metodai yra mažai tinkami realioms sistemoms kurti, nes nei užsakovo atstovai, nei būsimieji sistemos vartotojai, nei dalykinės srities ekspertai yra nepajėgūs suprasti formalias reikalavimų specifikacijas ir juo labiau jas vertinti. Tam, kad reikalavimų specifikacija būtų suprantama užsakovams, vartotojams ir ekspertams, ji turi būti parašyta jiems suprantama kalba ir iliustruota lengvai suprantamais paveikslėliais. Jei iš užsakovo reikalaujama, kad jis perprastų

kokią nors specifikavimo kalbą, tarkime, UML, arba kokią nors reikalavimų vertinimo būdą, sumažėja vertinimo procedūrų patikimumas, nes padidėja tikimybė, kad užsakovo atstovai ką nors ne taip suprato. Visas užsakovų, vartotojų ir ekspertų dėmesys turi būti sutelktas į vertinamų reikalavimų esmę, o ne į vienus ar kitus techninius vertinimo procedūrų aspektus.

Technologo požiūriu, reikalavimų tikrinimo ir vertinimo procesas yra svarbus kaip kuriamam produktui pritaikyto specializuoto gyvavimo ciklo modelio projektavimo uždavinys. Ši ciklą technologas turi taip suprojektuoti, kad tikrinimo ir vertinimo procedūroms, vykdomoms reikalavimų aiškinimosi ir formulavimo metu, būtų skirtas išskirtinis dėmesys. Be abejo, tos procedūros turi būti įterptos taip pat ir į kitus procesus, visų pirma į sistemos projektavimo bei programavimo procesus. Jos turi būti ir sistemos tinkamumo testavimo ir tos sistemos per davimo užsakovams procesuose. Kitaip suprojektuotas sistemos kūrimo procesas negali būti pripažintas tinkamu.

Taigi, baigus analizuoti dalykinę sritį, suformulavus kuriamos sistemos reikalavimus ir juos dokumentavus, sisteminiam analitikui tenka atlkti paskutinį ir labiausiai kritišką darbą – įvertinti tuos reikalavimus. Reikalavimų vertinimas yra atliekamas siekiant įsitikinti, kad:

- visos suinteresuotosios šalys pritaria suformuluotiemis reikalavimams ir juos aprobuoja;
- tie reikalavimai tikrai yra grindžiami analizuojant dalykinę sritį surinkta informacija;
- kuriamoji sistema tikrai spręs realias verslo problemas ir tenkins realius dalykinės srities specialistų poreikius.

7.2 Reikalavimų tikrinimo ir vertinimo metodai

Yra gana daug skirtingų programų sistemos reikalavimų vertinimo metodų. Svarbiausieji iš jų yra šie:

- reikalavimų darnos analizė,
- reikalavimų peržiūra,
- reikalavimų inspektavimas,
- reikalavimų maketavimas,
- juodraštinės vartotojui skirtos dokumentacijos rengimas,
- reikalavimų reformalizavimas,
- testų reikalavimams tikrinti specifikavimas.

Reikalavimų peržiūros, inspektavimo ir maketavimo metodai išsamiai nagrinėjami bakalauro studijų pakopos studentams skirtame programų sistemos inžinerijos kurse. Apie juos galima paskaityti [8, 20, 29, 70, 96, 106, 107, 117] ir daugelyje kitų vadovelių. Todėl čia apie juos nebekalbėsime. Priminsime tik tai, kad reikalavimų peržiūra yra naudojama patikrinti, ar reikalavimų specifikacija tenkina projekto standartus, t. y. ar joje nėra praleista kokių nors skyrių ar poskyrių, ar visos jos dalys yra išsamios ir suprantamai parašytos, ar dokumente apibrėžti visi vartojami terminai, ar išsamūs naudojamų dokumentų sąraše pateiktų dokumentų bibliografiniai duomenys ir pan. Trumpai tariant, yra tikrinama, ar dokumentas tikrai yra taip parašytas, kaip jis turėtų būti parašytas pagal projekto standartais numatytyus reikalavimus. Reikalavimų inspektavimas yra skirtas įvertinti reikalavimų išsamumą ir rasti specifikacijoje esančius prieštaringus reikalavimus ar nerealius reikalavimus. Reikalavimams vertinti panaudojant maketus, paprastai yra naudojami tie patys maketai, kurie buvo naudoti reikalavimams aiškintis. Vertinant reikalavimus yra

naudojami tik vykdomi maketai, be to paprastai jie yra praplečiami, realizuojant juose ne tik neaiškius reikalavimus, kaip buvo daroma kuriant reikalavimams aiškintis naudojamą maketą, bet visus svarbiausius reikalavimus. Be to, aiškinantis reikalavimus su makedu dalykinės srities specialistai dirba kartu su analitikais, o vertinant reikalavimus dalykinės srities specialistai su makedu dirba patys vieni ir jį vertina. Todėl šiuo atveju makedo patikimumas turi būti daug didesnis.

Toliau trumpai aptarsime kitus reikalavimų tikrinimo ir vertinimo metodus.

7.2.1 Reikalavimų darnos analizė

Reikalavimų darnos analizė atliekama tikslu įsitikinti, kad suformuluotieji programų sistemos reikalavimai yra išsamūs ir tarpusavyje suderinti. Darnos analizė apima:

- reikalavimų lygmenų darnos analizę,
- duomenų darnos analizę,
- užduočių darnos analizę,
- verslo procesų darnos analizę.

Analizuojant skirtinį lygmenį reikalavimų darną yra tikrinama, ar skirtinį lygmenį – verslo, vartotojo, informacinės sistemos ir programų sistemos – reikalavimai tikrai yra tarpusavyje suderinti. Tam naudojami būdai aptarti 5.7 ir 5.13 poskyriuose.

Analizuojant duomenų darną yra tikrinama, ar suformuluotuose reikalavimuose yra:

- apibrėžti visi programų sistemos darbui reikalingi duomenys, ar jie yra apibrėžti pakankamai išsamiai bei korektiškai ir ar jie turi tinkamus, t. y. suprantamus dalykinės srities specialistams, pavadinimus;
- išsamiai apibrėžtos visos reikalingos duomenų transformacijos ir ar tos apibrėžtys yra pakankamai išsamios ir korektiškos;
- yra aprašyti visos duomenų pateikties formos ir ar jos aprašyti pakankamai išsamiai ir korektiškai;
- apibrėžti visi duomenys, kuriuos turi gauti kuriamoji sistema (t. y. visi sistemos generuojami rezultatai), ir ar yra pakankamai išsamiai ir korektiškai aprašyti tų duomenų skaičiavimo būdai;
- pakankamai išsamiai aprašyti visos ataskaitos, kurias privalo generuoti kuriamoji programų sistema, ir kiti gautų rezultatų pateikties vartotojams būdai.

Analizuojant užduočių darną yra tikrinama, ar suformuluotose reikalavimuose yra:

- aprašyti visos užduotys, kurias turi vykdyti kuriamoji programų sistema (visi sistemos sprendžiami uždaviniai);
- aprašyti visos tų užduočių tarpusavio sąsajos ir ar užduotys yra teisingai susietos viena su kita;
- aprašyti visų kuriamos programų sistemos vykdomų užduočių laiko ribojimai ir ar tie ribojimai tenkina kompiuterizuodamos verslo sistemos poreikius;
- aprašyti visi kuriamos programų sistemos vykdomoms užduotims vykdyti reikalingi duomenys bei kiti ištakliai;
- aprašyta kokie vartotojai (asmens, padaliniai, procesai) kokias kuriamos programų sistemos vykdomas užduotis turi teisę vykdyti;

- visos užduotys aprašyti užsakovui, išskaitant vadovybę ir asmenis, kurie jas vykdys, suprantama kalba.

Taip pat reikia įsitikinti, kad užsakovas tikrai teisingai suprato užduočių aprašus, išskaitant ir jų formulavimo bei pateikties kompiuteriui būdus.

Analizuojant verslo procesų darną yra tikrinama, ar suformuluotose reikalavimuose yra:

- aprašyti visi kompiuterizuojami verslo procesai ir yra nurodyta, kokios kuriamos programų sistemos užduotys su kokiui verslo procesu yra susietos;
- aprašyta, kokius duomenis vienas procesas turi gauti iš kito, ir ar tokie duomenų mainai tikrai išplaukia iš verslo sistemos logikos.

Reikalavimų darnos analizė turi būti derinama su kitais reikalavimų vertinimo metodais. Šiuo metodu visų pirma yra tikrinamas reikalavimų išsamumas, t. y. žiūrima, ar kas nors nėra praleista.

7.2.2 Juodraštinės vartotojui skirtos dokumentacijos rengimas

Bandymas pagal turimą reikalavimų specifikaciją parašyti juodraštinį vartotojo vadovo variantą padeda rasti daugelį netikslių ir praleistų reikalavimų. Taip yra dėl to, kad rašant vartotojo vadovą tenka detaliai išnagrinėti kiekvieną reikalavimą ir jį perrašyti visai kitais terminais. Tai taip pat padeda geriau suvokti kaipgi vartotojui tekė dirbtį su sistema ir patobulinti sistemos panaudojamumo reikalavimus. Toks dokumentas taip pat yra naudingas ir todėl, kad jį galima duoti maketą vertinantems dalykinės srities specialistams. Viena vertus, tai palengvins jų darbą, kita vertus, jie iš karto pastebės, ko jiems trūksta šiame dokumente. Vartotojo vadove turi būti ne tik aprašyti užduočių formulavimo kalba ir darbo su sistema procedūros, bet ir išaiškinta sprendžiamų uždavinių esmė bei aprašyti visos sistemos galimybės. Jei kokią nors sistemos galimybę ar funkciją yra sunku išaiškinti vartotojams, tai reiškia, kad atitinkami reikalavimai yra prastai suformuluoti. Tačiau šitaip galima vertinti tik dalį sistemos reikalavimų, nes daugelis reikalavimų sistemoje virsta vartotojui nematomais dalykais ir vartotojo vadove tie dalykai nėra aprašinėjami.

Toks reikalavimų vertinimo metodas yra brangus. Vartotojo vadovui parašyti reikia daug darbo ir laiko. Galima galvoti, kad visa tai vėliau atsipirks, nes bus jau parengta galutinio vartotojo vadovo pradinė versija. Tačiau taip yra ne visuomet, nes projekto eigoje reikalavimai gali būti tiek pakeisti, kad iš pradinio vartotojo vadovo mažai kas beliks.

7.2.3 Reikalavimų reformalizavimas ir modelių vertinimas

Paprastai, analizuojant reikalavimus jie yra modeliuojami. Tai daroma UML ar kokia nors kita kalba. Po to neretai įvyksta taip, kad reikalavimų specifikacija ir reikalavimų modeliai pradeda gyventi atskirus gyvenimus, vienas nuo kito atitrūksta. Kadangi projektuotojai ir programuotojai iš esmės dirba tik su modeliais, tai neretai įvyksta taip, kad kokie nors pakeitimai daromi tiktais modeliuose, pamirštant juos įnešti į specifikaciją. Tačiau vertinant reikalavimus dalykinės srities specialistai skaito tik reikalavimų specifikaciją, nes modelių jie paprasčiausiai nesuprantą. Panaši problema kyla ir tuomet, kuomet reikalavimų specifikacija yra parašyta kokia nors formaliaja specifikavimo kalba. Todėl yra tikslinga modelius bei formaliomis kalbomis parašytas specifikacijas „išversti“ atgal į natūraliąjį kalbą. Panašiai, kaip ir rašant vartotojo vadovą, čia randami įvairūs reikalavimų netikslumai, neišsamumas bei kiti reikalavimų trūkumai.

Kaip ir vartotojo vadovo rašymas, šis reikalavimų vertinimo metodas yra gana brangus ir dėl to gana retai naudojamas.

Vertinant reikalavimų specifikaciją, be abejo reikia vertinti ir ją papildančių modelių kokybę. Pavyzdžiu, dirbant su objektiniais modeliais reikia įsitikinti, kad modelis numato komunikavimo maršrutus tarp visų objektų, kurie dalykinėje srityje keičiasi duomenimis. Tą galima patikrinti statinės analizės metodais. Jei reikalavimai specifikuoti kokia nors formalia specifikavimo kalba, daugelį specifikacijos savybių galima patikrinti ar net įrodyti formaliais metodais.

7.2.4 Reikalavimams tikrinti skirtų testų specifikavimas

Viena iš svarbiausių reikalavimų savybių yra reikalavimas, jog reikalavimai būtų taip suformuluoti, kad būtų galima patikrinti jų įgyvendinimo sistemoje laipsnį. Reikalavimai, kurių įgyvendinamumo patikrinti negalima, iš tiesų yra tik pageidavimai. Galimybę patikrinti reikalavimų įgyvendinamumą galima vertinti pabandant specifikuoti testus, kurie bus naudojami reikalavimų įgyvendinimui tikrinti atliekant sistemos tinkamumo testavimą. Šis metodas taip pat padeda rasti reikalavimų neišsamumo ir jų dviprasmybes. Jei negalima sugalvoti, kaip reikės tikrinti reikalavimo įgyvendinimą, tai tas reikalavimas yra arba apskritai netinkamas, arba netinkamai suformuluotas. Metodo privalumas dar yra ir tas, kad šitaip iš esmės jau pradedami sistemas testavimo darbai. Tiesa, kaip ir rašant vartotojo vadovo juodraštį, gali įvykti taip, kad projekto eigoje reikalavimai bus tiek pakeisti, kad tinkami liks tik keli iš specifikuotų testų. Be to, šitaip specifikuoti testai tinka tik atskiriems reikalavimams, o ne jų visumai tikrinti. Sistemai testuoti reikia testų, tikrinančių iš karto daugelį reikalavimų ir jų tarpusavio sąveikas. Realūs testai, skirtingai nuo reikalavimams vertinti specifikuojamų testų, turi atsižvelgti į testavimo kaštus ir į daugelį kitų veiksnių. Taigi nereikia stengtis sukurti realias testų specifikacijas, pagal kurias bus kuriami testai sistemai testuoti.

Specifikuojant reikalavimų vertinimo testus rekomenduojama atsakyti į tokius klausimus [107]:

- kokiam darbo su sistema scenarijuje galėtų būti svarbus toks testas (tai padeda įvertinti reikalavimo svarbą vartotojui);
- ar reikalavime pakanka informacijos testui specifikuoti ir konstruoti (jei tam prireikia kitų reikalavimų, tai jie tarpusavyje yra priklausomi ir turi būti sujungti trasomis (žr. 8.7 poskyri));
- ar reikalavimo įgyvendinimui patikrinti pakanka vieno testo ar jų reikia daugiau (jei reikia daugiau kaip vieno testo, tai į reikalavimą tikriausiai yra sujungti keli reikalavimai ir jų reikia išskaidyti);
- ar galima reikalavimą taip formuliuoti, kad reikalavimo įgyvendinimui patikrinti skirtas testas būtų akivaizdus.

Šio metodo kaina yra vidutinė ir priklauso nuo to, kokioje apimtyje jis yra taikomas. Kai kuriuos testus galima specifikuoti per kelias minutes, kitiems gali prireikti kelių valandų. Tačiau, šitaip tikrinant visus reikalavimus, tai trunka ilgiau negu reikalavimų inspektavimas. Tam tikrų problemų gali kilti dėl darbo pasidalinimo. Analitikai gali manyti, kad specifikuoti tokius testus yra testuotojų darbas, o pastarieji taip pat gali nenorėti daryti tokio darbo, nes testai nėra realūs ir darbas yra daromas reikalavimams vertinti, o ne sistemai testuoti.

8 Reikalavimų tvarkymas

8.1 Reikalavimų tvarkymo esmė

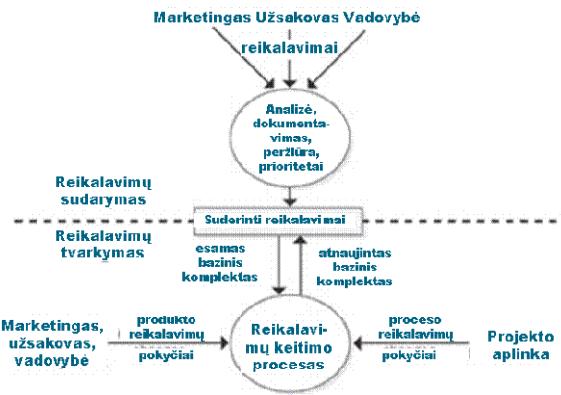
Kaip jau kalbėjome, reikalavimų tvarkymas (angl. *requirements management*) – tai veikla, vykdoma lygiagrečiai su visomis kitomis reikalavimų inžinerijos veiklomis. Be kita ko, ši veikla apima reikalavimų atributų apibrėžimą, reikalavimų klasifikavimą, reikalavimų versijavimą ir reikalavimų konfigūracijos valdymą. Tačiau visų pirma reikalavimų tvarkymas yra suprantamas kaip reikalavimų kontrolė. Tieki projekto eigoje, tiek ir jį pabaigus, reikalavimai kinta. Manoma, jog yra normalu, jei per mėnesį keičiamas maždaug vienas procentas reikalavimų. Jei ši norma nepasiekiamama, tai vykdytojai turėtų rimtai suabejoti, ar apskritai kam nors reikia jų kuriamos sistemos. Kita vertus, jei per mėnesį pakeičiamama daugiau kaip du procentai reikalavimų, projektas tampa nebeplanuojamas ir nebevaldomas [18]. Pirmiausiai (apie 1993 m.) tai buvo suprasta kariškių užsakymu vykdomuose projektuose ir, norint išvengti tokios dalykų, pradėtos naudoti formalios reikalavimų kontrolės procedūros.

Reikalavimai kinta dėl įvairių priežasčių. Natūralu, kad būsimieji kuriamos programų sistemos naudotojai projekto eigoje vis geriau ir geriau suvokia savo poreikius ir keičia reikalavimus. Sukurta programinė įranga yra naudojama informacinėse ar kokiose nors kitokiose sistemoje ir, pradėjus tą įrangą diegti, organizacija pradeda kitaip dirbti, atsiranda nauji operacinių poreikiai, dėl ko yra keičiami reikalavimai. Taip esti visuomet ir tai reiškia, kad yra kuriama iš tiesų reikalinga sistema ir kad ja iš tiesų yra naudojamas. Taip sakant, tai šalutinis projekto efektas. Pagaliau, reikalavimus keičia pats gyvenimas. Keičiami teisės aktai, kinta visa verslo sistemos aplinka. Visa tai daro poveikį kuriamos programų sistemos reikalavimams. Juo ilgiau trunka projektas, tuo daugiau reikalavimų yra pakeičiamos. Taigi, apskritai reikalavimų keitimo išvengti yra neįmanoma. Tačiau yra dar viena reikalavimų keitimo priežastis. Tai prasta reikalavimų specifikacija. Šią priežastį galima pašalinti ir tai, bent jau iš dalies, padeda padaryti geras reikalavimų tvarkymas.

Reikalavimų tvarkymo paskirtis yra susitarti su užsakovu dėl sistemos ir galbūt jos kūrimo projekto reikalavimų bei užtikrinti to susitarimo priežiūrą (angl. *maintenance*). Susitarimą įkūnija reikalavimų specifikacija ir įvairūs ją papildantys modeliai. Nereikia pamiršti, kad susitarimas yra daugiašalis arba bent jau dvišalis. Vien tik užsakovo pritarimo nepakanka. Vykdymas taip pat turi pritarti reikalavimų specifikacijai ir įsipareigoti ją įgyvendinti. Aišku, jai turėtų pritarti ir visos kitos suinteresuotosios šalys.

Reikalavimų tvarkymas apima:

- bazinio reikalavimų komplekto (angl. *requirements baseline*) sudarymą ir tvarkymą;
- reikalavimų klasifikavimą;
- reikalavimų matavimą;
- reikalavimų duomenų bazės projektavimą, formavimą ir tvarkymą;
- reikalavimų pokyčių valdymą ir projekto planų koregavimą, atsižvelgiant į aprobuotų pokyčių poveikį tam projektui;
- reikalavimų trasavimą, nuolatinį kiekvieno reikalavimo statuso ir pokyčių stebėjimą, atitinkamų ataskaitų rengimą;
- reikalavimams tvarkyti reikalingų instrumentinių priemonių įsigijimą, diegimą, aptarnavimą ir priežiūrą.

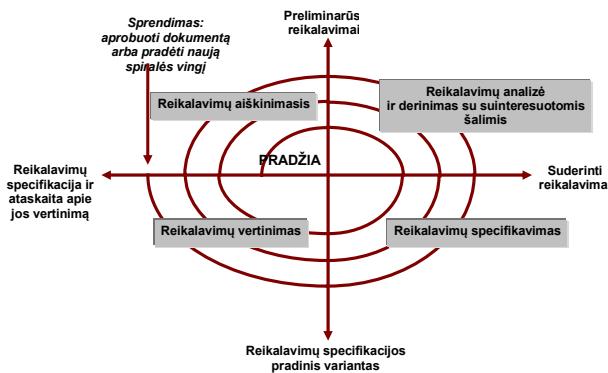


120 pav. Bazinio reikalavimų komplekto formavimas ir tvarkymas.

Bazinio reikalavimų komplekto sudarymas (120 pav.) yra konfigūracijos valdymo uždavinys. Tam tikru laiko momentu fiksuojamas konkrečios sistemos versijos ar konkrečios atmainos suderintų reikalavimų rinkinys ir paskelbiamas baziniu reikalavimų komplektu. Baziniame kompleekte surašyti reikalavimai yra laikomi galutiniai, juos keisti galima tik griežtai kontroliuojamu būdu, dažniausiai, panaudojant formalias pokyčių valdymo procedūras. Vėliau į bazinį komplektą pridedami projektavimo dokumentai, programų tekstai, testai, vartotojo dokumentacija ir visa kita informacija, susijusi su ta sistemos versija ar atmaina. Tai taip pat daroma griežtai kontroliuojamu būdu. Už bazinio komplekto tvarkymą yra atsakingas konkretus asmuo, jis priima, aprobuoja ir registruoja visus daromus to komplekto keitimus.

8.2 Iteracinis reikalavimų formulavimo proceso pobūdis

Reikalavimai formuluojami ne iš karto, bet žingsnis po žingsnio juos detalizuojant ir gerinant jų kokybę, kol pasiekama tokia reikalavimų branda, kad jau galima bandyti išsigyti tuos reikalavimus tenkinančią sistemą arba pradėti tokios sistemos projektavimą. Kai kuriuose projektuose bazinis reikalavimų komplektas yra sukuriamas anksčiau, negu reikalavimai yra iki galio išsiaiškinami ir perprantami. Šitaip elgtis yra gana rizikingas, nes vėliau, susidūrus su kokiomis nors problemomis, gali tekti atliliki brangiai kainuojančius perdibimus. Tačiau kartais vykdymo kitaip elgtis negali, nes jie yra ribojami terminu ir, kurdamai bazinį reikalavimų komplektą, stengiasi parodyti, kad jiems jau pavyko pasiekti tokią reikalavimų kokybę, kokią su turimais ištekliais buvo įmanoma pasiekti. Nesant kitos išeities, šitaip elgtis galima, tačiau reikėtų išreikštiniu būdu suformuluoti prielaidas, kuriomis remiantis buvo suformuluoti į bazinį komplektą įrašyti reikalavimai, ir aprašyti visas neišspręstas problemas.



121 pav. Iteracinis reikalavimų formulavimo proceso pobūdis (paimta iš [63]).

Bendruoju atveju bazinis reikalavimų komplektas kuriamas ne iš karto, o iteraciniu būdu (121 pav.). Kaip parodyta šiame paveikslėlyje, dažniausiai yra dirbama pagal spiralinį modelį. Kiekvieno lygmens reikalavimus atitinka ne mažiau kaip vienas spiralės vingis. Tačiau paprastai jų esti keli. Nepriklausomai nuo to, ar spiralės vingis aprašo darbą su to paties lygmens ar su skirtingu lygmenų reikalavimais, yra dirbama taip pat. Pradedama nuo reikalavimų aiškinimosi ir rinkimo, po to surinkti reikalavimai yra analizuojami, ieškoma jų trūkumų, tie trūkumai yra šalinami ir sutvarkyti reikalavimai yra derinami su visomis suinteresuotomis šalimis. Suderinti reikalavimai yra specifikuojami ir parengiamas juodraštinius reikalavimų specifikacijos variantas. Šiame dokumente surašyti reikalavimai yra vertinami, rengiama ataskaita apie vertinimo rezultatus ir rengiamas reikalavimų specifikacijos variantas, kuriame yra specifikuoti visi vertinimo metu aprobuoti reikalavimai. Šiuo dviejų dokumentų pagrindu yra sprendžiama, ar jau turima galutinė einamojo lygmens reikalavimų specifikacija, ar dar ne. Pastaruoju atveju aprobuoti įtraukiami į bazinį to lygmens reikalavimų komplektą ir yra pradedamas naujas spiralės vingis to pačio lygmens reikalavimų specifikacijai tobulinti. Naujo spiralės vingio prieikia todėl, kad analizuojant ir vertinant reikalavimus yra nustatoma, kad kokių nors reikalavimų trūksta arba kad jie yra nepakankamai išsamūs ir tikslūs.

Jeigu yra nusprendžiama, kad turimos reikalavimų specifikacijos kokybė jau yra priimtina, tai baigiamas formuoti to lygmens reikalavimų bazinis komplektas, reikalavimai lokalizuojami žemesnijame lygmenyje, nuleidžiami į tą lygmenį (žr. 5.14 poskyrį) ir šitaip yra suformuojamas pradinis žemesniojo lygmens reikalavimų rinkinys. Naujame spiralės vingyje dirbama su šiuo pradiniu reikalavimų rinkiniu, jis yra tikslinamas aiškinantis ir renkant papildomus to lygmens reikalavimus, analizuojamas, specifikuojamas ir vertinamas. Visa tai tēsiasi tol, kol nėra pasiekiamas žemiausias reikalavimų lygmuo – programų sistemos žemiausio lygmens komponentų reikalavimai. Be abejo, greta reikalavimų inžinerijos veiklų vyksta ir kitos veiklos. Suformavus kurio nors lygmens bazinį reikalavimų komplektą yra pradedami projektavimo, konstravimo ir kiti darbai, susiję su to lygmens sistemos kūrimu.

8.3 Reikalavimų klasifikavimas

Reikalavimus galima klasifikuoti pagal daugelį skirtingu kriterijų, pavyzdžiui, juos galima skirstyti į:

- produkto ir projekto;
- funkcinius ir nefunkcinius;
- pradinius (suformuluotus užsakovų), išvestinius (gautus iš aukštesnio lygmens reikalavimų)
- privalomus, pageidaujamus ir papildomus arba į kitokius nustatytaus prioritetus turinčias grupes;
- lokalizuojamus (susiejamus su konkretiais sistemos komponentais) ir nelokalizuojamus (tokius, kurių negalima susieti su konkretiais sistemos komponentais);
- stabilius ir kintančius.

Yra ir kitokių reikalavimų klasifikavimo būdų. Visi reikalavimų klasifikavimo būdai vienaip ar kitaip siejasi su reikalavimų atributais.

Apskritai reikalavimai yra klasifikuojami tam, kad juos būtų galima suskirstyti į tarpusavyje susijusių reikalavimų grupes ir po to, nagrinėjant kiekvienu grupę

atskirai, išsiaiškinti tai grupei priklausančių reikalavimų bendrybes ir atskleisti neprognozuotus jų tarpusavio sąryšius. Kitaip tariant, visų pirma yra klasifikuojama tam, kad palengvėtų reikalavimų analizę (žr. 5 skyrių). Pavyzdžiui, nagrinėjant reikalavimų bendrybes galima aptikti, kad keli kaip skirtini traktuojami reikalavimai iš tiesų specifikuojama vieną ir tą pačią kuriamos sistemos savybę ir juos reikia apjungti. Reikalavimų persidengimai ir prieštaravimai irgi dažniausiai yra tarp tos pačios klasės reikalavimų. Klasifikavimas taip pat padeda analizuoti reikalavimų išsamumą, nes analizuojant vienos klasės reikalavimus yra daug lengviau pastebeti, kad joje kokių nors reikalavimų trūksta.

Kitas reikalavimų klasifikavimo tikslas yra palengvinti reikalavimų pokyčių valdymą (žr. 8.4 poskyri). Keičiant kokiai nors klasei priklausančius reikalavimus visų pirma reikia analizuoti, kokį poveikį tai padarys kitiems tos pačios klasės reikalavimams. Todėl reikia žinoti, kokie kiti reikalavimai priklauso tai klasei ir visus tos klasės reikalavimus susieti tarpusavyje kryžminėmis nuorodomis.

Kadangi reikalavimai vieni su kitais gali būti siejami vadovaujantis pačiais įvairiausiais kriterijais, tai ir jų klasifikavimo būdų yra labai daug. Tvardinti reikalavimus vieno jų klasifikavimo būdo paprastai nepakanka. Todėl čia naudojamas vadinamasis daugiadimensinis reikalavimų klasifikavimas [107] arba, kitaip tariant, reikalavimai iš karto yra suklasifikuojami keliais būdais. Naudojant daugiadimensinį klasifikavimą, vienas ir tas pats reikalavimas gali būti priskirtas kelioms skirtinomis reikalavimų klasėms. Paprastai yra naudojami visi šio poskyrio pradžioje išvardinti klasifikavimo būdai. Funkciniai ir nefunkciniai produkto reikalavimai toliau gali būti klasifikuojami naudojant ISO/IEC 9126 standartu [ST20] numatyta klasifikaciją arba kokiui nors kitu būdu, pavyzdžiui, taip, kaip aprašyta darbe [20], t. y. skirstant reikalavimus į:

- funkcinius;
 - specifikuojančius pagrindines sistemos funkcijas;
 - specifikuojančius pagalbines sistemos funkcijas;
- nefunkcinius;
 - interfeisių;
 - naudotojo interfeisių;
 - užduočių formulavimo kalbos;
 - ergonominius;
 - sąveikos su operacine sistema interfeisių;
 - kompiuterinio ryšio interfeisių;
 - sąveikos su programavimo kalbos vykdymo aplinka interfeisių;
 - sąveikos su duomenų bazėmis interfeisių;
 - dokumentų mainų interfeisių;
 - veikimo;
 - tikslumo;
 - vaizdavimo tikslumo;
 - skaičiavimų tikslumo;
 - patikimumo;
 - robastiškumo;
 - našumo;
 - diegimo;
 - aptarnavo ir priežiūros;
 - tiražuojamumo;

- apsaugos nuo nesankcionuoto naudojimo;
- teisinius.

Galima reikalavimus klasifikuoti ir dar smulkiau [20], bet praktiniams poreikiams dažniausiai to nereikia ir paprastai, atvirkščiai, imama dar grubesnė klasifikacija, skirtant produkto reikalavimus į 5-6 grupes.

Projekto reikalavimai paprastai yra skirtomi į [20]:

- proceso;
- technologinius;
 - projektavimo technologijos;
 - programavimo technologijos;
 - bandymų;
 - konfigūracijos valdymo;
- kokybės valdymo;
- vykdymo terminų;
- finansavimo;
- rezultatų ir jų pateikties;
- rezultatų aprobatavimo;
- garantijų;
- ginčų sprendimo tvarkos.

Projekto reikalavimus galima klasifikuoti ir daugeliu kitų būdų.

Įdiegti organizacijoje reikalavimų klasifikavimą nebrangu ir nesudėtinga.

Pagrindinis darbas yra parengti reikalavimų schema ir visiems darbuotojams ją tinkamai išaiškinti. Programuotojai dažnai neskiria sistemos užduočių ir jos funkcijų, nesuvokia, kas tai yra užduočių formulavimo kalba ir netgi suabsoliutina kokį nors vieną užduočių formulavimo būdą, pavyzdžiui, tą, kuris dominuoja MS Windows operacinėse sistemose. Kita problema yra tikslumo reikalavimai. Gana dažnai klaidingai yra manoma, kad vaizdavimo ir apdorojimo paklaidos yra sietinos tik su skaitiniais duomenimis, o ne su bet kuria sistemos apdorojama informacija, išskaitant grafinę, audio ir video informaciją. Dar viena problema – robastiškumo reikalavimai. Daugelis analitikų ir programuotojų neskiria jų nuo patikimumo reikalavimų ir nelabai supranta, kam jie yra apskritai reikalingi. Galimos ir kitos vienų ar kitų reikalavimų suvokimo problemos. Tačiau visoms joms išspręsti paprastai pakanka 1-2 dienų seminaro, kuriame pateikiame ir aptariame konkretūs skirtingoms klasėms priskirtini reikalavimai.

8.4 Reikalavimų matavimo problemos

Dėl praktinių sumetimų dažnai esti svarbu įvertinti konkrečios programų sistemos reikalavimų „apimtį“. To, pavyzdžiui, reikia vertinant reikalavimų pakeitimų „apimtį“, vertinant tiketiną projekto kainą, jo trukmę, sukurtos programų sistemos priežiūros kaštus ir kt. Funkcinių reikalavimų apimčiai vertinti standartas IEEE14143.1-00 [ST16] numato funkcinės apimties matą (angl. *functional size measurement*). ISO/IEC ir kai kurie kiti standartai siūlo kaip šiuo matuoti funkcinių reikalavimų apimtį.

Naudojami ir kiti reikalavimams matuoti skirti matai. Vienas iš jų yra reikalavimų kintamumas (angl. *requirements volatility*). Tai per mėnesį pakeistų reikalavimų procentas. Kaip jau minėjome, vyrauja nuomonė, kad normalus reikalavimų kintamumas yra 1% per mėnesį. Taigi, atrodo, paprasčiausiai pakanka stebeti, ar nėra nukrypstama nuo šios normos. Tačiau yra ir kitaip manančių.

Pavyzdžiui, Kolorado universitete (JAV) atlikti tyrimai parodė, jog priimtinas keičiamų reikalavimų procentas priklauso nuo laiko, kada tie pakeitimai yra padaryti:

„Gauti rezultatai rodo, jog kuo mažiau laiko likus iki sistemos testavimo pabaigos yra keičiami reikalavimai, tuo daugiau defektų lieka sistemoje. Priklausomybė yra eksponentinė. Projekto pradžioje daromi keitimai gali neturėti beveik jokių pasekmių, o tie, kurie yra daromi vėliau, gali labai ženkliai padidinti defektų skaičių.“ [38].

Kai kurie autoriai mano dar kardinaliau. Jų nuomone, priimtinas keičiamų reikalavimų procentas priklauso „nuo daugelio veiksnių, įskaitant projekto stadiją, vykdytojus, reikalavimų sudėtingumą, sistemos sudėtingumą, sistemos dydį, užsakovo lūkesčius, projekto vykdymo terminus, sistemos kūrimo technologiją, naudojamas metodikas, naudojamas instrumentines priemones ir kt.“ [54].

Tačiau pats svarbiausias reikalavimų matas yra jų įgyvendinimo laipsnio matas. Apie tai daug buvo kalbėta 2.3, 2.7 ir 5.13 poskyriuose. Šis matas paprastai yra naudojamas nefunkcinių reikalavimų įgyvendinimo laipsniui matuoti, nes funkcinių reikalavimų įgyvendinimui matuoti dažniausiai pakanka paprasto kokybinio mato Taip/Ne. Nefunkciniai reikalavimai specifikuojant tokias sistemos savybes, kaip patikimumas, sauga ar patogumas. Šių savybių priimtinos reikšmės gali kisti tam tikrose ribose ir todėl, specifikuojant nefunkcinius reikalavimus, reikia tas ribas nurodyti. Kokiais matais tas savybes matuoti, nustato ISO/IEC programų sistemų kokybės standartas[ST20]. Yra įvairių pasiūlymų ir kaip reikia matuoti tas savybes, tačiau praktikoje atlikti tokius matavimus visuomet yra gana sudėtinga, nes tam reikia suprojektuoti ir sukurti visą specialią testų sistemą. Kita vertus, specifikavus leistinas tų sistemos savybių reikšmes ir atlikus atitinkamus matavimus, yra išvengiama ginčų su užsakovu apie tai, ar visi reikalavimai buvo tinkamai įgyvendinti.

Formuluojant matuojamus reikalavimus, reikia turėti omenyje, kad nefunkcinių reikalavimų paprastai konfliktuoja vieni su kitais (žr. 4.4 poskyri). Pavyzdžiui, gali būti pareikalauta, kad sistema apdorotų N transakcijų per sekundę ir kad jos reakcijos laikas neviršytų M sekundžių. Tačiau, paanalizavus sistemai eksploatuoti naudojamos technikos charakteristikas, gali paaiškėti, kad prie maksimalių apkrovų (t. y. prie N transakcijų per sekundę) procesorius yra taip apkraunamas, kad gauti rezultatus per M sekundžių tampa neįmanoma. Todėl nustatant priimtinas sistemos savybių reikšmes negalima jų nagrinėti po vieną. Reikia analizuoti savybių visumą ir gauti priimtiną savybių balansą. Be to, reikia turėti omenyje, kad kai kurių savybių matavimas gali kainuoti labai brangiai. Pavyzdžiui, specifikuojant patikimumą nustačius, kad priimtina patikimumo reikšmė yra 1 triktis 100 000 transakcijų, reikalavimo įgyvendinimui patikrinti prireiktų kelių šimtų tūkstančių testų.

Diegiant matuojamus reikalavimus visuomet yra susilaukiama didelio pasipriešinimo. Priešinasi ir užsakovas, ir vykdytojo vadybininkai, ir programuotojai. Išskyrus technologinių procesų ar įrenginių valdymo sistemų atvejus, užsakovas dažniausiai bijo formuliuoti labai tikslius reikalavimus ir prisiimti visą atsakomybę už sistemos kokybę. Be to jis retai kada ir pajėgia tai padaryti. Vykdytojo vadybininkams taip pat yra paprasčiau, kada jie turi didesnę veikimo laisvę ir gali neformaliai įrodinėti užsakovui, kad sukurtoji sistema yra aukštostas kokybės. Programuotojai gi dažniausiai nėra susipažinę su saugių, patikimų ar patogių sistemų kūrimo metodais ir net neįsivaizduoja, kaip jie galėtų sukonstruoti testus taip specifikuotoms sistemos savybėms patikrinti. Todėl nusprenodus organizacijoje diegti matuojamus reikalavimus, tą reikia daryti palaipsniui, per keletą projektų, ir pradėti nuo pačių paprasčiausių, lengvai matuojamų sistemos savybių.

8.5 Reikalavimų atributai

Reikalavimai susideda ne tik iš specifikacijų, ką reikia padaryti kuriant sistemą, bet ir iš papildomos informacijos, padedančios interpretuoti ir tvarkyti tuos reikalavimus. Ši informacija apima reikalavimų klasifikavimo būdus, jų tikrinimo metodus ir sistemos tinkamumo testavimo planus. Taip pat galima aprašyti kiekvieno reikalavimo motyvaciją, nurodyti jo šaltinį, aprašyti jo pokyčių istoriją ir pateikti kitą papildomą informaciją. Tokia informacija padeda geriau suprasti reikalavimus ir yra reikalinga su tais reikalavimais dirbantiems projektuotojams, testuotojams, vadybininkams bei kitims specialistams. Ji užduodama susiejant su kiekvienu reikalavimu tam tikrą atributų rinkinį.

Ir reikalavimai, ir jų atributai paprastai yra saugomi specialistai tam skirtose duomenų bazėse, vadinamose projekto repozitorijais. Priklausomai nuo to, kaip yra realizuotas repozitorijus, jo struktūra gali būti projektuojama kiekvienam konkrečiam projektui arba būti užduota iš anksto ir nekeičiama. Jei repozitorijus yra realizuotas kaip kompiuterinė lentelė arba jei jis yra realizuojamas panaudojant kokią nors universalaus pobūdžio DBVS, galima suprojektuoti ir naudoti bet kokios norimos struktūros repozitorijų. Tačiau tokiu atveju patiemis reikia programuoti ir visas reikalavimų tvarkymo procedūras. Išigyjant kokią nors komercinę reikalavimams tvarkyti skirtą instrumentinę sistemą, repozitorijaus struktūra gali būti iš anksto nustatyta ir nekeičiama. Tačiau ir tokiais atvejais dažniausiai yra reikalaujama tiktais, kad bazėje būtų tam tikri sistemių reikalingi atributai, o kitus atributus leidžiama apibrėžti patiemis. Kitaip tariant, naudojant komercines instrumentines sistemas, bent jau iš dalies repozitorijaus struktūrą kiekvienam projektui galima keisti.

Reikalavimai gali turėti labai daug įvairių atributų, tačiau ne visi jie yra reikalingi reikalavimams tvarkyti. Todėl, norint suprojektuoti repozitorijaus struktūrą, reikia žinoti, kokia informacija apie reikalavimus dažniausiai esti reikalinga arba, kitaip tariant, kokie reikalavimų atributai gali būti naudingi tvarkant reikalavimus. Projektuojant repozitorijaus struktūrą iš tų atributų pasirenkami tie, kurie yra prasmingi konkrečiam projektui.

Kokio nors standartinio tokio atributų rinkinio nėra. Tačiau yra sukaupta tam tikra praktinė patirtis. Todėl prasmingus reikalavimų atributus aptarsime vadovaudamiesi būtent tokia patirtimi [2]. Bene svarbiausias šio darbo privalumas yra atributų parinkimo metodika. Siūloma nagrinėti reikalavimų inžinerijos procesą ir visą programų sistemos kūrimo procesą, ir kiekvienai šiuose procesuose numatytais veiklais analizuoti, kokios informacijos apie reikalavimus prireiks vykdant tas veiklas. Su kiekvienu reikalavimų inžinerijos proceso veikla galima susieti tam tikrą rinkinį atributų, kuriems vykdant tą veiklą yra priskiriamos reikšmės. Tačiau paprastai tos reikšmės yra naudojamos vykdant ne tik tą, bet ir kitas reikalavimų inžinerijos bei kitų programų sistemai kurti skirtų procesų veiklas. Taigi, parenkant atributus, reikia vadovautis dviem kriterijais: kokios informacijos reikia kiekvienai iš veiklų ir kokią informaciją gali sukurti kiekviena iš veiklų. Tokia atributų parinkimo metodika pasiteisino daugelyje projektų ir, mano nuomone, ja tikrai yra tikslinga vadovautis. Tuo tarpu pateiktas atributų rinkinys yra tiktais pavyzdinis ir galima ne tik iš jo pašalinti konkrečiam projektui nereikalingus atributus, bet ir papildyti jį naujais atributais.

Pereisime prie konkrečių atributų aptarimo.

Pats svarbiausias reikalavimo atributas yra *reikalavimo žymuo* (angl *unique identifier*). Šis atributas turi būti bet kuriame repozitorijuje. Dėl daugelio priežasčių – reikalavimų keitimas, papildomų reikalavimų pridėjimas, reikalavimų anuliavimas, reikalavimų trasavimas ir kt. – yra reikalaujama, kad kiekvienas reikalavimas būtinai

turėtų unikalų žymenį. Naudojamos yra skirtingos reikalavimų žymėjimo sistemos, bet paprastai patogiausios esti hierarchinės reikalavimų numeravimo sistemos.

Dabar panagrinėkime, kokių atributų reikia atliekant reikalavimų aiškinimąsi. Šiuo atributu pavadinimai ir jų tipinės reikšmės pateikiti 8 lentelėje. Priminsime, kad aiškinantis reikalavimus yra dirbama ne tik su pačiais reikalavimais, bet ir su įvairia kita informacija, pavyzdžiui, su įvairiais ribojimais. Kadangi su šia informacija yra dirbama panašiai, kaip ir su reikalavimais, tai ją yra tikslinga saugoti taip pat projekto repozitorijuje. Todėl prieikia specialių atributų, leidžiančių nustatyti repozitorijuje saugomo konkretaus informacijos elemento pobūdį. Nagrinejant tipines šio atributo reikšmes, reikia turėti omenyje, kad reikalavimų specifikacija arba bent jau kai kurios jos dalys iš projekto repozitorijaus paprastai yra kuriamas automatiškai ir todėl repozitorijuje turi būti saugomos terminų apibrėžtys bei specifikacijos struktūrinių dalių pavadinimai.

8 lentelė. Atributai, naudojami aiškinantis reikalavimus.

Atributas	Tipinės atributo reikšmės
Informacijos pobūdis (angl. <i>information type</i>)	<ul style="list-style-type: none"> • IS reikalavimas; • PS reikalavimas; • vartotojo reikalavimas; • žinios apie verslo sistemą (pvz., vizija, tikslai ir t.t.); • sistemos kontekstą aprašanti informacija (t. y. konteksto diagramos fragmentas); • termino apibrėžtis; • specifikacijos struktūrinės dalies pavadinimas; • kokia nors kita informacija.
Reikalavimo tipas (angl. <i>requirement type</i>)	<ul style="list-style-type: none"> • funkcinis reikalavimas; • nefunkcinis reikalavimas; • proceso reikalavimas; • ribojimas; • nereikalavimas (t. y. kokia nors kita informacija).
Reikalavimo potipis (angl. <i>secondary type</i>)	<ul style="list-style-type: none"> • našumo reikalavimas; • patikimumo reikalavimas; • apsaugos reikalavimas; • saugos reikalavimas; • ir t.t.
Reikalavimo svarba (angl. <i>key feature</i>)	Dažniausiai reikalavimai skirstomi į dvi grupes: <i>esminiai</i> ir <i>neesminiai</i> reikalavimai. Taip pat gana dažnai reikalavimai anotuojami naudojant keturias reikšmes: <i>privalomas reikalavimas, įgyvendintinas reikalavimas, norimas reikalavimas, pageidautinas reikalavimas</i> (angl. <i>moscow list: must, should, could, would</i>). Naudojant kokybės funkcijų sklaidą, reikalavimai skirstomi į tris grupes: <i>bazinė kokybė, normali kokybė, išskirtinė kokybė</i> . Galimi ir kiti reikalavimų klasifikavimo pagal jų svarbą būdai.
Reikalavimo šaltinis (angl. <i>source data</i>)	Atributo reikšme yra nuoroda į interviu protokolą ar kokį nors kitą dokumentą, kuriame pirmą kartą buvo suformuluotas tas reikalavimas.

Atributas	Tipinės atributo reikšmės
Reikalavimo pagrindimas (angl. <i>rationale</i>)	Atributo reikšme yra arba trumpas reikalavimą pagrindžiantis tekstas, arba nuoroda į dokumentą, kur tokis pagrindimas yra padarytas.
Šalis, suinteresuota reikalavimu (angl. <i>owner or stakeholder</i>)	Atributo reikšmė nurodo kam sistemoje reikalinga tokia paslauga (funkcionalumas) arba kas reikalauja, kad sistema turetų tokią savybę ar kad joje būtų tenkinamas tokis ribojimas. Šiuo atveju suinteresuotaja šalimi gali būti ne tik pareigybė ar padalinys, bet ir įstatymas ar kitas norminis aktas arba verslo taisykla.
Žinių apie verslo sistemą detalės (angl. <i>domain knowledge record details</i>)	Atributo reikšme yra nuoroda į atitinkamą verslo taisyklę ar kokį nors kitą dokumentą, kuriame aprašytos tos žinių detalės. Tokia nuoroda yra galima tik tuomet, kuomet atributo informacijos pobūdis reikšmė yra „žinios apie verslo sistemą“
Žinių apie verslo sistemą statusas (angl. <i>domain knowledge checked</i>)	Atributo reikšmė nurodo ar tos žinios jau yra patikrintos ir jomis jau galima naudotis, ar jas dar toliau reikia tikrinti.
Nurodymas projektuotojui (angl. <i>guidance/notes</i>)	Atributo reikšme yra standarto, metodo ar technologijos, kuriuos rekomenduojama naudoti reikalavimui realizuoti, pavadinimas.

Kaip buvo kalbėta bakalauro studijų pakopos skirtame programų sistemų inžinerijos kurse [20], gerai suformuluotas reikalavimas turi turėti tam tikras savybes. Jis turi būti atomarinis, abstraktus, įgyvendinamas, išsamus, suprantamas, glaustas ir verifikuojamas. Atliekant reikalavimų testavimą, yra tikrinama kiekvieno reikalavimo kokybė, t. y. kokių mastų jis turi aukščiau išvardintas savybes. Iš repozitorijuje sukauptų reikalavimų yra generuojama reikalavimų specifikacija. Ji taip pat turi turėti tam tikras savybes. Joje pateiktai reikalavimai turi būti nepriestaringi, susieti vienas su kitu ir nepersidengti (būti unikalūs). Atliekant reikalavimų testavimą, yra tikrinama ar tokią specifikaciją galima sugeneruoti. Atributai, naudingi reikalavimų testavimui atliki, yra pateikti 9 lentelėje.

9 lentelė. Atributai, naudojami testuojant reikalavimus.

Atributas	Tipinės atributo reikšmės
Reikalavimui tikrinti atliktu testų sąrašas (angl. <i>individual requirement& comments</i>)	Atributo reikšmė pasako, kurios iš reikalaujamų reikalavimo savybių jau yra patikrintos.
Reikalavimo testavimo rezultatai (angl. <i>testing results</i>)	Atributo reikalavimo testavimo rezultatai parodo, kokias reikalaujančias savybes reikalavimas turi, o kokių – ne.
Reikalavimų rinkiniui tikrinti atliktu testų sąrašas (angl. <i>requirement set&comments</i>)	Atributo reikšmė pasako, kurios iš reikalaujamų reikalavimų rinkinio savybių yra patikrintos.
Reikalavimų rinkinio	Atributo reikšmė parodo, kokias reikalaujančias savybes šis

Atributas	Tipinės atributo reikšmės
testavimo rezultatai (angl. <i>requirement set testing results</i>)	rinkinys turi, o kokių – ne.

Dar viena svarbi veikla, kuriai repozitorijuje reikia numatyti atitinkamus atributus, yra reikalavimų inžinerijos darbų planavimas ir rizikos valdymas. Šiai veiklai vykdyti naudojami atributai pateikti 10 lentelėje.

10 lentelė. Atributai, naudojami planuoti reikalavimų inžinerijos darbus ir valdyti riziką.

Atributas	Tipinės atributo reikšmės
Prielaidos (angl. <i>assumptions</i>)	Atributas įgyja reikšmę „taip“, jei reikalavimas arba rezultatas remiasi kokiomis nors prielaidomis, ir „ne“, jeigu tokią prielaidą nėra.
Turiniai (angl. <i>issues</i>)	Atributo reikšmė parodo, ar yra žinomi kokie nors papildomi reikalavimo, žinių apie verslo sistemą ar rezultato aspektai. Pavyzdžiui, gali būti parodyta, ar reikalavimas teisingas turimų žinių apie verslo sistemą požiūriu arba ar yra tikimasi, kad šiame įraše aprašytos žinios apie verslo sistemą kis projekto eigoje.
Pasitenkinimo laipsnis (angl. <i>satisfaction factor</i>)	Atributo reikšmė pasako, kaip reaguos užsakovas, jei reikalavimas bus įgyvendintas (pvz., apsidžiaugs, manys, kad taip ir turėjo būti ir pan.).
Nepasitenkinimo laipsnis (angl. <i>unsatisfaction factor</i>)	Atributo reikšmė pasako, kaip reaguos užsakovas, jei reikalavimas nebus įgyvendintas (pasipiktins, perdaug nesisielos ir pan.).
Prioritetas/īgyvendinimo terminas (angl. <i>priority/release date</i>)	Atributo reikšmė arba nurodo reikalavimo īgyvendinimo prioritetą, arba pasako, kada (t. y. kurioje sistemos versijoje) reikia īgyvendinti reikalavimą.
Rizikos veiksniai (angl. <i>risk factors</i>)	Atributo reikšmė aprašo reikalavimo operacinio ir techninio īgyvendinamumo rizikos veiksnius. Pavyzdžiui, gali būti atsakoma į klausimus: Koks reikalavimo ir jo īgyvendinamumo būdo sudėtingumo laipsnis? Koks reikalavimui īgyvendinti reikalingos technologijos ir verslo technologijos, kurioje bus naudojamas numatytu funkcionalumu, brandos laipsnis? Ar vykdytojų ir vartotojų gebėjimų brandos laipsnis pakankamas reikalavimui īgyvendinti ir pasinaudoti šitaip sukurtu sistemos funkcionalumu?
Rizikos laipsnis (angl. <i>risk</i>)	Atributo reikšmė nusako, kiek tikėtina, kad reikalavimas bus īgyvendintas. Tai gali būti nusakyta skaitine reikšme arba tekstu, su kurio dažniausiai yra susiejamas išsamesnis komentaras. Rizikos laipsniui nustatyti naudojamas atributų <i>Rizikos veiksniai</i> ir <i>Reikalavimo testavimo rezultatai</i> reikšmėmis (pvz., ar reikalavimas yra išsamus, ar jis īgyvendinamas ir kt.). Kartais atributo reikšmė yra naudojama aprašyti ir tai, kokios būtų reikalavimo neįgyvendinimo pasekmės, koks tikėtinumas

Atributas	Tipinės atributo reikšmės
	(angl. <i>likelihood</i>), kad šitaip iš tiesų įvyks, atsižvelgiant į atributų <i>Prielaidos</i> ir <i>Turiniai</i> reikšmes. Gali būti pateiktas ir tik kokybinis vertinimas, kuris taip pat yra gaunamas vadovaujantis minėtų atributų reikšmėmis.
Nauda (angl. <i>benefits</i>)	Atributo reikšmė nusako, ką laimės verslas iš to, kad reikalavimas bus įgyvendintas. Atributas įgyja reikšmę tik įrašuose, aprašančiuose reikalavimus. Kitokio pobūdžio informaciją aprašančiuose repozitorijaus įrašuose atributo reikšmė yra neapibrėžta.
Reikalingi atlikti darbai (angl. <i>work planning</i>)	Atributo reikšmė nurodo, kokie darbai iš tų, kuriuos reikia atliki su reikalavimu, dar nebuvu atlikti ir turi būti padaryti (pvz., reikalavimą reikia toliau dekomponuoti, testuoti, tikslinti ir t.t.).
Statusas (angl. <i>status</i>)	Atributo reikšmė aprašo reikalavimo statusą: dar nenuspresta, ar jis apskritai reikalingas; dekomponuotas; galutinai suformuluotas; patikrintas (pratestuotas); aprobuotas.

Atributai *Pasitenkinimo laipsnis*, *Nepasitenkinimo laipsnis*, *Prioritetas/igyvendinimo terminas* yra naudojami reikalavimams prioretizuoti. Kiti atributai – tolimesniems darbams planuoti ir rizikos veiksniams vertinti.

Kaip jau minėjome, su reikalavimais intensyviai yra dirbama atliekant ne tik reikalavimų inžinerijos, bet ir kitus programų sistemos kūrimo darbus. Ypač daug su reikalavimais dirbama projektuojant sistemą. Projektavimo metu reikalavimai yra dekomponuojami, lokalizuojami, nuleidžiami žemyn, su jais atliekami kiti veiksmai. Projektavimo darbams atlikti naudojami atributai pateikti 11 lentelėje.

11 lentelė. Atributai, naudojami projektuojant sistemą.

Atributas	Tipinės atributo reikšmės
Kur lokalizuotas (angl. <i>functional allocation</i>)	Atributo reikšmė nusako, kuriose sistemos versijoje ir koriuose tų versijų komponentuose yra lokalizuotas atributas.
Nuoroda į projektinę dokumentaciją (angl. <i>design reference</i>)	Atributo reikšmė susieja reikalavimą su atitinkamu projektinės dokumentacijos fragmentu.
Realizavimo technologijos (angl. <i>design review factors</i>)	Atributo reikšmė nusako, kokias technologijas numatoma panaudoti reikalavimui numatytam funkcionalumui, patikimumui, našumui apsaugai ar kt. įgyvendinti.
Projekto peržiūros pastabos (angl. <i>design review comments</i>)	Atributo reikšmė kiekvienai problematiškai atributu <i>Realizavimo technologijos</i> numatytais technologijai aprašo argumentus, kuriais vadovaujantis buvo nuspresta tą technologiją naudoti.

Pagaliau, specialūs atributai turi būti numatyti dar dviem veiklom: reikalavimų verifikavimui ir vertinimui planuoti; verifikavimo ir vertinimo rezultatų peržiūrai atlikti. Šie atributai pateikti 12 ir 13 lentelėse.

12 lentelė. Reikalavimų verifikavimą ir vertinimą planuoti naudojami atributai.

Atributas	Tipinės atributo reikšmės
VV planavimas (angl. <i>V&V planning</i>)	Atributo reikšmė nurodo, kaip turi būti patikrintas reikalavimas, t. y. nurodo, kokius verifikavimo ir vertinimo metodus (peržiūrą, inspektavimą, maketavimą, analizę, bandymą ir kt.) reikia panaudoti. Tuos metodus turėtų nustatyti užsakovas. Skirtingose projekto stadijose gali būti naudojami skirtingi VV metodai. Kokius metodus reikia naudoti, aprašo pora atributų <i>VV stadija</i> ir <i>VV metodai</i> .
VV stadija (angl. <i>V&V planning</i>)	Atributo reikšmė nurodo projekto stadiją, kurioje reikia panaudoti atitinkamą VV metodą.
VV metodai (angl. <i>V&V method type</i>)	Atributo reikšmę sudaro nuorodos į atributę <i>VV planavimas</i> nurodytus metodus.
Poreikis atsisakyti reikalavimo (angl. <i>waiver or deviation needed</i>)	Atributo reikšmė yra „taip“ arba „ne“. Tą reikšmę nustato užsakovas, atsižvelgdamas į VV rezultatus.
VV rezultatas (angl. <i>V&V Results Summary</i>)	Atributo reikšme yra reikalavimo verifikavimo ir vertinimo rezultatų reziume.
Nuoroda į VV protokolą (angl. <i>V&V results source document</i>)	Atributo reikšme yra nuoroda į dokumentą, detaliai aprašantį reikalavimo verifikavimo ir vertinimo rezultatus.

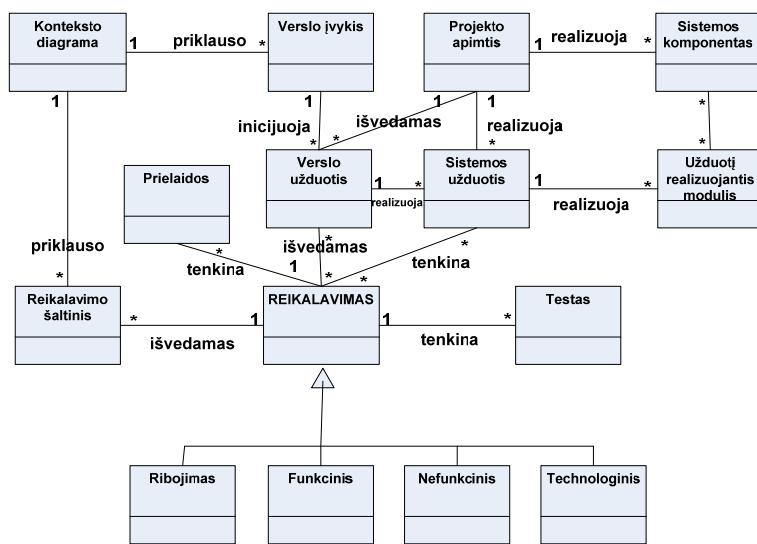
13 lentelė. Verifikavimo ir vertinimo rezultatų peržiūrai naudojami atributai.

Atributas	Tipinės atributo reikšmės
VV peržiūros grupė (angl. <i>reviewers name</i>)	Atributo reikšme yra reikalavimo verifikavimo ir vertinimo rezultatų peržiūrą atlikusių asmenų sąrašas.
VV trasa (angl. <i>V&V Trace Object</i>)	Atributo reikšme yra „taip“ ir „ne“ reikšmių sąrašas, kuriame kiekvienam reikalavimo verifikavimo ir vertinimo metu atliktam tikrinimui nurodoma ar jis buvo sėkmingas, ar ne.
VV trasos vertinimas (angl. <i>V&V trace suitability</i>)	Atributo reikšme yra „taip“ ir „ne“ reikšmių sąrašas, kuriame kiekvienam reikalavimo verifikavimo ir vertinimo metu atliktam tikrinimui nurodoma ar peržiūros grupės nuomone jis buvo tinkamas tam reikalavimui tikrinti.
Progresas (angl. <i>V&V Status Summary</i>)	Atributo reikšmė (nederintas su užsakovu, deramas su užsakovu, pasiūlytas aprobuoti, aprobuotas ir pan.) pasako kaip toli reikalavimų inžinerijos procese darbas su reikalavimu pasistūmėjo į priekį.
Tolimesnio VV plano vertinimas (angl. <i>V&V planning acceptance</i>)	Atributo reikšmė pateikia atributu <i>VV planavimas</i> aprašyto tolimesnio VV proceso vertinimą.

Projekto repozitorijuje saugomi įrašai apie reikalavimus bei kitą informaciją tarpusavyje yra susiejami įvairiais ryšiais. Tam tikslui dažniausiai yra naudojami šitokie ryšiai:

- **tenkina** (angl. *satisfies*): sieja žemesnio lygmens reikalavimą su aukštesnio lygmens reikalavimu, kurį jis realizuoja;
- **išvedamas** (angl. *derives*): sieja žemesnio lygmens reikalavimą su aukštesnio lygmens reikalavimu, iš kurio jis buvo išvestas nuleidžiant reikalavimus žemyn;
- **konfliktuoja** (angl. *conflicts*) sieja du vienas kitam prieštaraujančius reikalavimus.

Galimi ir įvairūs kiti ryšiai (žr. 122 pav.). Kokius ryšius galima apibrėžti, nustato reikalavimams tvarkyti naudojama instrumentinė sistema, nes ji turi mokėti repozitorijuje apibrėžtus ryšius apdoroti.



122 pav. Paprasto repozitorijaus struktūrą aprašančios DB schemas pavyzdys.

122 paveikslėlyje pateiktas paprasto repozitorijaus struktūrą aprašančios DB schemas pavyzdys. Čia reikalavimai per ryšius yra susieti su jų šaltiniais, prielaidomis, kuriomis jie yra grindžiami, jų įgyvendinimą tikrinančiais testais, verslo užduotimis, iš kurių jie yra išvedami ir kt.

8.6 Pokyčių valdymas

Efektyvus pokyčių valdymas reikalauja, kad reikalavimų būklė būtų nuolat stebima ir kontroliuojama. Tai padeda daryti specialiai tam tikslui skirtos instrumentinės sistemos (žr. 8.8 poskyrij), pavyzdžiui, *Requisite Pro*, *DOORS*, *RTM*, *Caliber-RM*). Jos padeda stebeti, kas vyksta su reikalavimais skirtingose sistemas gyvavimo ciklo stadijose, tvarkyti bazinius reikalavimų komplektus bei dokumentuoti reikalavimus. Tokios instrumentinės sistemos ypač svarbios projektuose, kuriuose reikalavimų specifikacija yra neatskiriamai sandorio su užsakovu dalis. Technologo požiūriu, tokį instrumentinių sistemų įsisavinimas ir naudojimas niekuo nesiskiria nuo kitų instrumentinių sistemų įsisavinimo ir naudojimo. Reikia suformuluoti instrumentinės sistemos reikalavimus, susieškoti tuos reikalavimus tenkinančią sistemą ir parengti jos diegimo planą.

Programinės įrangos gebėjimų modelis (angl. *Software Capability Maturity Model*, SW-CMM), o iš dalies taip pat sistemų inžinerijos gebėjimų brandos modelis (angl. *Systems Engineering CMM*) ir programinės įrangos įsigijimo gebėjimų brandos

modelis (angl. *Software Acquisition CMM*), reikalavimų tvarkymą traktuojama kaip 2-jo brandos lygmens esminę proceso sritį (angl. *key process area*). Tuose modeliuose reikalavimų tvarkymas iš esmės netgi yra tapatinamas su pokyčių valdymu. Kitos reikalavimų tvarkymo veiklos, visų pirma reikalavimų trasavimas ir jų statuso kontrolė, traktuojami kaip SW-CMM 3-jo brandos lygmens esminės proceso srities dalis. Diegiant CMM, būtina prisiminti, kad pirmiausia reikia įgyvendinti užsakovo interfeiso reikalavimų pokyčių valdymą. Kitas žingsnis – visose sistemos gyvavimo ciklo stadijose palaikomų reikalavimų statuso trasų konstravimas ir šitaip surinktos informacijos panaudojimas projektui valdyti.

Teisingai organizuotas reikalavimų pokyčių valdymas leidžia geriau kontroliuoti projekto kaštus ir vykdymo terminus, nes, padarius kokius nors reikalavimų pakeitimų, iš karto išsiaiškinama, kokį poveikį tie pakeitimai padarys visam projektui, kiek tai kainuos ir kaip tai paveiks sistemos vykdymo terminus. Posakis „užsakovas visuomet teisus“ yra šiek tiek perdėtas. Griežtai vadovaudamas šiuo posakiu, vykdytojas gali patirti nemažų nuostolių dėl užsakovo ar kitų suinteresuotujų šalių daromų dažnų ir menkai apgalvotų reikalavimų keitimų. Todėl patariama reikalavimų keitimo procesą formalizuoti ir užsakovui ar kam nors kitam panorėjus keisti reikalavimus, jam išaiškinti visas tokio keitimo pasekmes. Šitaip užsakovas ir kitos suinteresuotosios šalys pripranta daryti pakeitimus labai atsakingai, nes žino, kad tai atsilieps į projekto kainą ir terminus. Kita vertus, vykdytojo inžinerinis personalas taip pat yra disciplinuojamas, nes visi žino, kad jeigu kokius nors pakeitimų reikia daryti dėl jų kaltės, tarkime, dėl prastai atlikto analitikų darbo, tai iš karto išaiškės ir visus nuostolius teks dengti patiemis, dirbant viršvalandžius, mokant baudas užsakovui ar kitaip prarandant dalį savo uždarbio. Be abejo, net ir tokiais atvejais vykdytojas pats vienas jokių pradinių reikalavimų keisti negali. Juos turi aprobuoti užsakovas bei kitos suinteresuotosios šalys. Vykdytojas gali keisti tik projektavimo ar realizavimo reikalavimus. Tokie reikalavimų pokyčiai turi būti valdomi lygiai taip pat, kaip ir užsakovo ar kitų suinteresuotujų šalių daromi reikalavimų keitimai. Būtina įvertinti jų pasekmes ir spręsti ar jie yra tikslingi. Be abejo, visus tokią pakeitimų darymo kaštus tenka dengti vien tik užsakovui. Todėl planuojant projektą ir darant jo rizikos veiksnių analizę, reikia įvertinti ir į šį veiksnį ir numatyti tam atitinkamas lėšas projekto biudžete.

Apibrėžiant reikalavimų pokyčių valdymo taisykles, yra nustatomi pokyčių valdymo procesai, procedūros ir standartai. Aišku, visa tai turi būti nustatoma, atsižvelgiant tiek į visą konkretaus programų sistemų inžinerijos proceso kontekstą, tiek ir į konkretaus reikalavimų inžinerijos proceso ypatumus. Tačiau tos taisykles visuomet turi numatyti, kad apie bet kurį siūlomą daryti reikalavimų keitimą turi būti surinkta tokia pati informacija ir pagal tas pačias procedūras turi būti vertinami pakeitimo įgyvendinimo kaštai, projekto vykdymo terminų pokyčiai ir nauda, kurią gaus verslas įgyvendinus tą pakeitimą.

Taigi reikalavimų pokyčių valdymo taisykles turi nustatyti:

- kaip ir kokia forma turi būti teikiami pageidavimai keisti reikalavimus ir kokia informacija turi būti surinkta pakeitimui vertinti;
- pakeitimo pasekmių analizės ir vertinimo procedūras ir joms vykdyti reikalingus reikalavimų atributus (žr. 8.5 poskyrį) ir reikalavimų trasų konstravimo būdus (žr. 8.7 poskyrį);
- kas nagrinės siūlomus pakeitimus bei jų pasekmes, kas tuos pakeitimus aprobuos arba atmes ir kokiais terminais visa tai turi būti daroma;
- kaip bus registruojami daromi pakeitimai ir kokia informacija turi būti saugoma pakeitimų registre.

Patariama bet kuriame didesniame projekte pakeitimų registrą realizuoti kaip kompiuterinę duomenų bazę. Tada, savaime suprantama, turi būti nuspręsta, kokios programinės priemonės bus naudojamos tai bazei tvarkyti ir kaip ji bus susieta su pačių reikalavimų baze (žr. 8.5 poskyri). Šiuo metu yra sukurta keletas specialiai pokyčiams valdyti skirtų instrumentinių sistemų [107]. Tačiau dauguma jų neatsižvelgia į tai, kad esti skirtingų lygmenų reikalavimai, ir yra pritaikytos tik darbui su programų sistemų reikalavimų pokyčiais. Be to, tokios sistemos dažniausiai yra susietos su kokia nors komercine konfigūracijos valdymo instrumentine sistema ir negali būti naudojamos vienos pačios. Nedideliuose projektuose pakeitimų registrą galima realizuoti kaip kompiuterinę lentelę, panaudojant tam, pavyzdžiui, MS Excel.

Pakeitimų registre apie bet kurį pakeitimą turėtų būti saugomi bent jau šie jo atributai:

- identifikatorius,
- pateikties data,
- kas pateikė;
- dėl kokių priežasčių daromas;
- kokią naudą iš to gaus užsakovas ar kuri nors kita suinteresuotoji šalis,
- statusas (pateiktas, analizuojamas, vertinamas, atmestas, aprobuotas, įgyvendinamas, įgyvendintas),
- pasekmės projektui;
- įgyvendinimo būdas.

Technologijos, padedančios tvarkytis su reikalavimų pokyčiais, yra palyginti paprastos. Visų pirma reikia numatyti griežtai apibrėžtą interfeisą reikalavimams daryti pakeitimus pateikti. Antra, visos suinteresuotosios šalys turi turėti prieigą prie bazinio reikalavimų komplekto (angl. *requirements baseline*) ir galėti pasižiūrėti, kaip atrodo paskutinė aprobuotų reikalavimų versija. Trečia, reikia dirbti pagal tokį programų sistemos gyvavimo ciklo modelį, kuris yra pritaikytas dirbti su nuolat keičiamais reikalavimais. Tai galėtų būti, pavyzdžiui, evoliucinis gyvavimo ciklo modelis. Be to dar reikia padaryti du papildomus dalykus:

- įtikinti organizacijos vadovybę, kad pokyčius valdyti tikrai reikia;
- numatyti, kaip visas suinteresuotąsias šalis priversti naudotis nustatytomis pokyčių valdymo procedūromis.

Dažnai nėra paprasta padaryti nei vieną, nei kitą. Dažnai pokyčių sistemos diegimui priešinasi ne tik vadovybė, bet ir programuotojai, nes jiems atrodo, kad tiesiogiai viską išsiaiškinantis su užsakovais yra daug greičiau ir kad pokyčių valdymo sistema yra tik niekam nereikalingas biurokratinis balastas. Be to, pokyčių valdymo diegimas nėra pigus. Parengti pokyčių valdymo taisykles ir jas įdiegti dažniausiai prireikia kelių mėnesių. Po to kažkas turi skirti savo laiką pokyčiams registruoti, analizuoti ir vertinti. Visa tai kainuoja pinigus. Kita vertus, bent šiek tiek didesniuose projektuose be pokyčių valdymo apseiti negalima. Tai išsilieja į daug didesnius finansinius nuostolius, nekalbant jau apie tai, kad žlunga projekto vykdymo terminai, visiems tenka dirbti viršvalandžius, o dažniausiai dar ir susilaukiamą užsakovo nepasitenkinimo.

8.7 Reikalavimų trasavimas

IEEE standartas, aprašantis rekomenduojamas reikalavimų inžinerijos praktikas, sako, kad "programų sistemos reikalavimų specifikacija yra trasuojama, jei: (i) yra žinomos kiekvieno reikalavimo ištakos ir (ii) į kiekvieną reikalavimą galima padaryti nuorodą būsimuose projektavimo bei sistemos plėtotei ar tobulinimui

skirtuose dokumentuose" [ST3]. Pagrindinė reikalavimų trasavimo paskirtis - padėti atsekti bet kurio reikalavimo istoriją. Istorija gali būti peržvelgiamą abiem kryptim, tiek atgal, atsekant iš kokių aukštesniųjų lygmenų reikalavimų analizuojamasis reikalavimas išsirutuliojo, tiek ir pirmyn, atsekant kokius žemesniųjų lygmenų reikalavimus jis pagimdė ar paveikė. Be to, kartais yra svarbūs ir prieistoriniai bei postistoriniai reikalavimo gyvavimo aspektai. Kitaip tariant, yra svarbu, kas dėjosi su reikalavimu iki įjungiant jį į reikalavimų specifikaciją ir kaip jį paveikė tas faktas, kad pagaliau jis buvo įjungtas į specifikaciją.

Iš pirmo žvilgsnio šios problemos atrodo labai paprastos, gryna techninio pobūdžio. Iš tiesų taip nėra. Jos iki šiol nėra galutinai išspręstos [38].

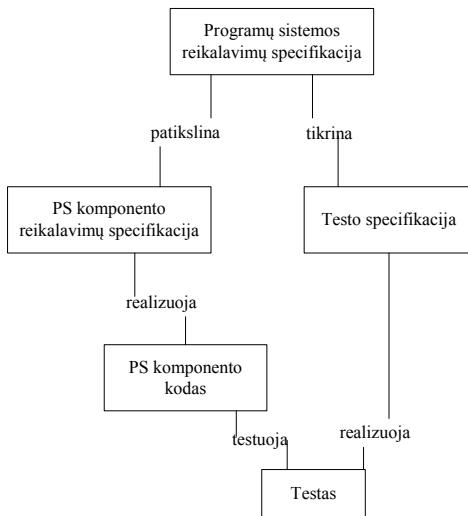
Reikalavimams trasuoti naudojama daug metodų:

- įvairios kryžminiu nuorodų sistemos [28],
- raktažodžių priklausomybės [50],
- įvairūs šablonai [105],
- trasavimo matricos [23], [111],
- matricų sekos [9],
- hipertekstinės priklausomybės [59],
- integruiantys dokumentai [71],
- prielaidomis grindžiami teisingumo palaikymo tinklai [104],
- ribojimų tinklai [7].

Šie metodai vienas nuo kito skiriasi trasomis siejamos informacijos apimtimi bei įvairove, kontroliuojamų sąsajų skaičiumi ir tuo, kiek jie padeda pertvarkyti reikalavimus, juos keičiant. Ne visi reikalavimų trasavimo metodai yra universalūs. Kai kurie iš jų yra susieti su kokia nors konkrečia reikalavimų specifikavimo kalba, pavyzdžiui, RSL [24], su kokia nors konkrečia reikalavimų modeliavimo kalba, pavyzdžiui, UML [113], arba su kokia nors konkrečia reikalavimų formulavimo metodika, pavyzdžiui, [76] ar [116]. Specializuotais metodais galima pasinaudoti tik tuomet, kuomet naudojamas su juo suderintas reikalavimų inžinerijos procesas ir tą metodą palaikančios specializuotos programinės priemonės.

Trumpai aptarsime svarbiausius reikalavimų trasavimo metodus.

Kryžminiu nuorodų sistemos. Jei reikalavimus siejančios trasos konstruojamos naudojant kryžmines nuorodas, tai kiekviename reikalavime daroma nuoroda į aukštesnio lygmens reikalavimus arba į ankščiau suformuluotus to pačio lygmens reikalavimus, iš kurių tas reikalavimas yra gautas. Paprasčiausiu atveju nuorodą sudaro dokumento pavadinimas ir atitinkamos to dokumento pastraipos pavadinimas arba numeris. Turint tokias nuorodas galima judėti tiktais viena kryptimi – atgal. Šiek tiek sudėtingesnės nuorodos yra vadinamosios *raktinės frazės*. Jos taip pat leidžia judėti tik viena kryptimi ir yra nukreiptos atgal, bet tokia nuoroda dar yra papildyta vadinamuoju *raktiniu žodžiu*, aprašančiu kokiu ryšiu reikalavimas yra susietas su reikalavimais, į kuriuos iš jo daromas nuorodos, pavyzdžiui, „patikslina“, „realizuoj“ „tikrina“ ar kt. (123 pav.). Be to, raktinėje frazėje, jeigu to reikia, galima nurodyti dokumento, į kurį daroma nuoroda, versiją bei kitą papildomą informaciją. Kadangi abiem aptartais atvejais nuorodos yra nukreiptos atgal, aukščiausiojo lygmens reikalavimai neturi jokių nuorodų.



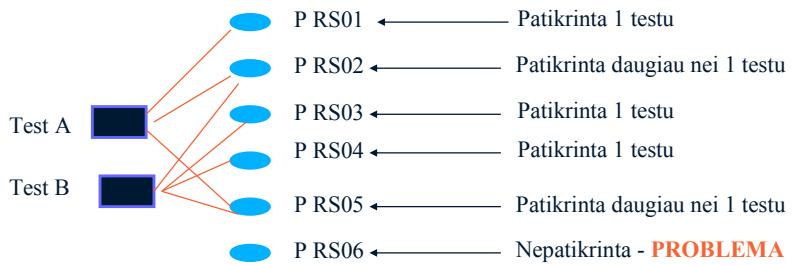
123 pav. Raktinių žodžių panaudojimo pavyzdžiai

Raktines frazes galima panaudoti ir atvirkštinėms nuorodoms, t. y. pirmyn nukreiptoms nuorodoms aprašyti. Tada nuorodos daromos ir iš aukštesnio lygmens reikalavimų į žemesnio lygmens reikalavimus bei iš anksčiau suformuluotų reikalavimų į jų vėlesnes versijas. Tačiau tai reikalauja papildomo darbo, nes rašant į repozitorijų kokius nors naujus reikalavimus, nuoradas reikia įterpti ne tik tuos reikalavimus, bet ir iš anksčiau į repozitorijų įrašytus reikalavimus. Pirmyn nukreiptose nuorodose raktinės frazės aprašo, ką reikia padaryti įgyvendinant reikalavimą, iš kurio yra daromos nuorodos. Tam naudojami atitinkami raktiniai žodžiai. Žemiausiojo lygmens reikalavimai pirmyn nukreiptų nuorodų neturi.

Naudojant abiem kryptim nukreiptas nuorodas, gauname kryžminiu nuorodų sistemą. Bendruoju atveju kryžminės nuorodos susieja kiekvieną kokioje nors reikalavimų specifikacijoje pateiktą reikalavimą tiek su aukštesnio lygmens (ankstesniais), tiek ir su žemesnio lygmens (vėlesniais) reikalavimais ar kitokio pobūdžio informaciniiais įrašais. Bet kuriame repozitorijaus įraše gali būti bet kuris skaičius tiek į vieną, tiek ir į kitą pusę nukreiptų nuorodų ir tos nuorodos gali būti daromos į įrašus paimtus iš skirtingu dokumentu.

Dažniausiai repozitorijuje yra saugomi informacinių įrašai, iš kurių automatiškai yra generuojami visi reikiams dokumentai. Bet gali būti ir kitaip, repozitorijuje gali būti saugomi patys rankiniu būdu parengti dokumentai. Tada raktinės frazės įterpiamos į atitinkamas tų dokumentų vietas, o trasavimą palaikančios programinės priemonės radusios peržiūrimame dokumente raktinę frazę suranda ir pateikia to paties ar kito dokumento tekštą, į kurį yra daroma nuoroda raktinėje frazėje. Aišku, leidžiami tik tokie raktiniai žodžiai, kuriuos apdoroja trasavimo programa. Be to, kad programa galėtų rasti nuorodas, jos turi būti pažymėtos tekste specialiu būdu. Paprastai pakanka sudėti tik atgal nukreiptas nuorodas, o pirmyn nukreiptas nuorodas formuoja pati programa. Šitaip sumažinama trasoms konstruoti reikalingo darbo apimtis.

Kryžminės nuorodos gali būti saugomos ir atskirai nuo pačių dokumentų, pavyzdžiui specialiai tam skirtoje duomenų bazės lentelėje. Šitaip pagreitinamas vaikščiojimas trasomis.



124 pav. Tikrinimas, kokių mastu testavimo planas tikrina reikalavimų įgyvendinimą.

Dauguma trasavimo programų tiktai suranda ir pateikia tekstus, paimtus iš skirtinės lygmenės reikalavimų specifikacijų arba iš skirtinės tos pačios specifikacijos versijų. Kaip parodyta 123 pav., tekstai gali būti paimti ir iš projektinės dokumentacijos, testavimo planų, testų, ataskaitų apie testavimą ir netgi iš kokia nors programavimo kalba parašytų programų. Tačiau yra ir sudėtingesnių reikalavimų trasavimo programų, leidžiančių patikrinti, ar tikrai buvo panaudoti visi reikalavimai ir ar jų sistemą nebuvo įvesta kokio nors perteklinio funkcionalumo. Tokios programos taipogi leidžia pamatyti, kokius kitus reikalavimus ir kokius projektavimo sprendimus paveiks daromi to ar kito reikalavimo pakeitimai. Jei reikalavimai trasomis yra susieti su baigiamuoju testavimo planu, tai galima sužinoti, kokių mastu tas planas patikrina programų sistemos reikalavimų įgyvendinimą (124 pav.).

Pavyzdys

Tegul programų sistemos P reikalavimų specifikacijoje turime tokį tekstą:

<<P RS: 001>> Sistemoje turi būti numatytos priemonės prieigai prie kitų sistemų (Navision, RP/3, etc.) sukurtų failų”.

Tekstas programų sistemos komponento F reikalavimų specifikacijoje yra toks:

*<<F RS:001; **detalizuoj**; P RS: 001>>. Vartotojas turi turėti galimybę nurodyti išorinio failo tipą.*

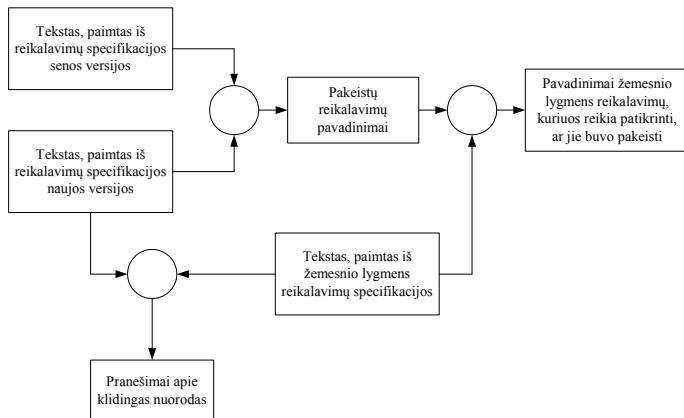
*<<F RS:002; **detalizuoj**; P RS: 001>>. Su kiekvieno tipo failu turi būti susieta jam apdoroti skirta priemonė.*

*<<F RS:003; **detalizuoj**; P RS: 001>> Kiekvienas leistinas tipas ekrane turi būti pavaizduotas specialia pikograma.*

*<<F RS:004; **detalizuoj**; P RS: 001>> Vartotojas turi turėti galimybę apibrėžti naujus failų tipus ir su jais susieti naujas pikogramas.*

*<<F RS:005; **detalizuoj**; P RS: 001>> Vartotojui parinkus failą vaizduojančią pikogramą ir nurodžius failo pavadinimą, turi būti iškviesta tam failo tipui apdoroti skirta priemonė ir jai kaip parametras turi būti perduotas apdorojamo failo pavadinimas.*





125 pav. Viena iš galimų automatinio lyginimo schemų

Trasavimo programos pateikti tekstai gali būti lyginami rankiniu būdu, bet yra sukurta ir specialiai tam skirtų programų. Tekstai gali būti lyginami skirtingai. Vienas būdas parodytas 125 pav.

Integruojantys dokumentai. Pagrindinė šio metodo idėja yra konstruoti reikalavimų trasas, panaudojant vadinamuosius integruojančiuosius dokumentus. Metodas buvo sukurtas IPSE projekte [71] ir yra pritaikytas kompiuteriniam trasavimui, atliekamam panaudojant atitinkamas instrumentines sistemas. Galima tarpusavyje susieti bet kokius dokumentus, išskaitant skirtingų lygmenų reikalavimų specifikacijas, projektinius dokumentus ir netgi programų tekstu. Nereikalaujama, kad tarpusavyje būtų siejami tik du dokumentai. Tieki aukštesniojo lygmens dokumentų skaičius, tiek ir žemesniojo lygmens dokumentų skaičius nėra ribojami. Aukštesnio lygmens dokumento teksto fragmentai susiejami su tais žemesniojo lygmens teksto fragmentais, kurie keičiasi, padarius pakeitimus atitinkamuose aukštesniojo lygmens teksto fragmentuose. Fragmentai siejami vadinamuoju *tarpdokumentiniu ryšiu*. Bendruoju atveju tarpdokumentinio ryšio kardinalumas yra m:n. Integruojantysis dokumentas aprašo visus apibrėžtus tarpdokumentinius ryšius. Greta jo dar yra užduodama vadinamoji *transformavimo lentelė*, aprašanti taisykles, kaip transformuoti aukštesniojo lygmens dokumentų teksto fragmentus į žemesniojo lygmens dokumentų teksto fragmentus. Kadangi kartais taisykles gali būti nepakankamai tiksliai suformuluotos ir interpretuoamos nevienareikšmiai, numatyta galimybė užklausti žmogų, kurią interpretaciją reikia pasirinkti konkrečiu atveju. Žmogaus duoti nurodymai yra išrašomi į integravantį dokumentą ir susiejami su atitinkamu tarpdokumentiniu ryšiu.

Trasavimo matricos. Tai pats paprasčiausias ir pats populiausias reikalavimų trasavimo metodas. Naudojant ši metodą, konstruojamos dviejų rūšių matricos: *reikalavimų lokalizavimo matrica* (126 pav.) ir *reikalavimų ryšio matrica* (127 pav.). Reikalavimų lokalizavimo matrica parodo, kokių sistemų komponentuose koks reikalavimas yra lokalizuotas, o reikalavimų ryšių matrica – koks reikalavimas iš kokių kitų reikalavimų yra gautas.

Reikalavimai	1 posistemis	2 posistemis	IRS
P RS01	X		1 IRS
P RS02		X	
P RS03	X		
P RS 04		X	1 IRS
.....

126 pav. Reikalavimų lokalizavimo matricos fragmento pavyzdys.

Reikalavimų lokalizavimo matricos pavyzdje (126 pav.) parodyta, kaip lokalizuoti (t. y. susieti su komponentais) sistemos P reikalavimų specifikacijoje surašyti reikalavimai, dekomponavus tą sistemą 107 pav. parodytu būdu. Priminsime, kad 1 IRS žymi pirmo sistemos hierarchijos lygmens komponentų interfeisių specifikaciją. Taigi reikalavimas P RS01 taikomas pirmajam sistemos posistemiui ir posistemių interfeisių specifikacijai, P RS02 – antrajam posistemiui, P RS03 – pirmajam posistemiui ir P RS 04 – antrajam posistemiui ir posistemių interfeisių specifikacijai.

Reikalavimas	Iš kokių reikalavimų jis gautas
.....
1 PS F001	P RS01
1 PS F002	P RS01
1 PS F003	P RS01
1 PS F004	P RS01
1 PS F005	P RS01
.....

127 pav. Reikalavimų ryšių matricos fragmento pavyzdys.

Reikalavimų ryšių matricos pavyzdje (127 pav.) parodyta kaip susiejami reikalavimai, atlikus sistemos P reikalavimo P RS01 nuleidimą į pirmą posistemį 109 pav. parodytu būdu. Raidė F čia žymi reikalavimo tipą (funkcinis). Kaip matome, ši matrica užduoda tik vieno tipo nuorodas – nuorodas atgal. Kaip buvo kalbėta 5.14 poskyryje, šios matricos gali būti sujungtos ir į vieną matricą (112 pav.). Tačiau bet kuriuo atveju, reikalavimų trasavimo programa, naudodamas informaciją apie reikalavimų ryšius, gali automatiškai suformuoti dar vieną matricą (128 pav.), aprašančią nuorodas pirmyn.

Reikalavimas	Iš kokių reikalavimų jis gautas
P RS01	1 PS F001, 1 PS F002, 1 PS F003, 1 PS F004, 1 PS F005, 1IRS 001
.....

128 pav. Transformuotos reikalavimų ryšių matricos fragmento pavyzdys.

Akivaizdu, kad reikalavimų ryšių matricoje galima daryti nuorodas ir į kitų dokumentų (projektinės dokumentacijos, programos teksto, testavimo plano, vartotojo vadovo ir kt.) įvardintas struktūrines dalis.

8.8 Instrumentinės sistemos

Esamos instrumentinės sistemos kol kas nepajėgia visiškai automatizuoti reikalavimų tvarkymo darbų. Jos tuos darbus tik supaprastina ir palengvina. Savaime aišku, kad naudojant specializuotas instrumentines sistemas reikalavimus tvarkyti galima daug efektyviau, negu naudojantis tik standartinėmis tekstų redagavimo sistemomis, kompiuterinėmis lentelėmis ar reliacinėmis DBVS.

Didžioji dauguma reikalavimų inžinerijai pritaikytų šiuolaikinių instrumentinių sistemų gali:

- kurti ir palaikyti reikalavimų saugyklas (repozitorijus);
- atlikti paiešką tose saugyklose;

- kurti bazinius reikalavimų komplektus ir valdyti reikalavimų pokyčius;
- trasuoti reikalavimus;
- generuoti įvairius reikiamus dokumentus bei ataskaitas.

Sugeneruotas specifikacijas ir ataskaitas paprastai galima eksportuoti į standartines tekstų redagavimo ar kompiuterinių lentelių sistemas.

Duomenys apie šiuo metu populiausias instrumentines sistemas pateikti 14 lentelėje.

14 lentelė. Duomenys apie populiausias reikalavimams tvarkyti skirtas instrumentines sistemas.

Pavadinimas	Tiekėjas	Apytikslė kaina litais	Kur galima susipažinti plačiau
Borland Calibre Analyst		25 000	http://www.borland.com/
DOORS		25 000	www.telelogic.com
DOORS XT			www.telelogic.com
DOORS/Analyst		2 500	www.telelogic.com
DOORS/Net		50 000 (10 darbo vietų)	www.telelogic.com
Rational Requite Pro		10 000	www.rational.com
Requirements Analyst		2 500	www.analysttool.com
Team Trace		2 500	www.teamtrace.com
RTM		25 000	www.chipware.com

Trumpai aptarkime porą iš šių sistemų.

Borland Calibre Analyst (Calibre). Ši sistema gali aptarnauti ne tik vieną konkretų projektą, bet ir visą programinę įrangą kuriančią organizaciją. Kitaip tariant, šios sistemos saugykloje galima saugoti visų organizacijoje vykdomų projektų reikalavimus ir juos efektyviai tvarkyti. Organizacija gali būti didelė ar maža, centralizuota ar išskirstyta. Reikalavimai saugomi centralizuotame sistemos repozitoriuje, juos galima susieti trasomis su projektiniais dokumentais, programų tekstais, testais ir bet kuria kita informacija. Idomu, kad į reikalavimus leidžiama įterpti paveikslėlius. Tai ypač naudinga tuomet, kuomet aiškinantis reikalavimus naudojamas vadinamaja *vaizdžių paveikslėlių* (angl. *rich picture*) metodika. Prie repozitorijaus prieinama per internetą. Organizacija gali naudoti bet kurį reikalavimų inžinerijos procesą ir pritaikyti sistemą prie jo. Tai daroma dialogo režimu, kokios nors procesui aprašyti skirtos kalbos sistemoje nėra.

Sistema palengvina užsakovų ir vykdytojų bendradarbiavimą, padeda formuluoti reikalavimus, juos keisti ir vertinti tokijų pakeitimų pasekmes. Užsakovų ir vykdytojų bendravimui palengvinti ir reikalavimams formuluoti sistemoje reikia sukurti dalykinės srities terminų žodyną. Sistemoje sukurtais žodynais galima naudotis dialogo režimu ir panaudoti juos reikalavimams detalizuoti. Sistema padeda vertinti reikalavimų įgyvendinamumą ir geriau planuoti projektą. Sistema pati informuoja visas suinteresuotas šalis apie daromus reikalavimų pakeitimus ir padeda analizuoti ir vertinti bet kurioje projekto stadioje padarytų pakeitimų pasekmes. Tai daroma panaudojant įvairius trasavimo metodus ir vizualizuojant trasas. Sistema gali pateikti visų projekto eigoje darytų pakeitimų istoriją. Ji kuria ir palaiko bazinius reikalavimų komplektus, prioreituojančius reikalavimus pagal kainą, terminus ar kitus nurodytus kriterijus ir pateikia rezultatus kompiuterinių lentelių forma. Kaip ir kitos panašaus pobūdžio sistemos, Calbre gali automatiškai generuoti reikalavimų

specifikacijas, įvairias ataskaitas ir kitus dokumentus. Ji turi lengvai suvokiamą ir patogų grafinį interfeisą.

DOORS produktų šeima. Kaip parodyta 14 lentelėje, yra visa DOORS produktų šeima: sistema DOORS skirta tvarkyti reikalavimus išskirstytuose projektuose, t. y. tokiuose projektuose, kuriuose dalyvauja kelios skirtingose vietose dirbančios grupės; DOORS XT skirta tvarkyti daugelio projektų reikalavimus, t. y. tvarkyti reikalavimus visos organizacijos mastu; DOORS/net leidžia tvarkyti reikalavimus per Internetą ir DOORS/Analyst yra skirta modeliuoti reikalavimus UML kalba.

Priemones reikalavimams tvarkyti turi ne tikai specialiai tam skirtos instrumentinės sistemos, bet ir daugelis integruotųjų viso programų sistemų kūrimo proceso palaikymo sistemų. Duomenys apie šiuo metu populiarusias tokio tipo instrumentines sistemas pateikti 15 lentelėje¹⁷.

15 lentelė. Duomenys apie populiarusias programų sistemų inžinerijos sistemas, palaikančias ir reikalavimų tvarkymą.

Pavadinimas	Tiekėjas	Apytikslė kaina litais	Kur galima susipažinti plačiau
Focal Point		25 000	www.focalpoint.se
Speedev, supaprastinta versija		25 000 10 000	www.speedev.com
RDD supaprastinta versija		50 000 25 000	www.holagent.com
CORE supaprastinta versija		50 000 25 000	www.vtcorp.com
Cradle supaprastinta versija		50 000 25 000	www.threesl.com

Taip kaip ir sistemoje Calibre, daugumoje instrumentinių sistemų galima pačiam aprašyti naudojamą reikalavimų inžinerijos procesą ir šitaip pritaikyti sistemą prie pasirinkto proceso. Aprašant reikalavimų inžinerijos procesą yra apibrėžiami reikalavimų atributai, aprašoma repozitorijaus struktūra struktūrą, sukuriami repozitorijaus rodiniai, pritaikyti skirtingoms projekto dalyvių grupėms, aprašomi sistemos generuojamų dokumentų šablonai, apibrėžiami kontroliniai taškai ir daromi kiti panašaus pobūdžio aprašai.

¹⁷ Instrumentinių sistemų apžvalgas galima rasti tarptautinės sistemų inžinierių sąjungos INCOSE tinklalapyje (paper-review.com/tools/rms/read.php), VOLERE tinklalapyje (volere.co.uk/tools.htm) ir Ian Alexander tinklalapyje (<http://easyweb.easynet.co>).

Literatūra

1. Dennis M. Ahern, Aaron Clouse, Richrd Turner. *CMMI Distilled*. Addison-Wesley, Reading, Mass., 2001.
2. *A Primer on Requirements Engineering Introducing Beavers Requirements Engineering Services Capability*. White paper. Version 0.4, September 2003, Beaver Computer Consultants Ltd. (galima rasti adresu www.beaver-consulting.co.uk).
3. Ian F. Alexander, Richard Stevens. *Writing Better Requirements*. Addison-Wesley, 2002.
4. Stephen Barlas. FAA shifts focus to sealed-back DSR. *IEEE Software*, March 1996, p. 110.
5. Hugh Beyer, Karen Holtzblatt. *Contextual Design: Defining Customer-centered Systems*, Morgan Kaufmann, 1997, ISBN: 1558604111.
6. Barry W. Boehm, Ellis Horowitz, Ray Madachy, Donald Reifer, Bradford K. Clark, Bert Steece, A. Winsor Brown, Sunita Chulani, Chris Abts. *Software Cost Estimation with COCOMO II*. Prentice Hall, Upper Saddle River (NJ), 2000.
7. Jonathan Bowen, Peter O'Gdy, Larry Smith. A constraint programming language for life cycle engineering. *Artificial Intelligence in Engineering*, 1990, Vol. 5, No. 4, 206-220.
8. Ian K. Bray. *Introduction to Requirements Engineering*. Addison Wesley (an imprint of Pearson Education), 2002, ISBN 0201 767929.
9. Kevin Brennan. The Economics of Software Development. September 25, 2004 12:19 PM, *International Trade and Finance*.
10. Patrick G. Brown. QFD: echoing the voice of the customer. *AT&T Technical Journal*, 1991, March/April, 21-31.
11. Luigi Buglione, Alain Abran, *The Software Measurement Body of Knowledge*, Software Measurement European Forum, Rome, 2004.
12. *Certified Software Development Professional Program*. Web site. IEEE Inc., 2006, (galima rasti adresu <http://computer.org/certification>).
13. Peter P. Chen. The entity-relationship model. Towards unified view of data. *ACM Transactions on Data Base Systems*, Vol. 1, No. 1, 1976, 9-36.
14. Lawrence Chung, Brian A. Nixon, Eric Yu, John Mylopoulos. *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers, 2000.
15. CMMI Version. 1.1. Pittsburgh, PA: Software Engineering Institute, CMMI Product Team, March 2002.
16. F. Coallier. TRILLIUM: A model for the assessment of telecom product development and support capability. In R. B. Hunter, R. H. Thayer (eds.). *Software Process Improvement*, IEEE Computer Society Press, Los Alamitos, Calif., 1999.
17. Alistair Cockburn. *Writing Effective Use Cases*. Addison-Wesley Professional; 2000, ISBN: 0201702258.
18. David A. Cook, Leslie Dupaix, Larry Smith. *A Gentle Introduction to Software Engineering*. Rev. 3.0. Hill Air Force Base, UT: Software Technology Support Center, 31 Mar. 1999.
19. Anthony Coppola. Quality Function Deployment. *Start*, vol. 4, no. 1, 1-4.
20. Albertas Čaplinskas. Programų sistemų inžinerijos pagrindai. Matematikos ir informatikos institutas, Vilnius, I dalis, 1996, II dalis, 1998.
21. Anne Dardenne, Axel van Lamsweerde, Stephen Fickas. Goaldirected requirements acquisition. *Science of Computer Programming*, 20, 1993, 3-50.

22. Gordon B. Davis, Margrethe H. Olson. *Management Information Systems: Conceptual Foundations, Structure and Development*. 2nd edition, McGraw-Hill, 1985.
23. Alan Davis. *Software Requirements: Objects, Functions and States*. Prentice Hall PTR; 2nd edition, 1993, ISBN: 013805763X.
24. Carl G. Davis, Charles R. Vick. The software development system, *IEEE Transactions on Software Engineering*, 1977, Vol. 3, No. 1, 69-84.
25. Tom DeMarco. *Structured Analysis and System Specification*. Prentice Hall PTR, 1979.
26. Merlin Dorfman. System and software requirements engineering. Richard H. Thayer and Merlin Dorfman (eds.). *System and Software Requirements Engineering*. IEEE CS Press, Los Alamitos, Calif., 1990, 4-16.
27. 24. Khaled El Emam, Jean-Normand, Drouin, Walcélío Melo. *SPICE: The Theory and Practice of Software Process Improvement and Capability Determination*. IEEE Computer Society Press, Los Alamitos, Calif., 1997.
28. Michael W. Evans. *The Software Factory*. John Wiley and Sons, 1989.
29. Michael Fagan. Design and Code Inspections. *IBM System Journal*, vol. 15, no.3, 1976, 182-211.
30. Kazi Farooqui, Luigi Logrippo, Jan de Meer. The ISO reference model for open distributed processing: an introduction. *Computer Networks and ISDN Systems*, Vol. 27, No. 8, 1995, 1215-1229.
31. Patricia L. Ferdinandi. *A Requirements Pattern: Succeeding in the Internet Economy*. Addison-Wesley Professional; 2001.
32. Anthony Finkelstein. Requirements Engineering: a review and research agenda In: Proc 1st Asian & Pacific Software Engineering Conference, IEEE Computer Society Press, 1994, 10-19 (galima rasti adresu <http://www.cs.ucl.ac.uk/staff/A.Finkelstein/papers/rereview.pdf>).
33. Donald Firesmith. Creating a project-specific requirements engineering process. *Journal of Object Technology*, Vol. 3, No. 5, 2004, 31-44 (galima rasti adresu http://www.jot.fm/issues/issue_2004_05/column4).
34. *Function Point Counting Practices Manual*. IFPUG, June 2002 (galima rasti adresu <http://www.ifpug.org/publications/manual.htm>).
35. Donald C. Gause, Gerald M. Weinberg. *Exploring Requirements: Quality before Design*. Dorset House Publishing Company, Inc., 1989, ISBN: 0932633137.
36. Joseph A. Goguen, Charlotte Linde. Techniques for requirements elicitation. *International Symposium on Requirements Engineering*, 1993.
37. Robin F. Goldsmith. *Discovering Real Business Requirements for Software Project Success*, Artech House Publishers, 2004, ISBN: 1580537707.
38. Orelana C.Z. Gotel, Anthony C.W. Finkelstein. *An Analysis of the Requirements Traceability Problem*. Technical Report TR-93-41. 1993, Department of computing, Imperial College (galima rasti adresu http://csis.pace.edu/~ogotel/papers/RT_PAP.pdf).
39. Ellen Gottesdiener. *Requirements by Collaboration: Workshops for Defining Needs*. Addison-Wesley Professional, 2002, ISBN: 0201786060.
40. Ian Graham. *Requirements Engineering and Rapid Development, an object-oriented approach*. Addison-Wesley 1998, ISBN: 0 201 36047 0.
41. Jennifer Gremba, Chuck Myers. The IDEAL model: a practical guide for improvement, *Bridge*, Software Engineering Institute, Issue 3, 1997 (taip pat yra paskelbta adresu <http://www.sei.cmu.edu/ideal/ideal.bridge.html>).

42. *Guide to the SE Body of Knowledge – G2SEBoK*. INCOSE, 2001-2004 (galima rasti adresu <http://g2sebok.incose.org/>).
43. Mike Hammer, James Champy. *Reengineering the Corporation: A Manifesto for Business Revolution*. Harper Business Books, 1994.
44. David Harel. Statecharts: a visual formalism for complex systems. *Sience of Computer Programming*, 8, 1987, 231-274.
45. Paul Harmon. *Business Process Change: A Manager's Guide to Improving, Redesigning, and Automating Processes*. Morgan Kaufmann, 2003, ISBN 1558607587.
46. Derek Hatley, Peter Hruschka and Imtiaz Pirbha. *Process for System Architecture and Requirements Engineering*. Dorset House Publishing Company, Inc., 2000), ISBN: 0932633412.
47. David C. Hay. *Requirements Analysis. From Business Views to Architecture*. Pearson Education (publishing as Prentice Hall PTR), 2003, ISBN 0-13-028228-6.
48. Ivy Hooks, Kristen A. Farry. *Customer-Centered Products: Creating Successful Products through Smart Requirements Management*. AMACOM, 2000.
49. Elizabeth Hull, Kenneth Jackson, Jeremy Dick. *Requirements Engineering*. Springer; 2004, ISBN: 1852338792.
50. ISEBOK – Information Systems Engineering BoK (galima rasti adresu <http://osiris.cs.kun.nl/pubmngr/Data/2000/Proper/I%20SEBOK/2000-ProperISEBOK.refs.html>).
51. Justin Jackson. A key phrase based traceability scheme. In *Tools and Techniques for Maintaining Traceability During Design*, IEE Colloquium, Computing and Control Division, Professional Group C1, Digest No. 1991/180, IEE, 1991, 2/1-214.
52. Michael Jackson. *Software Requirements and Specifications. A Lexicon of Practice, Principles and Prejudices*. ACM Press and Addison-Wesley. 1995
53. Michael Jackson, Pamela Zave. Deriving specifications from requirements: An example. In *Proceedings of the 17th International Conference on Software Engineering*, ACM, 1995, 15-24, ISBN 0897917081.
54. Deb Jacobs. Requirements Engineering So Things Don't Get Ugly. *CrossTalk*, October 2004, 1-10 (galima rasti adresu <http://www.stsc.hill.af.mil/CrossTalk/2004/10/0410Jacobs.html>).
55. Ivari Jacobson, Magnus Christerson, Patrik Jonsson, Gunnar Övergaard. *Object-Oriented Software Engineering: A Use Case Driven Approach*. Addison-Wesley, 1992.
56. Ivar Jacobson, Maria Ericson, Agneta Jacobson. *The Object Advantage: Business Process Re-engineering with Object Technology*. Addison-Wesley Professional, 1994, ISBN 0201422891.
57. Pankaj Jalote. *An Integrated Approach to Software Engineering*. Springer, ISBN: 0-387-20881-X.
58. Ron Jeffries, Ann Anderson, and Chet Hendrickson. *Extreme Programming Installed*. Addison-Wesley, Boston, 2001.
59. Hermann Kaindl. The missing link in requirements engineering, *ACM SIGSOIT Software Engineering Notes*, 1993, Vol. 18, No. 2, 30-39.
60. Noriaki Kano, Nobuhiko Seraku, Fumio Takahashi, Shinichi Tsuji. Attractive Quality and Must-Be Quality (japonų kalba). *Hinshitsu*, Vol. 14, No. 2, 1984, 39-48.
61. Joachim Karlsson. Managing software requirements using quality function deployment. *Software Quality Journal*, 6, 1997, 311–325.

62. Joachim Karlsson. Software requirements prioritizing. In *Proceedings of the 2nd IEEE International Conference on Requirements Engineering, April 15–18, 1996, Colorado Springs*. 1996, 110–116 (galima rasti adresu <http://www.requirements-engineering.org>).
63. Gerald Kotonya, Ian Sommerville. *Requirements Engineering: Processes and Techniques*, John Wiley & Sons, 2000, ISBN 0471972088.
64. Benjamin Kovitz. *Practical Software Requirements: a Manual of Content & Style*. Manning Publications, 1998, ISBN: 1884777597.
65. Philippe Kruchten. *The Rational Unified Process: An Introduction*. 3rd ed., Addison-Wesley Professional, 2003.
66. Daryl Kulak, Eamonn Guiney. *Use Cases: Requirements in Context*. Addison-Wesley Professional; 2004, ISBN: 0321154983.
67. Pasi Kuvaja, Jouni Similä, Lech Krzanik, Adrian Bicego, Samuli Saukkonen, Günter Koch. *Software Process Assessment and Improvement: The BOOTSTRAP Approach*. Blackwell Publishers, Oxford, 1994.
68. Axel van Lamsweerde. *Goal-Oriented Requirements Engineering: From System Objectives to UML Models to Precise Software Specifications*. Wiley, 2005.
69. Soren Lauesen. *Software Requirements: Styles and Techniques*. Addison-Wesley; 2002, ISBN: 0201745704.
70. Dean Leffingwell, Don Widrig. *Managing Software Requirements: A Use Case Approach*, 2nd edition. Addison-Wesley, 2003.
71. Martin Leferring. An incremental integration tool between requirements engineering and programming in the large. *Proceedings of the IEEE International Symposium on Requirements Engineering. San Diego, California, Jan. 4-6*, IEEE Press, 1993, 82-89.
72. Timothy C. Lethbridge and Robert Laganière. *Object-Oriented Software Engineering: Practical Software Development using UML and Java*. McGraw Hill, 2001.
73. Dennis Linscomb. Requirements Engineering Maturity in the CMMI. *CrossTalk*, 2003, 12 (galima rasti adresu <http://www.stsc.hill.af.mil/Crosstalk/2003/12/0312linscomb.html>).
74. Pericles Loucopulos, Vassilios Karakostas. *Systems Requirements Engineering*. McGraw-Hill, 1995.
75. Linda A. MacAulay. *Requirements Engineering*. Springer-Verlag, 1996.
76. Robert G. Mays, Leonard S. Orzech, William A. Ciarella, Richard W. Phillips. PDM: a requirements methodology for software system enhancements. *IBM Systems Journal*, 1985, Vol. 24, No. 2. 134-149.
77. Glenn H. Mazur. Customer Encounters of the QFD Kinds. *Proceedings of the 6th Annual Quality Service Conference, September 16-17, 1997. Colorado Springs, 1997*, 1-20.
78. Doug W. McDavid. A standard for business architecture description. *IBM System Journal*, vol. 38, no. 1, 1999, 12-31.
79. Geoffrey A. Moore. *Crossing the Chasm: Marketing and Selling High-Tech Products to Mainstream Customers*. Harper Business, 1999, ISBN 0066620023.
80. Albert E. Motley III. *Quality Function Deployment (QFD)*. Old Dominion University, November 5, 2002.
81. Satoshi Nakui. Comprehensive QFD. In *The Transactions of the Third Symposium on QFD*, 1991.

82. Jerzy Nawrocki, Bartosz Walter, Adam Wojciechowski. Towards Maturity Model for eXtreme Programming, *Proceedings of the 27th EUROMICRO Conference*, IEEE Computer Society, Los Alamitos, 2001, 233-239.
83. Jerzy Nawrocki, Bartosz Walter, Adam Wojciechowski. Comparison of CMMI Level 2 and eXtreme Programming. In: J. Kontio, R. Conradi (eds.). *Software Quality – ECSQ 2002, Lecture Notes in Computer Science*, 2349, Springer, 288-297.
84. Jerzy Nawrocki, Michał Jasiński, Bartosz Walter, Adam Wojciechowski. Extreme Programming Modified: Embrace Requirements Engineering Practices. In: *Proceedings of the 10th IEEE Joint International Requirements Engineering Conference RE'02*, IEEE Press, Los Alamitos, 303-310.
85. Alex Faickney Osborn. *Applied imagination: Principles and procedures of creative problem solving*. Third Revised Edition. New York, NY: Charles Scribner's Sons, 1963.
86. Mark C. Paulk. Extreme Programming from a CMM perspective, *IEEE Software*, November/December 2001, 19-26.
87. Mark C. Paulk, B. Curtis, Mary Beth Chrissis, Charles V. Weber. *Capability Maturity Model for Software*. Version 1.1, Software Engineering Institute, CMU/SEI-93-TR-24, February 1993 (galima rasti adresu <http://www.sei.cmu.edu/cmmi/>).
88. Mark C. Paulk, Charles V. Weber, Suzanne M. Garcia, Mary Beth Chrissis, Marilyn Bush. *Key Practices of the Capability Maturity ModelSM*. Version 1.1, Software Engineering Institute, CMU/SEI-93-TR-025, ESC-TR-93-178, February 1993.
89. Mark C. Paulk, Michael D. Konrad. An overview of ISO's SPICE project. *IEEE Computer*, 1994, Vol. 27, No. 4, 68–70.
90. Shari Lawrence Pfleeger. *Software Engineering: Theory and Practice*. 2nd ed., Prentice Hall, 2001.
91. Howard Podeswa. *UML for the IT Business Analyst: A Practical Guide to Object-Oriented Requirements Gathering*, Course Technology PTR, 2005, ISBN: 1592009123.
92. *Proceedings of IEEE International Symposium on Requirements Engineering (RE'93)*, 1993, San Diego, CA, U.S.A, 1993 (galima rasti adresu <http://www.requirements-engineering.org>).
93. *QFD Awareness Seminar*. Quality Education & Training Center, Ford Motor Company, May 1989.
94. Samuel T. Redwine Jr. Creating a Software Assurance Body of Knowledge. *CrossTalk*, Oct 2005 Issue (galima rasti adresu <http://www.stsc.hill.af.mil/crosstalk/2005/10/0510Redwine.html>).
95. *Requirements Engineering Roundtable* (March 1999). Carnegie Mellon University, 2000 (galima rasti adresu <http://interactive.sei.cmu.edu/Features/1999/March/Roundtable/Roundtable.mar99.htm>).
96. Suzanne Robertson, James Robertson. *Mastering the Requirements Process*. Addison-Wesley Professional; 2006, ISBN: 0321419499.
97. Shelly Cashman Rosenblatt. Systems Analysis and Design 5th edition.
98. Thomas L. Saaty. *The Analytic Hierarchy Process*, McGraw-Hill, New York, 1980.

99. Pete Sawyer. Maturing Requirements Engineering Process Maturity Models. In: J. Maté, A. Silva (eds.). *Requirements engineering for sociotechnical systems*. IDEA Group Inc., 2004, 84-99.
100. Pete Sawyer, Stephen Viller, Ian Sommerville. Requirements process improvement through the phased introduction of good practice. *Software Process Journal*, 1998, Vol. 3, No. 1, 19–34.
101. Pete Sawyer, Ian Sommerville, Gerald Kotonya. Improving market-driven RE processes. In *Proceedings of International Conference on Product-Focused Software Process Improvement (ProFES '99), Oulu, Finlan*, 1999, 222–236.
102. Pete Sawyer, Ian Sommerville, Stephen Viller. Capturing the Benefits of Requirements Engineering. *IEEE Software*, March/ April 1999, 78-85.
103. Stephen R. Schach. *Object-Oriented and Classical Software Engineering*. 6th edition, McGraw-Hill Higher Education, 2005.
104. Tim Smithers, Ming-Xi Tang, Nils Tomes. The maintenance of design history in AI-based design. In *Tools and Techniques for Maintaining Traceability During Design, IEE Colloquium, Computing and Control Division*, Professional Group C1, Digest No. 1991/180, IEE, 1991, 8/1-8/3.
105. *Software through Pictures: Products and Services. Overview*. IDE, Inc., 1991
106. Ian Sommerville. *Software Engineering*. 7th ed., Addison-Wesley, 2005.
107. Ian Sommerville, Pere Sawyer. *Requirement Engineering. A good practice guide*. John Wiley&Sons, 1997, ISBN 0 471 97444 7.
108. Ian Sommerville, Jane Ransom. An Empirical Study of Industrial Requirements Engineering Process Assessment and Improvement. *ACM Transactions on Software Engineering and Methodology*, Vol. 14, No. 1, 2005, 85–117.
109. John Sowa, John A. Zachman. Extending and formalizing the framework for information systems architecture. *IBM Systems Journal*, 1992, Vol. 31, No.3, 590-616.
110. Lawrence P. Sullivan. *Quality Function Deployment*. Quality Press, 1986.
111. *SWEBOK®. Guide to the Software Engineering Body of Knowledge*. A project of the IEEE Computer Society Professional Practices Committee, 2004 Version. IEEE Computer Society, 2004. ISBN 0-7695-2330-7.
112. Richard H. Thayer and Merlin Dorfman (eds.). *System and Software Requirements Engineering*. IEEE CS Press, Los Alamitos, Calif., 1990.
113. *Unified Modeling Language: Superstructure*. Version 2.0, formal/05-07-04, Object Management Group, 2005.
114. Jim Van Buren, David A. Cook. Experiences in the adoption of requirements engineering technologies. *CROSSTALK*, December 1998, 3-10.
115. Debbie Walkowski. *Visio 2003 For Dummies*. Hungry Minds Inc, 2004, ISBN 0764559230.
116. Martin West. The use of quality function deployment in software development. In *Tools and Techniques for Maintaining Traceability During Design, IEE Colloquium, Computing and Control Division*, Professional Group C1, Digest No. 1991/180, IEE, 1991, 5/1-5/7.
117. Karl E. Wiegers. *Software Requirements*. 2nd ed., Microsoft Press, 2003, ISBN 0-7356-1879-8.
118. Bill Wiley. *Essential System Requirements: A Practical Guide to Event-Driven Methods*. Addison-Wesley, 1999.
119. Daniel R. Windle, Leonides Rene Abreo. *Software Requirements Using the Unified Process*. Prentice Hall PTR, 2002, ISBN 0130969729.

120. Ralph Rowland Young. *Effective Requirements Practices*. Addison-Wesley Professional, 2001, ISBN: 0201709120.
121. Ralph Rowland Young. *The Requirements Engineering Handbook*. Artech House Publishers, 2003, ISBN: 1580532667.
122. John Zachman. A framework for information systems architecture. *IBM Systems Journal*, Vol. 26, No. 3, 1987, 276-292.
123. Pamela Zave, Michael Jackson. Four dark corners of requirements engineering. *ACM Transactions on Software Engineering and Methodology*, Vol. 6, No. 1, 1997, 1-30.

Reikalavimų inžinerijos standartai

- ST1. ANSI/IEEE Std 610.12-1990. *IEEE Standard Glossary of Software Engineering Terminology*, IEEE, February 1991.
- ST2. IEEE Std 828-1998. *IEEE Standard for Software Configuration Management Plans*. IEEE, 1998.
- ST3. IEEE Std 830-1998. *IEEE Recommended Practice for Software Requirements Specifications*. IEEE, 1998 (galima rasti adresu <http://users.snip.net/~gbooker/INFO627/IEEE-830-1998.pdf>).
- ST4. IEEE Std 1028-1997 (R2002). *IEEE Standard for Software Reviews*. IEEE, 1997.
- ST5. IEEE Std 1058-1998. *IEEE Standard for Software Project Management Plans*. IEEE, 1998.
- ST6. IEEE Std 1061-1992. *IEEE Standard for Software Quality Metrics Methodology*. IEEE, 1992.
- ST7. IEEE Std 1062-1998. *IEEE Recommended Practice for Software Acquisition*. IEEE, 1998.
- ST8. IEEE Std 1074-1997. *IEEE Standard for Developing Software Life Cycle Processes*. IEEE, 1997.
- ST9. IEEE Std 1233-1998. *IEEE Guide for Developing System Requirements Specifications*. IEEE, 1998 (galima rasti adresu <http://ieeexplore.ieee.org/iel4/5982/16016/00741940.pdf?anumber=741940>).
- ST10. IEEE Std 1320.1-1998. *IEEE Standard for Functional Modeling Language-Syntax and Semantics for IDEF0*. IEEE, 1998.
- ST11. IEEE Std 1320.2-1998. *IEEE Standard for Conceptual Modeling Language-Syntax and Semantics for IDEFIX97 (IDEFObjetct)*. IEEE, 1998.
- ST12. IEEE Std 1362-1998. *IEEE Guide for Information Technology-System Definition-Concept of Operations (ConOps) Document*. IEEE, 1998.
- ST13. IEEE Std 1465-1998//ISO/IEC12119:1994. *IEEE Standard Adoption of International Standard ISO/IEC12119:1994(E), Information Technology-Software Packages-Quality requirements and testing*. IEEE, 1998.
- ST14. IEEE Std 1471-2000. *IEEE Recommended Practice for Architectural Descriptions Software Intensive Systems*. Architecture Working Group of the Software Engineering Standards Committee, 2000.
- ST15. IEEE/EIA 12207.0-1996//ISO/IEC12207:1995. *Industry Implementation of Int. Std. ISO/IEC 12207:95, Standard for Information Technology-Software Life Cycle Processes*. IEEE, 1996 (galima rasti adresu <http://www.abelia.com/docs/12207cpt.pdf>).
- ST16. IEEE Std 14143.1-2000//ISO/IEC14143-1:1998. *Information Technology-Software Measurement-Functional Size Measurement-Part 1: Definitions of Concepts*. IEEE, 2000.
- ST17. ANSI/PMI 99-001-2000. *A Guide to the Project Management Body of Knowledge*. 3 edition. The American National Standard Institute, December 2004.
- ST18. ISO/IEC TR 15504. *Information Technology. Process Assessment*:
ISO/IEC 15504-1:2004. Part 1: *Concepts and vocabulary*.
ISO/IEC 15504-2:2003. Part 2: *Performing an assessment*.
ISO/IEC 15504-3:2004. Part 3: *Guidance on performing an assessment*.
ISO/IEC 15504-4:2004. Part 4: *Guidance on use for process improvement and process capability determination*.

ISO/IEC 15504-5:2006. Part 5: *An exemplar Process Assessment Model.*

(galima rasti adresu <http://isospice.com/standard/tr15504.htm>).

ST19. ISO 9000 standartų šeimai priklauso:

ISO 9000:2000. *Quality management systems - Fundamentals and vocabulary.*

ISO 9001:2000. *Quality management systems – Requirements.*

ISO 9004:2000. *Quality management systems - Guidelines for performance improvements.*

Kartais šiai šeimai yra priskiriami dar ir kai kurie kiti standartai, neturintys numerio 900x.

ST20. ISO/IEC 9126. *Software Engineering. Product Quality:*

ISO/IEC 9126-1-2001. Part 1: *Quality Model.*

ISO/IEC TR 9126-2-2003. Part 2: *External Metrics.*

ISO/IEC TR 9126-3-2003. Part 3: *Internal Metrics.*

ISO/IEC TR 9126-4-2004. Part 4: *Quality in Use.*

ST21. MIL STD 498. *Software Development and Documentation.* Department of Defence, 1994 (galima rasti adresu http://www.pogner.demon.co.uk/mil_498).

ST22. MIL-STD-882D. *Standard Practice for System Safety.* Department of Defence, 10 February 2000 (galima rasti adresu <http://www.safetycenter.navy.mil/instructions/osh/milstd882d.pdf>).

ST23. MIL-STD-2167A. *Defense System Software Development.* Department of Defence, 1994 (galima rasti adresu <http://www2.umassd.edu/SWPI/DOD/MIL-STD-2167A/DOD2167A.html>).

ST24. MIL-STD-490A. *Specification Practices.* Department of Defence, 1985.

ST25. DI-MCCR-80025A. *Software Requirements Specification.* Department of Defence, 1988. 02. 29 (priedas prie MIL-STD-2167A).

ST26. DOD-Std-79354. *DOD Automated Information Systems (AIS).Documentation Standards.* U.S. Department of Defense, October 31, 1988.

ST27. DoD Regulation 5000.2-R *Mandatory Procedures for Major Defense Acquisition Programs (MDAPs) and Major Automated Information System (MAIS) Acquisition Programs,* 1997.

ST28. ITU-T Recommendation X.901 | ISO/IEC 10746-1: 1996. *Reference Model of Open Distributed Processing (RM-ODP)-Part 1: Overview and Guide to Use.* ISO/IEC, 1996.

ST29. ITU-T Recommendation X.902 | ISO/IEC 10746-2: 1995. *Reference Model of Open Distributed Processing (RM-ODP)-Part 2: Descriptive Model.* ISO/IEC, 1995.

ST30. ITU-T Recommendation X.903 | ISO/IEC 10746-3: 1995. *Reference Model of Open Distributed Processing (RM-ODP)-Part 3: Prescriptive Model.* ISO/IEC, 1995.

ST31. ITU-T Recommendation X.904 | ISO/IEC 10746-4: 1996. *Reference Model of Open Distributed Processing (RM-ODP)-Part 4: Architectural Semantics.* ISO/IEC, 1996.

ST32. FIPS PUBS 38. *Guidelines for Documentation of Computer Programs and Automated Data Systems.* Federal Information Processing Standards Publication, National Bureau of Standards, February 15, 1976.

ST33. FIPS PUB 140-2. *Security Requirements for Cryptographic Modules.* Federal Information Processing Standards Publication, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD

- 20899-8900, Issued May 25, 2001 (galima rasti adresu <http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf>).
- ST34. ESA PSS-05-01 Issue 1 Revision 1. *Guide to the Software Engineering Standards*. European Space Agency, March 1995 (galima rasti adresu www.atlsysguild.com).
- ST35. SMAP-DID-P200-SY. *Product Specification Documentation Standard: Information System Requirements*. Release 4.3 National Aeronautics and Space Administration, February 28, 1989.
- ST36. BS 6719: 1986. *British Standard Guide to Specifying User Requirements for a Computer-Based Systems*. British Standards Institution, 1986.
- ST37. CSA Standard Z243.15.4-1979. *Basic Guidelines for the Structure of Documentation of System Design Information*. Canadian Standards Association, 1979.
- ST38. IPTC Standards. Events ML. *Business Requirements Specification Document*. IPTC, 2004.
- ST39. CEFAC/T/ICG/005. *Business Requirements Specification (BRS)*. UN/CEFAC/T, 2004.

Terminų žodynėlis

- Trm1. Bendro darbo grupė** (angl. *joint application development (JAD)*) – Programų sistemų kūrimo būdas, kuomet projekte lygiomis teisėmis bendrai dirba informatikos ir dalykinės srities specialistai. Daroma prielaida, kad niekas geriau neišmano kompiuterizuojamų procedūrų už žmones, kurie jas vykdo, ir niekas neišmano kaip kurti programų sistemas už programų sistemų inžinierius. Todėl gerą sistemą galima sukurti tik tuomet, kuomet abi specialistų grupės dirba kartu. Kuriant sistemą tokiu būdu dalykinės srities specialistai yra įtraukiami ne tik į reikalavimų inžinerijos, bet ir į sistemos projektavimo bei testavimo darbus. Lyginant su kitais sistemų kūrimo būdais bendro darbo grupės kainuoja daugiau, nes nuolatiniam darbui reikia ne tik vykdytojų, bet ir užsakovo darbuotojų grupės. Tuos darbuotojus tenka keliems mėnesiams atitraukti nuo tiesioginio darbo. Be to, šitaip organizuotą projektą yra sunkiau valdyti. Kita vertus, yra sukuriamas geresnės sistemos ir supaprastėja jų diegimas, nes pradedant diegti sistemą grupė užsakovo darbuotojų jau yra išsamiai susipažinusi su ta sistema.
- Trm2. Biuro sistema** (angl. *office system*) – sistema, padedanti padidinti raštvedybos ir kitų kanceliarinio pobūdžio darbų efektyvumą ir patikimumą. Paprastai sistema teikia tekstų redagavimo ir spausdinimo, paprastų skaičiavimų, prezentacijų rengimo ir komunikavimo paslaugas. Bet kuri biuro sistema skirta tam tikriems darbo objektams (angl. *workpiece*) apdoroti ir pateikia tam tikrą tam reikalingą priemonių (instrumentų) rinkinį. Biuro sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Pavyzdys: biuro techninio aptarnavimo grupė, kurioje dirba redaktorė, mašininkė, braižytojas ir laiškus išsiunčianti bei gautą korespondenciją paskirstanti sekretorė (be abejo, tai archaiška, moraliai pasenusi sistema); MS Office.
- Trm3. Darbų srautų tvarkymo sistema** (angl. *workflow management system*) – sistema, skirta tvarkyti organizacijos verslo užduočių sekoms, kuriomis kuriami kokie nors į organizacijos išore pateikiamiems produktams, t. y. gaminiams arba paslaugoms. Sistema leidžia apibrėžti užduočių sekas, priskirti užduotims jų vykdytojus ir, baigus vykdyti eilinę užduotį, informuoja apie tai kitos užduoties vykdytojus, kartu pateikdama jiems visą tos užduoties vykdymui reikalingą informaciją, paprastai, vykdant ankstesnę užduotį parengtus ar kaip nors apdorotus dokumentus. Darbo srautų tvarkymo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos.
- Trm4. Darbuotojų lavinimas** (angl. *education*) – Reikalavimų inžinerijoje suprantamas kaip teorinių principų, kuriais yra grindžiamas koks nors metodas ar procesas, dėstymas. Paprastai yra vykdomas skaitant paskaitas ir rengiant seminarus, kuiuose šie principai yra aiškinamiesi kolktivui.
- Trm5. Darbuotojų mokymas** (angl. *mentoring*) – Reikalavimų inžinerijoje suprantamas kaip kokio nors metodo ar proceso procedūrų aiškinimas ir demonstravimas, kaip jas reikia atlikti.
- Trm6. Darbuotojų treniravimas** (angl. *training*) – Reikalavimų inžinerijoje suprantamas kaip mokomojo pobūdžio praktinių užduočių vykdymas, atliekamas tikslu įgyti darbo su kokio nors metodo ar proceso procedūromis praktinius įgūdžius.

Trm7. Dokumentų tvarkymo sistema (angl. *document management system*) – speciali išteklių tvarkymo sistemų rūšis. Sistema kaupia organizacijoje naudojamus dokumentus specialiose saugyklose, juos sistemina, tvarko, archyvuoja ir pateikia visiems norintiems jais pasinaudoti ir turintiems tam teise. Dokumentų tvarkymo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Kompiuterizuotas dokumentų tvarkymo sistemas paprastai sudaro popierinių dokumentų skenavimo ir indeksavimo posistemis, dokumentų bazė, kurioje tam tikru standartiniu formatu yra saugomi patys dokumentai, bazėje saugomų dokumentų indeksas, tą bazę aptarnaujančios programos, dokumentų paieškos mašina, dokumentų pateikties internete programos ir galbūt archyvuotų dokumentų bazė ir dokumentų archyvavimo programos. Pavyzdžiai: spinta, kurioje aplankuose sudėti tam tikru būdu suklasifikuoti dokumentai, aplankų sąrašas ir sekretorė, tvarkanti tuos dokumentus ir išduodanti juos organizacijos darbuotojams laikinam naudojimui;

Trm8. Duomenų apdorojimo sistema (angl. *data processing system*) – sistema, kuri, atlikdama tam tikrus iš anksto nustatytus skaičiavimus, iš jai pateiktų pradinių duomenų gauna kokius nors iš anksto žinomo pobūdžio rezultatus. Duomenų apdorojimo sistema gali būti nekompiuterizuota, iš dalies kompiuterizuota arba visiškai kompiuterizuota. Nekompiuterizuotose sistemoje gali būti naudojami mechaniniai, elektriniai arba elektroniniai kalkuliatoriai bei kokie nors kitokie duomenų apdorojimo įrenginiai. Pavyzdys: darbo užmokesčio skaičiavimo sistema.

Trm9. Ekspertinė sistema (angl. *expert system*) – sistema, skirta kam nors diagnozuoti, prognozuoti, vertinti ar interpretuoti, remiantis grupės ekspertų sukauptomis žiniomis apie atitinkamą dalykinę sritį. Ekspertinės sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos, nors pats terminas ekspertinė sistema pradėtas vartoti tiktais pradėjus kurti kompiuterizuotas sistemas. Kompiuterizuotose ekspertinėse sistemoje ekspertų žinios yra koduojamos produkcijų tipo taisyklėmis, freimais arba scenarijais. Greta to yra naudojamos formalizuotos ekspertinių samprotavimų procedūros ir paprastai yra pateikiamas ne tik galutinis rezultatas, bet ir paaškinimai, kodėl buvo gautas būtent toks rezultatas. Pavyzdžiai: iš palydovo padarytų nuotraukų interpretavimo sistema; rūko ir viesulų prognozavimo sistema.

Trm10. Ekstremalusis programavimas [58] (angl. *extreme programming (XP)*) – agilus (supaprastintas) programų sistemų kūrimo būdas. Šis būdas atmeta daugumą tradicinių programų sistemų kūrimo praktikų, aprašytų IEEE standarte 830 [ST3], išskaitant reikalavimą rengti reikalavimų specifikaciją, klasikinius inspektavimo metodus [29], darbo apimties vertinimą pagal funkcinių taškų skaičių [34] ar pagal COCOMO modelį [6] ir kt. Ekstremaliojo programavimo požiūriu, programų sistemai dokumentuoti pakanka programų kodo, tam kodui testuoti skirtų testų ir programuotojo atminties. Reikalavimus aiškinamasi žodžiu, tampriai bendraujant su *nuolat prieinamu užsakovo atstovu* <Trm33> (ekstremalusis programavimas nepripažista jokių kitų reikalavimų šaltinių, taip pat ir kitų šalių, suinteresuotų kuriamaja programų sistema) ir dokumentuojami kaip labai trumpos *naudotojų istorijos* <Trm31>, kiekviena iš kurių užrašoma atskiroje kortelėje. Ekstremaliojo programavimo terminais, ji vadinama *indekso korta*. Daroma prielaida, kad

bet kuriuo laiko momentu galima greitai susisekti su užsakovo atstovu ir išsiaiškinti visas dviprasmybes ar kitus bet kurios naudotojo istorijos neaiškumus. Programos kuriamos vadovaujantis evoliuciniu programų sistemų gyvavimo ciklo modeliu, skaidant produktą į kuo mažesnius prieaugius (vienos prieaugio kūrimas neturi trukti daugiau kaip 2 mėnesius). Pradedant kurti naują sistemos prieaugį, programuotojai susitinka su užsakovo atstovu ir kartu įvertina, kiek laiko jiems prireiks tam prieaugui sukurti. Ekstremaliojo programavimo terminais tai vadinama *planavimo žaidimu* (angl. *planing game*). Inspektavimo metodai keičiami vadinamuoju programavimu poromis: du programuotojai dirba prie vieno kompiuterio, vienas iš jų, pasitardamas su kitu, kompiuterio ekrane rašo programos tekštą. Antrasis stebi programos tekštą ir ieško joje klaidų. Dar vienas iš specifinių ekstremaliojo programavimo ypatumu yra dažnas automatiškai vykdomas testavimas. Reikalaujama, kad testai būtų ne tik kuriami, bet ir prižiūrimi, t. y., kad jie visuomet būtų aktualūs. Be to, skirtingai nei naudojant kitas programų sistemų kūrimo metodikas, testai yra kuriami prieš pradedant programą rašymą. Naudojant ekstremaliojo programavimo metodiką, testai būtinai turi būti vykdomi automatiškai, nes reikalaujama, kad vyktų permanentinis parašyto kodo integravimas į kuriamą programų sistemą (paprastai, programuotojas tą daro kelis kartus per dieną).

Trm11. Fokuso grupė (angl. *focus group*) – Kolektyvinio svarstymo metodas.

Surenkama nedidelė suinteresuotų šalių atstovų grupė ir jai pateikiamas koks nors vienas klausimas. Siekiama suvienodinti galimai skirtingus požiūrius. Reikalavimų inžinerijoje naudojamas nuomonėms bei paprastiems ar labai detalizuotiems duomenims rinkti.

Trm12. Funkciniai reikalavimai (angl. *functional requirements*) –

Reikalavimai, nusakantys, kokias paslaugas privalo teikti sistema, kokia turi būti jos reakcija į konkrečius stimulus ir kaip ji turi elgtis konkrečiose situacijose.

Trm13. Galimybės keisti sistemos mastą (angl. *system scalability*) –

Galimybė palaipsniui didinti sistemos pajėgumus (duomenų apimtis, aptarnaujamų darbo vietų ar įrenginių skaičių ir kt.), beveik neprarandant sistemos našumo ir nedarant esminių esamos programinės ir techninės įrangos pakeitimų ir nekeičiant sistemos naudojimo procesų.

Trm14. Gamybos proceso valdymo sistema (angl. *manufacture control system*) – sistema,. Gamybos proceso valdymo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos.

Trm15. Grupinio darbo organizavimo sistema (angl. *group support system*) –

sistema, įgalinanti grupę asmenų tokiu būdu bendrai naudotis kokia nors informacija, kad bet kurio asmens padaryti pakeitimai iš karto tampa matomi visiems grupės nariams. Be to, paprastai tokia sistema numato priemones grupės darbui planuoti, koordinuoti, stebeti bei vertinti ir priemones grupės nariams bendrauti tarpusavyje. Grupinio darbo organizavimo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Pavyzdys: kompiuterinė konferencijų organizavimo sistema.

Trm16. Horizontalusis maketas (angl. *horizontal prototype, behavioral prototype, mock-up prototype*) – Maketas <Trm30>, naudojamas

naudotojo interfeisų reikalavimams aiškintis. Horizontaliaisiais tokie maketai yra todėl vadinami, kad jie makeduoja tik vieną sistemos architektūros sluoksnį – naudotojo interfeiso sluoksnį. Horizontalusis maketas makeduoja sistemos interfeiso langus ir leidžia juos keisti, naudotis meniu ir kitomis interfeiso galimybėmis, tačiau jis nerealizuojatokio funkcionalumo, o tik imituojat sistemos darbą, pateikdamas atitinkamus pranešimus. Tokie maketai dar yra vadinami imitaciniams maketais, elgsenos maketais.

Trm17. Informacijos apdorojimo taisykles (angl. *information processing rules*) – Informacijos apdorojimo taisykles apima buhalterines taisykles, saskaitybos taisykles, duomenų apibendrinimo bei agregavimo (sustaminimo) taisykles ir kitas informacinių objektų transformavimo taisykles (skaičiavimo algoritmus) bet jos taip pat nustato kaip yra organizuojami bei apdorojami duomenų srautai, per kiek laiko turi būti atlikti vieni ar kiti skaičiavimai, koks turi būti gautų rezultatų tikslumas, jų patikimumas bei kitus panašius dalykus. Dalis informacijos apdorojimo taisyklių yra išvedamos iš verslo taisyklių, dalis yra nustatyta atitinkamais teisės aktais ar standartais, dalis išplaukia iš organizacijos praktikų bei tradicijų. Informacijos apdorojimo taisykles nėra programų sistemų reikalavimai, nes didžioji jų dauguma nepriklauso nuo to ar organizacija yra kompiuterizuota ar ne ir kokiui mastui tai yra padaryta. Kita vertus, kai kurios informacijos apdorojimo taisykles, ypač su duomenų srautais susijusios taisykles, kompiuterizuojant organizaciją gali būti keičiamos, bet ir tada jos lieka neprilausomos nuo to, kokia konkreti programinė įranga yra naudojama organizacijai kompiuterizuoti. Dažnai informacijos apdorojimo taisykles yra traktuojamos kaip verslo taisyklių dalis ir yra vadinamos tiesiog verslo taisykliemis.

Trm18. Informavimo sistema (angl. *help desk*) – speciali užklausų sistemų rūsis, teikianti vartotojams tam tikro iš anksto nustatyto pobūdžio informaciją ir galbūt asistuojanti jiems vykdant tam tikro pobūdžio užduotis. Informavimo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Pavyzdžiai: organizacijos padalinys, telefonu teikiantis darbuotojams konsultacijas, kaip pasinaudoti organizacijos informacine sistema; MS Word informacinės pagalbos sistema (angl. help); interneto svetainė, kurioje galima klausti apie organizacijos teikiamų paslaugų ypatumus ir kainas.

Trm19. Išmetamasis maketas (angl. *throw away prototype, exploratory prototype*) – Maketas, kuris kuriamas tik reikalavimams formuluoti, analizuoti ar vertinti, bet nėra traktuojamas kaip kuriamosios sistemas prototipas (t. y. neperauga į realią sistemą) ir, atlirkus planuotus tyrimus ar bandymus, yra sunaikinamas.

Trm20. Ištaklis (angl. *resource*) – Esybė (žaliava, energija, įrengimas, personalas ar pan.), reikalinga kokiai nors veiklai vykdyti. Ištaklis gali būti vienkartinio panaudojimo (pvz., energija) arba daugkartinio panaudojimo (pvz. įrenginys). Organizacijos turimų ištaklių kiekis visuomet yra ribotas, todėl iškyla ištaklių skirtumo (angl. *resource allocation*) problema, t. y. problema, kaip paskirstyti turimus ištaklius, kad, viena vertus, tų ištaklių būtų sunaudota kuo mažiau, ir, kita vertus, būtų pasiekti norimi tikslai.

Trm21. Ištaklių tvarkymo sistema (angl. *resource management system*) – Sistema, planuojanti ištaklių įsigijimą, atliekanti turimų ištaklių apskaitą ir

Trm22. Igyvendinamumo analizė (angl. *feasibility study*) – išsamus projekto nagrinėjimas bei vertinimas, ar įmanoma ir ar tikslinga ji įgyvendinti techniniu, operaciniu, ekonominiu, juridiniu ir galbūt kitais požiūriais. Atliekant įgyvendinamumo analizę, paprastai yra analizuojami keli galimi projekto įgyvendinimo būdai ir iš jų pagal duotus vertinimo kriterijus yra pasirenkamas geriausias. Reikalavimų inžinerijos kontekste, atliekama kiekvieno lygmens reikalavimų įgyvendinamumo analizė. Pagal duotus kriterijus (kainą, realizavimo laiką, patikimumą, našumą, panaudojamumą ir kt.) parenkama to lygmens reikalavimams įgyvendinti geriausiai tinkanti sistemos architektūra.

Trm23. Juodoji dėžė (angl. *black box*) – Sistema ar įrenginys, kurių vidinė struktūra, vidinės būsenos ir veikimo principai nėra žinomi, tačiau žinomi įėties duomenys (įskaitant stimulus), rezultatai (įskaitant reakcijas), jų tarpusavio ryšiai ir iš išorės stebimos būsenos.

Trm24. Kokybės funkcijų sklaida (angl. *quality function deployment*, jap. *Hin Shitsu (kokybē) Ki No (funkcija) Ten Kai (sklaida, plētra, difuzija)*) – Reikalavimų inžinerijos metodika, sukurta Japonijoje apie 1966 m. Kokybės funkcijų sklaidos (KFS) metodika yra ypač tinkama tais atvejais, kada sistema kuriamą konkrečiam užsakovui ir jo nuomonė apie tai, kokią sistemą reikia sukurti, nulemia sistemos reikalavimus. Kalbant apie programą sistemą reikalavimus, kartais dar yra vartojoamas terminas programinės įrangos kokybės sklaida (angl. *software quality deployment*). Metodika numato būdus kaip užsakovo reikalavimus pertvarkyti į atitinkamus sistemas techninius reikalavimus. Reikalavimams, jų prioritetams ir kitiems su tuo susijusiais klausimams aprašyti KFS metodikoje paprastai yra naudojama speciali matricinė struktūra vadinama kokybės korta arba kokybės nameliu, tačiau kokybės kortos naudojimas nėra privalomas. Ją sudarančios matricos gali būti vizualizuojamos ir kitaip. Reikalavimų aprašymas matricomis priverčia užsakovą ir vykdymo apgalvoti visus kuriamos sistemas kokybės reikalavimus, nustatyti jų prioritetus ir apibrėžti matus, skirtus tų reikalavimų įgyvendinamumo laipsniui matuoti. Metodika leidžia dirbti su daugelio lygmenų reikalavimais, apimant ne tik skirtingų sistemos lygmenų, bet ir tos sistemas komponentų bei jos kūrimo proceso reikalavimus. Naudojant šią metodiką, reikalavimų trasos yra konstruojamos, tarpusavyje susiejant skirtingų lygmenų matricas. Sukurta daug instrumentinių sistemų, skirtų KFS metodikai palaikyti.

Trm25. Kokybės valdymo sistema (angl. *quality management system*) – Išreikštiniu būdu suformuluotos nuostatos ir organizacijoje oficialiai įteisintos procedūros, nustatancios kaip turi būti planuojama, vertinama ir kontroliuojama organizacijos gaminamų gaminių ar/ir teikiamų paslaugų kokybę.

Trm26. Kolektyvinė reikalavimų analizė (angl. *joint requirements analysis* (JRA)) – reikalavimų aiškinimosi būdas, primenantis fokusą grupę <Trm11>. Suinteresuotų šalių atstovai yra suburiami į darbo grupę, vadovaujamą analitiko-moderatoriaus, ir patys analizuoją verslo funkcijas,

procesus, veiklas bei duomenis. Nuo fokuso grupės skiriasi savo paskirtimi ir tuo, kad grupė yra ilgalaikė.

Trm27. Kolektyvinis svarstymas (angl. *facilitated meeting*) – reikalavimų aiškinimosi metodas, padedantis gauti tam tikrą sinergetinį, arba, kitaip tariant, sumarinį efektą. Dirbdama kartu grupė žmonių gali išsiaiškinti greičiau ir galbūt daugiau, negu tai pavyktų padaryti analitikui, dirbant su kiekvienu iš tų žmonių atskirai. Svarstymo metu gali vykti „smegenų šturmo“ sesijos, gali būti tikslinamos taip gimusios idėjos ir gali būti išsiaiškinta daugelis dalykų, kurių nepavyktų išsiaiškinti imant interviu, nes analitikas nieko apie tai nežinojo ir, savaimė aišku, apie tai nieko neklause. Be to, tokio svarstymo metu išryškėja skirtinges nuomonės bei prieštarangi reikalavimai ir galbūt netgi gali būti pasiekti tam tikri kompromisai. Kita vertus, analitikui nėra paprasta tokius pasitarimus vesti, nes galima „uzstrigtī“ ant smulkmenų, nukrypti į šoną, arba priimti neteisingus sprendimus, spaudžiant kokiai nors interesų grupei arba bijant paprieštarauti kokio nors autoritetingo grupės nario nuomonei.

Trm28. Kompiuterizuota programų sistemų inžinerijos palaikymo sistema (angl. *computer-aided software engineering system (CASE system)*) – tai integruotas rinkinys įrankių, palaikančių visas pasirinkto gyvavimo ciklo modelio visose numatytose stadijose vykdomas programų sistemų inžinerijos veiklas. Modernios CASE sistemos, be kita ko, palaiko grupinių darbų ir keletą sistemos kūrimo metodų. Kita vertus, CASE sistemomis dažnai yra vadinamos sistemos, palaikančios tik grafinį programų sistemų projektavimą ir suprojektuotų sistemų kodo generavimą. Pavyzdžiai: Enterprise Architect 3.10 (Sparx Systems), GDPro (Advanced Software Technologies, Inc.), Oracle Designer/2000 (Oracle Systems Corp.), Rational Rose (IBM).

Trm29. Konteksto diagrama (angl. *context diagram*) – kontekstą ir sudaryti vadinamąjį konteksto Diagrama, parodanti kaip kuriamoji sistema yra susieta su savo kontekstu <Trm69>, tiksliau su tais konteksto elementais, su kuriais ji keičiasi kokia nors informacija. Konteksto diagrama nustato kuriamosios sistemos ribas (t. y. jos apimtį) ir apibrėžia tos sistemos informacijos mainų interfeisus su jos išorėje esančiomis esybėmis, tokiomis kaip jos naudotojai, jos valdomi ar aptarnaujami įrenginiai, kitos sistemos, iš kurių ji gauna arba kurioms ji pateikia kokią nors informaciją, ir pan.

Trm30. Maketas (angl. *prototype*) – bet kuri dalinė, preliminari arba potencialiai galima kuriamos sistemos realizacija. Turint sistemos maketą galima geriau suvokti kaip ji atrodys ir lengviau išsiaiškinti jos reikalavimų trūkumus. Paprastai yra kuriami tik programų sistemų maketai. Verslo sistemos ir informacinės sistemos kol kas yra maketuojamos labai retai. Skiriami išmetamieji maketai <Trm19> ir kuriamos programų sistemos prototipai <Trm73>. Taip pat yra skiriami horizontalieji <Trm16> ir vertikalieji maketai <Trm93>. Maketams kurti yra naudojamos įvairios programavimo kalbos, skriptų rašymo kalbos, žymių kalbos, kompiuterinės brėžinių braižymo priemonės ir kt. Yra specialiai maketams kurti skirtų komercinių pakeų, vadinamų maketu generatoriais (angl. *screen painters, graphical user interface builders*). Dauguma šiuolaikinių programų sistemų inžinerijai palaikyti skirtų instrumentinių sistemų <Trm27> taip pat turi specialiai maketams kurti

pritaikytas priemones. Neturint specialių maketavimo priemonių arba norint atpiginti maketavimo darbus yra naudojami ant popieriaus lapų ar ant mokylinės lento braižomi maketai (angl. *paper prototype*, *lo-fi prototype*). Taip galima kurti tik horizontaliuosius maketus. Jie yra labai primityvūs, tačiau dažnai jų pakanka kai kuriems interfeiso reikalavimams išsiaiškinti.

Trm31. Naudotojo istorija (angl. *user story*) – programų sistemos reikalavimai ekstremaliojo programavimo metodikoje <Trm10>. Istorija susideda iš dviejų dalių: rašytinės ir žodinės. Rašytinė dalis yra labai trumpa, surašoma vienoje nedidelio formato kortelėje (vadinamojoje *indekso kortoje*). Joje surašoma tik apie ką bus kalbamasi su užsakovu. Istorija neturi būti nei išsami, nei paprastai suprantama, nes šioje metodikoje yra nuolat prieinamas užsakovo atstovas <Trm33>, su kuriuo bet kuriuo momentu galima išsiaiškinti bet kuriuos neaiškumus. Įgyvendinus istorijoje aptartus reikalavimus, korta, kurioje surašyta istorija, dažniausiai yra išmetama. Jei, bekuriant eilinių sistemos prieaugi, užsakovui prieikia naujos istorijos arba jei jis nori kurią nors istoriją pakeisti, jam leidžiama tą daryti. Užsakovo atstovas susitinka su vykdytojais, aptariamos naujos istorijos įgyvendinimo galimybės bei terminai ir nebeaktualios indekso kortos pakeičiamos naujomis.

Trm32. Nefunkciniai reikalavimai (angl. *nonfunctional requirements*) – Funkcinių reikalavimų įgyvendinimo būdo ribojimai.

Trm33. Nuolat prieinamas užsakovo atstovas (angl. *on-site customer*) – vienas iš vaidmenų ekstremaliojo programavimo metodikoje <Trm10>. Lyginant su kitomis programų sistemų kūrimo metodikomis, šis vaidmuo yra tuo ypatingas, kad su juo galima susisiesti bet kuriuo laiko momentu ir kad jis privalo duoti oficialų atsakymą į bet kurį vykdytojų pateiktą klausimą.

Trm34. Operaciniai poreikiai (angl. *operational needs*) – Informacinių, skaičiavimo, komunikavimo ir galbūt kitokios paslaugos, palengvinančios dalykinės srities specialistams vykdyti verslo užduotis. Operaciniai poreikiai ir vartotojo reikalavimai <Trm85> yra labai susiję, nes paprastai vartotojai reikalauja, kad būtų patenkinti jų operaciniai poreikiai. Paprastai operaciniai poreikiai yra aprašomi dokumente „Operacinių poreikių specifikacija“ <Trm35>.

Trm35. Operacinių poreikių specifikacija (angl. *needs specification*, *requirements document*, *user requirements specification*, *operational requirements document*, *concept of operations*, *operational concept*) – Dokumentas, skirtas vartotojo lygmens reikalavimams <Trm85> aprašyti. Jis aprašo, kokius vartotojų operacinius poreikius <Trm34> turi tenkinti sistema, tačiau nesigilina į tai, kokiu būdu tai bus padaryta.

Trm36. Organizacijos integruota informacinė sistema (angl. *enterprise system*) – Informacinių sistemos, aptarnaujanti visą organizaciją, o ne tik kokią nors tos organizacijos padalinį, veiklos barą ar verslo procesą.

Trm37. Organizacijos veiklos strategija (angl. *organisational strategy*) – nuostatos, siekiai ir nurodymai, nustatantys vidinių organizacijos padalinijų (pardavimų skyrius, buhalterija ir t.t.) tikslus ir veikimo būdus. Organizacijos veiklos strategijos dar yra vadinamos funkcinio lygmens strategijomis. Funkcinio lygmens strategijos paprastai turi būti palaikomos įvairiomis informaciniemis ir komunikavimo technologijomis.

Trm38. Paieškos mašina (angl. *search engine*) –speciali užklausų sistemų rūšis, padedanti vartotojui rasti jam reikiamą informaciją kokoje nors tekštinių dokumentų bazėje, pavyzdžiu, bibliotekos knygų kataloge, tokį bazių rinkinyje arba visame interne. Paieškos mašiną sudaro trys komponentai: dokumentų indeksas, informacinės paieškos pagal nurodytus kriterijus procedūros, rastų dokumentų reitingavimo taisyklės ir dokumentų indekso atnaujinimo priemonės. Paieškos mašinos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos, tačiau pats terminas paieškos mašina pradėtas vartoti tik pradėjus kurti paieškos interne mašinas. Kompiuterizuotose paieškos mašinose dokumentų indeksas saugomas specialiai tam skirtame serveryje arba net keliuose tokiuose serveriuose, informacinės paieškos procedūras ir reitingavimo taisyklės realizuoja speciali programa, o dokumentų indekso atnaujinimo priemonės realizuotas kaip programiniai robotai (angl. softbot), nuolat renkantys visame interne duomenis, reikalingus dokumentų indeksui papildyti ir atnaujinti. Pavyzdžiai: Alta Vista, Google.

Trm39. Pasiūlymas dalyvauti konkurse (angl. *request for proposal (RFP)*) – Užsakovo parengtas dokumentas, kuriuo visiems norintiesiems tiekėjams siūloma dalyvauti konkurse, paskelbtam sistemai įsigyti (t. y. pirkti turimą arba užsakyti sukurti). Dokumente turi būti specifikuotos perkamos sistemos savybės, todėl iš esmės jis turi būti rengiamas taip pat, kaip operacinių poreikių specifikacija <Trm35>.

Trm40. Planavimo sistema (angl. *scheduling system*) – sistema,. Planavimo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos.

Trm41. Produktų galimybė (angl. *feature*) – Programų sistemos galimybe yra vadinamas logiškai tarpusavyje susijusių funkcinių reikalavimų rinkinys, įgyvendinus kurį vartotojui sudaroma galimybė siekti tam tikro jam svarbaus tikslų. Kalbant apie rinkoje parduodamas programų sistemas, šis terminas apibūdina grupę reikalavimų, turinčią potencialiems pirkėjams aiškiai suprantamą dalykinę paskirtį. Pirkėjų pageidaujamų galimybių sąrašas nesutampa su jo operacinių poreikių sąrašu. Žinomais pageidaujamų galimybių pavyzdžiais yra galimybė interneto naršyklije išsaugoti peržiūrimų tinklalapių adresus, antivirusinės programos galimybė automatiškai atnaujinti virusų katalogą ir tekstu redagavimo programos galimybė surasti tekste padarytas rašybos klaidas. Galimybė galima pasinaudoti, vykdant skirtingas užduotis, todėl galimybių negalima tapatinti su operaciniais vartotojų poreikiais.

Trm42. Programų sistemos kūrimo (technologinis) procesas (angl. *software process*) – programų sistemai ir su ja susijusiems produktams (pvz., projekto planui, projektinei dokumentacijai, programų kodui, testams ir vartotojo dokumentacijai) sukurti ir prižiūrėti naudojamas veiklų, metodų, praktikų ir transformacijų rinkinys [88].

Trm43. Programų sistemos kūrimo (technologinio) proceso branda (angl. *software process maturity*) – Laipsnis, kurio procezas yra išreikštiniu būdu apibrėžtas, valdomas, matuojamas, kontroliuojamas ir efektyvus. Branda nusako turimų gebėjimų didinimo galimybes ir parodo tiek organizacijoje naudojamo programų sistemų kūrimo proceso turtingumą, tiek ir tai, kiek nuosekliai juo naudojasi organizacija, vykdyma projektus [88].

Trm44. Programų sistemos reikalavimai (angl. *software requirements*) – sąlygos, kurias turi tenkinti programų sistema, arba gebėjimai, kuriuos ji turi turėti, kad padėtų naudotojams siekti jiems reikiamų tikslų, ir spręsti jiems reikiamus uždavinius. Programų sistemos reikalavimai aprašomi dokumente „Programų sistemos reikalavimų specifikacija“ <Trm45>.

Trm45. Programų sistemos reikalavimų specifikacija (angl. *software requirements specification, functional specification, product specification, requirements document, system specification*¹⁸) – Dokumentas, aprašantis programų sistemos reikalavimus <Trm44>.

Trm46. Reikalavimų aiškinimasis (angl. *requirements elicitation*) – Reikalavimų inžinerijos veikla, apimanti reikalavimų suvokimą <Trm59> ir reikalavimų rinkimą <Trm56>. Kartais terminai reikalavimų aiškinimasis ir reikalavimų rinkimas vartojami kaip sinonimai.

Trm47. Reikalavimų analizė (angl. *requirements analysis*) – Reikalavimų inžinerijos veikla, vykdoma tikslu pašalinti svarbiausius surinktų reikalavimų trūkumus bei gauti tokios kokybės ir tokio detalumo reikalavimus, kad vykdytojas galėtų pakankamai objektyviai įvertinti projekto kainą, trukmę bei kitų projektui vykdyti reikalingų ištaklių apimtis, o inžinerinis personalas galėtų pradėti tuos reikalavimus įgyvendinti. Reikalavimų analizė apima kuriamos sistemos konteksto modeliavimą <Trm29>, reikalavimų konkretizavimą, mакетavimą <Trm51>, prieštaravimų ir kitų trūkumų paiešką bei šalinimą, darnos su aukštesnio lygmens reikalavimais analizę, įgyvendinamumo analizę <Trm49>, modeliavimą <Trm52>, duomenų žodyno sudarymą, sistemos architektūros parinkimą, reikalavimų prioretizavimą <Trm55>, lokalizavimą <Trm50>, nuleidimą žemyn <Trm53>, operacionalizavimą <Trm54> ir kokybės funkcijų sklaidą <Trm24>.

Trm48. Reikalavimų formulavimas (angl. *requirements discovery*) – Reikalavimų inžinerijos veikla, vykdoma tikslu pertvarkyti operacinius poreikius į funkcinius ir nefunkcinius sistemos reikalavimus.

Trm49. Reikalavimų įgyvendinamumo analizė (angl. *requirements feasibility study*) – Reikalavimų inžinerijos procesas, kuriuo siekiama parodyti, kad, atsižvelgiant į kainos ribojimus, projekto terminus ir projekto vykdytojų kvalifikaciją, sistemos reikalavimus yra realu įgyvendinti. Iš esmės tai reikalavimų įgyvendinamumo veiksnių analizė. Skirtingai nuo sistemos įgyvendinamumo analizės, daromos iš užsakovo požiūrio taško tikslu išsiaiškinti, ar projektą apskritai verta pradėti, reikalavimų įgyvendinamumo analizė yra daroma iš vykdytojo požiūrio taško. Pagal jos rezultatus vykdytojas sprendžia, ar jam verta imtis projekto, ar realūs yra projekto vykdymo terminai ir jo finansavimas ir kaip maksimaliai apsisaugoti nuo projekto rizikos veiksnių keliamų grėsmių. Atliekant reikalavimų įgyvendinamumo analizę, sistemos įgyvendinamumas yra nagrinėjamas techniniu ir iš dalies projekto darbų plano aspektais, nesiaiskinant operacinio, ekonominio, juridinio ir etinio sistemos įgyvendinamumo.

Trm50. Reikalavimų lokalizavimas (angl. *requirements allocation*) – Reikalavimų lokalizavimu vadinamas jų skaidymo į tam tikras prasminių

¹⁸ Kaip matome, anglų kalboje terminas *system specification* gali būti vartojamas dviems skirtiniams dokumentams įvardinti. Šitaip gali būti vadinama ir sistemos reikalavimų specifikacija, ir programų sistemos reikalavimų specifikacija

požiūriu susietų reikalavimų grupes, kuriose kai kurie reikalavimai gali kartotis, ir tų grupių susiejimas su joms priklausančius reikalavimus realizuojančiais kuriamos sistemos komponentais.

Trm51. Reikalavimų maketavimas (angl. *requirements prototyping*) – Maketų <Trm30>, skirtų kuriamos sistemos reikalavimams aiškintis, analizuoti ar vertinti, kūrimo procesas. Jei į maketavimo procesą pavyksta įtraukti dalykinės srities specialistus ar kitus būsimuosius sistemos naudotojus, tai aptarinėjant maketą kartu su jais vykdytojams paprastai pavyksta geriau suprasti kokių sistemos interfeisių jie nori ir kokius uždavinius iš tiesų turėtų spręsti kuriamoji sistema. Taigi maketai padeda analizuoti ne tik interfeiso, bet ir funkcinius sistemos reikalavimus.

Trm52. Reikalavimų modeliavimas (angl. *requirements modelling*) – Iš išorės stebimos kuriamosios sistemos elgsenos modeliavimas. Modeliuojant reikalavimus, sistema yra traktuojama kaip juodoji dėžė <Trm23>, nesigilinant į jokius jos realizavimo klausimus ir nepriimant jokių projektavimo sprendimų. Kita vertus, paprastai yra modeliuojama ne tik pati sistema, bet ir verslo aplinka, kurioje ta sistema bus naudojama. Reikalavimuss modeliuoti galima UML, Z bei kitomis konceptinio modeliavimo ar formaliosiomis reikalavimų specifikavimo kalbomis.

Trm53. Reikalavimų nuleidimas žemyn (angl. *requirements flowdown*) – Reikalavimų nuleidimu žemyn yra vadinamas jų išreiškimas to lygmens, į kurį jie yra nuleidžiami, terminais. Reikalavimai yra nuleidžiami iš verslo lygmens į vartotojo lygmenį, iš vartotojo lygmenės sistemos lygmenį, iš informacinės sistemos lygmenis į programų sistemos lygmenį, iš programų sistemos lygmenis į jos komponentų lygmenį ir t.t. Nuleidžiant reikalavimus žemyn, vienas reikalavimas gali būti suskaidytas į kelis arba, atvirkščiai, keli reikalavimai gali būti sujungti į vieną. Taip pat gali tekti reikalavimus papildyti, nes gali tekti priimti tam tikrus sprendimus apie jų įgyvendinimo būdą. Kita vertus, nefunkcinius reikalavimus gali tekti operacionalizuoti, t. y. išreikšti juos funkcinių reikalavimų ar proceso reikalavimų terminais.

Trm54. Reikalavimų operacionalizavimas (angl. *requirements operationalisation*) – Nefunkcinių reikalavimų pertvarkymas į funkcinius, proceso ar technologinius reikalavimus.

Trm55. Reikalavimų prioretizavimas (angl. *requirements prioritising*) – Reikalavimų svarbos nustatymo procesas. Prioritetai yra priskiriami kiekvienam reikalavimui. Reikalavimų svarba gali būti nagrinėjama užsakovo, vykdytojo, priežiūros tarnybos ar kurios nors kitos suinteresuotosios šalies požiūriu.

Trm56. Reikalavimų rinkimas (angl. *requirements gathering*) – Reikalavimų inžinerijos veikla Kartais terminai reikalavimų rinkimas ir reikalavimų aiškinimasis <Trm46> vartojami kaip sinonimai.

Trm57. Reikalavimų specifikacija (angl. *requirements specification*) – Nustatytu būdu struktūruotas dokumentas, aprašantis ką turi daryti sistema ir kokius kitus reikalavimus bei ribojimus ji turi tenkinti. Gali būti rengiamos ne tik sistemos, bet ir interfeiso, testo, proceso ar kokio nors kito objekto reikalavimų specifikacijos.

Trm58. Reikalavimų specifikavimas (angl. *requirements specification*) – Reikalavimų inžinerijos veikla, vykdoma tikslu dokumentuoti surinktus,

išanalizuotus bei įvertintus reikalavimus ir parengti reikalavimų specifikaciją <Trm57>.

Trm59. Reikalavimų suvokimas (angl. *requirements discovery*) – Reikalavimų inžinerijos veikla, vykdoma tikslu nustatyti, kokius sistemos ar proceso reikalavimus iš kokių šaltinių galima surinkti. Prieš pradedant šią veiklą jau turi būti žinomi visi galimi reikalavimų šaltiniai.

Trm60. Reikalavimų trasavimas (angl. *requirements traceability*) – Procesas, padedantis atsekti bet kurio reikalavimo istoriją. Istorija gali būti peržvelgiama abiem kryptim: tiek atgal, atsekant iš kokių aukštesniųjų lygmenų reikalavimų išsirutuliojo analizuojamas reikalavimas, tiek ir pirmyn, atsekant kokius žemesniųjų lygmenų reikalavimus jis paveikė. Kartais trasavimas gali apimti ir platesni kontekstą, t. y., viena vertus, susieti reikalavimus su jų pirminiais šaltiniais (verslo problemomis, verslo tobulinimo strategijos elementais, verslo nuostatomis, verslo taisyklėmis ir kt.) ir, kita vertus, su tuos reikalavimus įgyvendinančiu kodu bei jų įgyvendinimo laipsnį tikrinančiais testais.

Trm61. Reikalavimų tvarkymas (angl. *requirements management*) - Reikalavimų inžinerijos veikla, skirta reikalavimų duomenų bazei tvarkyti, reikalavimų konfigūracijos valdymo problemoms spręsti, reikalavimų būsenoms fiksuoti ir nuolat aktualizuoti reikalavimus. Ši veikla yra vykdoma lygiagrečiai su kitomis reikalavimų inžinerijos veiklomis ir tėsiasi visą projekto vykdymo laiką.

Trm62. Reikalavimų vertinimas (angl. *requirements validation*) – Reikalavimų inžinerijos veikla, kurią siekiama įvertinti suformuluotų sistemos reikalavimų atitikimą realiems verslo poreikiams. Vertinami visų lygmenų – verslo, vartotojo, informacinės sistemos ir programų sistemos – reikalavimai.

Trm63. Rinkos strategija (angl. *market strategy*) – Veikimo būdas, kuriuo siekiama įsiskverbti į rinką, joje įsitvirtinti ir sėkmingai konkuruoti su savo varžovais. Verslo strategija konkretizuojama, sukuriant kompleksą integrerotų ir vienas su kitu suderintų tikslingu sprendimų bei veiksmų, skirtų verslo tikslams skirti. Verslo vykdymo strategijos dar yra vadinamos konkuravimo strategijomis. Pagrindinės verslo vykdymo strategijos yra kaštų lyderio strategija (siekiama turėti kuo mažesnius gamybos bei pardavimo kaštus ir pardavinėti gaminius bei paslaugas pačiomis mažiausiomis kainomis), išskirtinumo strategija (siekiama teikti gaminius bei paslaugas, turinčius kokias nors išskirtines, vartotojų ypač vertinamas savybes) ir sutelkties strategija (visas dėmesys sutelkiamas į vieną ar kelias specifines gaminių ar paslaugų pirkėjų kategorijas). Šios strategijos yra siejamos su amerikiečių ekonomisto Michael Porter vardu ir dažnai yra vadinamos Porterio strategijomis. Strategijos yra lygiavertės, nei viena nėra geresnė už kitas. Dažniausiai jos yra derinamos viena su kita ir pasirenkamas tam tikras šių strategijų derinys.

Trm64. Ryšio sistema (angl. *communication system*) – informacijos perdavimo sistema, užtikrinanti pakankamą perdavimo proceso patikimumą ir efektyvumą bei gaunamos informacijos korektiškumą. Ryšio sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Pavyzdžiai: tradicinė pašto sistema; telefonų tinklas; kompiuterinio pašto sistemos; tradiciniai pasitarimai;

videokonferencijos sistemos. Pavyzdžiai: Pigeon Communication System, IBM Vocera Solution, CallTalk, EZ Keys XP.

Trm65. Sistemos aplinka (angl. *system environment, operating environment*)

– Visuma daiktų, esančių sistemos išorėje ir arba ją veikiančių, arba jos veikiamų.

Trm66. Sistemos architektūra (angl. *system architecture*) – Sistemos komponentų organizavimo būdas, užtikrinantis sistemos reikalavimų įgyvendinimą ir sistemos sąveiką su jos aplinka.

Trm67. Sistemos dalykinė sritis (angl. *application domain*) – Verslo ar kita gyvenimo sritis, kurioje planuojama naudoti kuriamą sistemą.

Trm68. Sistemos interesų sritis (angl. *domain*) – Išorinio pasaulio dalis, iškaitant išorinius ir vidinius sistemos naudotojus, kuri arba daro poveikį sistemai arba pati yra jos veikiamā.

Trm69. Sistemos interfeisas (angl. *system interface*) – Visuma priemonių, naudojamų sąveikai su sistema realizuoti. Interfeiso elementų negalima priskirti nei sistemai, nei jos aplinkai. Jie yra ant ribos. Interfeisas apima ne tik konkrečias priemones (pvz., meniu), bet ir taisykles, kaip reikia tomis priemonėmis naudotis (t. y. sąveikos su sistema protokolus). Svarbu suprasti, kad interfeisus turi ne tik programą, bet ir bet kurios kitos dirbtinės sistemos, iškaitant verslo ir informacines sistemas. Be to, sistema turi ne tik naudotojo, bet ir kitokius interfeisus, pavyzdžiui, sąveikai su įrenginiais ar procesais.

Trm70. Sistemos komponentas (angl. *component*) – Terminas vartojamas dviem prasmėmis. Pirmają prasme – tai bet kuri sistemos sudėtinė dalis (sandras). Antrąja prasme – tai tokia sistemos dalis, kuri inkapsuliuoja tam tikrą sistemos funkcionalumo dalį ir leidžia naudotis tuo funkcionalumu tik prašant per kokiai nors standartais standartizuotus interfeisus atitinkamų paslaugų. Be to tokiam komponentui negalima keisti esamų interfeisių ar jų pašalinti, jie turi veikti visą komponento gyvavimo ciklą (tai vadina komponento kontraktiniai įsipareigojimai). Papildyti komponentą naujais interfeisais ir keisti jo realizaciją yra leidžiama. Su komponentu yra siejami ne tik tie interfeisai per kuriuos galima gauti jo teikiamas paslaugas, bet ir interfeisai per kuriuos jis gauna jam reikalingas kitų komponentų teikiamas paslaugas. Jokiais kitais būdais naudotis sistemos galimybėmis komponentas negali. Trumpai tariant, antrąja prasme komponentais vadinamos pramoninio gaminio reikalavimus tenkinančios sistemos dalys, kurios dažniausiai yra perkamos gatavos. Komponentai, tiek pirmaja, tiek ir antrąja prasme, patys gali būti sudaryti iš kitų komponentų, t. y. gali būti sistemomis. Vidinių komponentų neturintys komponentai vadinami elementariaisiais, atomariniais arba primityviaisiais komponentais.

Trm71. Sistemos kontekstas (angl. *system context*) – Tie kuriamos sistemos aplinkos elementai, nuo kurių, kaip manoma, priklauso projekto sėkmė.

Trm72. Sistemos probleminė sritis (angl. *problem domain*) – Tikslinė sistemos paskirtis, visuma jos sprendžiamų uždavinių.

Trm73. Sistemos prototipas (angl. *prototype*) - Toks sistemos maketas, kuris palaipsniui yra perdaromas į reikalaujamą sistemą.

Trm74. Sistemos reikalavimai (angl. *system requirements*) – detalus sistemos teikiamų paslaugų ir jos tenkinamų ribojimų aprašas. Šioje knygoje terminas *sistemos reikalavimai* vartojamas kaip termino *informaciniés*

sistemos reikalavimai trumpinys. Tačiau bendruoju atveju gali būti kalbama apie bet kokią sistemą, kurios komponentu yra kuriamoji programų sistema. Sistemos reikalavimai aprašo ne tik tos sistemos programinės įrangos, bet ir kitų jos komponentų reikalavimus. Paprastai jie yra aprašomi dokumente „Sistemos reikalavimų specifikacija“ <Trm75>.

Trm75. Sistemos reikalavimų specifikacija (angl. *system specification*) – Dokumentas, skirtas sistemos reikalavimams <Trm74> aprašyti.

Trm76. „Smegenų šтурmas“ (angl. *brainstorming*) – Grupinis idėjų generavimo, tikslinimo ir tolesnio vystymo būdas. Grindžiamas bendra diskusija kokiui nors klausimu. Kiekvienas diskusijos dalyvis gali išsakyti viską, kas jam ateina į galvą. Manoma, kad juo didesnis skaičius idėjų bus išsakytas, tuo didesnė tikimybė, kad tarp jų bus bent viena vertinga idėja. Be to, turint daug idėjų, galbūt ką nors vertingo galima gauti kombinuojant jas vieną su kita. Todėl nei viena išsakyta idėja néra atmetama. Ji turi būti apsvarstyta ir visapusiškai įvertinta, kad ir kaip keista ir netgi nesusijusi su svarstomu klausimu ji atrodo iš pirmo žvilgsnio. Tačiau nei viena išsakyta idėja nepradedama svarstyti, kritikuoti ir vertinti, pakol visi neišsako savo idėjų. „Smegenų šтурmas“ grindžiamas prielaida, kad žmonės negeba generuoti ir svarstyti idėjas tuo pat metu. Todėl išsakomos idėjos yra tik protokoluojamos, Protokolavimas turi taip vykti, kad diskusijos dalyviai nebūtų stabdomi ir jo net nepajustu, pavyzdžiui, darant garso įrašą. Viena iš svarbiausių šio metodo idėjų yra ta, kad idėjomis turi būti keičiamasi labai greitai, „pagaunant“ ir pratesiant kito išsakyta mintį, diskusija turi vykti spontaniškai. Paprastai, idėjų išsakymui skiriama ne daugiau kaip valanda, nes po to žmonės pradeda idėjas dirbtinai „traukti iš savęs“. Patį terminą „smegenų šтурmas“ dvidešimtojo amžiaus ketvirtuojo dešimtmečio pabaigoje sugalvojo amerikiečių reklamos bendrovės BBDO įkūrėjas Alex Faickney Osborn. Jis buvo ir pirmasis šio metodo populiarizatorius, aprašės jį savo knygoje „Taikomoji vaizduotė“ [85]. Yra įvairių šio metodo variacijų, pavyzdžiui, galima surengti „smegenų šurmo“ sesiją kompiuteriniu paštu.

Trm77. Sprendimus priimti padedanti sistema (angl. *decision support system*) – sistema, skirta duomenims apie buvusias transakcijas agreguoti, apibendrinti, apdoroti ir pateikti sprendimus priimančiam asmeniui patogia forma. Sprendimus priimti padedančios sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Pavyzdžiai: WebFOCUS, Vanguard Studio, FacilitatePro™.

Trm78. Stebėjimo sistema (angl. *monitoring system*) – sistema, skirta reguliariai kaupti ir analizuoti duomenis apie kokio nors proceso, objekto ar kitokio reiškinio būklę ar vyksmą. Paprastai duomenis yra gaunami automatiškai paimant stebimų parametru reikšmes. Stebėjimo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Pavyzdžiai: eismo stebėjimo sistema, krovinių pervežimo stebėjimo sistema, geriamo vandens kokybės stebėjimo sistema, miesto taršos stebėjimo sistema

Trm79. Turinio tvarkymo sistema (angl. *content management system*) – sistema, skirta kokiaje nors informacijos saugykloje saugomiems informaciniams ištekliams kurti, modifikuoti, peržiūrėti ir naikinti. Numato saugykloje saugomos informacijos publikavimo, formato keitimo,

pokyčių kontrolės indeksavimo bei paieškos galimybes. Turinio tvarkymo sistemos gali būti nekompiuterizuotos arba kompiuterizuotos. Pavyzdžiai: Ariadne, atvirojo kodo turinio tvarkymo sistema (HTML redagavimo įrankis, foto albumas, grupinis kalendorius, adresų knyga, internetiniai dienoraščiai); Alfresco, korporacijos turinio valdymo sistema (turinio saugykla ir portalas); BBlog, sistema skirta dokumentams, visų pirma, internetiniams dienoraščiams (angl. blog), publikuoti internte; Epiware, dokumentų tvarkymo sistema (dokumentų įrašymas, tvarkymas, paieška).

Trm80. Transakcijų apdorojimo sistema (angl. *transaction processing system*) – sistema, renkanti duomenis apie vykdomas verslo transakcijas ir galbūt kontroliuojanti sprendimus, priimamus vykdant tas transakcijas. Verslo transakcijomis vadinami veiksmai, keičiantys kokių nors verslo objektą būseną. Pavyzdžiui, rezervuojant vietą viešbutyje, pakinta rezervuoto kambario statusas, jis tampa rezervuotu. Pasinaudojus banko kortele, pakinta asmeninės sąskaitos būsena, joje lieka mažiau pinigų. Kadangi verslo transakcijos visuomet yra registruojamos, tai reiškia, kad, atliekant verslo transakciją, turi būti keičiamama atitinkamų informacinių objektų būsena. Kompiuterinėse sistemoje, kuriose informacinių objektai yra saugomi duomenų bazėse, tų objektų būsenos keitimai yra vadinami duomenų bazių transakcijomis. Šis terminas gali būti išplėstas apimant ir nekompiuterizuotas duomenų bazes, pavyzdžiui, kokias nors verslo transakcijų registravimo knygas. Transakcijų apdorojimo sistemas skirtos duomenų bazių transakcijoms atlkti. Transakcijų apdorojimo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Pavyzdžiai: viešbučių rezervavimo sistema (pvz., iMagic Hotel Reservation, BugHotel Reservation System), banko kortelių aptarnavimo sistema (pvz., MultiXPac, ACI Card Management System™).

Trm81. Užduotis (angl. *use case*) – Tai, ką galima paliepti vykdyti programų, informacinei, verslo ar kokiai nors kitai sistemių. Šioje knygoje skiriamos verslo užduotys, informacinės sistemos užduotys ir programų sistemų užduotys. Jos yra formuluojamos atitinkamos sistemos terminais (t. y.). Užduotys realizuojamos sistemos transakcijomis (apie verslo transakcijas žr. <Trm90>). Todėl kartais sistemos užduotys ir transakcijos yra tapatinamos. Vietomis taip yra daroma ir šioje knygoje, tačiau, griežtai kalbant, toks sutapatinimas yra neteisingas. Užduotis nurodo, ką reikia daryti, o transakciją sudaro užduotį vykdantys veiksmai. Be to, reikėtų skirti tam tikrų užduočių klasės ir konkrečias užduotis, t. y. konkrečius tų klasių egzempliorius. Lygiai taip pat reikia skirti transakcijų klasės ir jų egzempliorius. Leistinos užduočių klasės apibūdina sistemos funkcionalumą arba, kitaip tariant, *galimus sistemas panaudojimo būdus*. Anglų kalbos terminas *use case* ir yra vartojuamas būtent šia prasme. Jį 1992 m. pasiūlė Ivar Jacobson [55]. Iki tol buvo vartojuamas terminas *task* <Trm82>. Šio termino buvo atsisakyta, nes anglų kalboje jis turi daug skirtinę prasmę ir nenorėta suteikti jam dar vienos prasmės. Lietuvių kalboje tiksliausias būtų terminas *galimas sistemas panaudojimo būdas*. Tiktų ir terminas *sistemos vykdomų užduočių klasė*. Tačiau abu šie terminai yra labai ilgi ir nepatogūs vartoti. Todėl šioje knygoje pasirinktas terminas *užduotis*. Kartais lietuvių kalboje vartojuamas terminas *panaudojimo atvejis*, mano nuomone yra nevartotinas, nes, viena vertus, tai yra tiesioginis vertinys ir neatspindi dalyko esmės ir, kita vertus, jo

vartojimas apsunkina analitikų ir dalykinės srities specialistų bendravimą, nes pastarieji nesupranta apie ką eina kalba. Jacobson pasiūlytas būdas nusakyti sistemų funkcionalumą aprašant jų vykdomas užduotis šiuo metu tai yra viena iš populiariausių reikalavimų analizės metodų [13]. Nors dažniausiai jis naudojamas programų sistemų funkcionalumui aprašyti, tačiau puikiai tinka bet kokios prigimties sistemų, įskaitant verslo ir informacines sistemas, funkcionalumui aprašyti. Ivar Jacobson išplėtojo šią idėją 1994 m. parašytoje knygoje *The Object Advantage: Business Process Re-engineering with Object Technology* [56].

Trm82. Užduotis (angl. *task*) – Kas nors, ką reikia padaryti, vykdant projektą ar kokį nors verslo procesą. Tapatinamos su elementariomis verslo proceso operacijomis arba su smulkiausiais projekto darbais.

Trm83. Užklausų sistema (angl. *query processing system*) – sistema, išrenkanti iš kokios nors informacijos saugyklos visus informacinius objektus, tenkinančius nurodytus paieškos kriterijus. Užklausų sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Pavyzdžiai: TinyDB, užklausų sensorių tinklams sistema (užklausos formuluojamos praplėsta SQL kalba); Eddies, užklausų federatyvinėms duomenų bazėms sistema (užklausos formuluojamos SQL kalba); Volcano, mokomoji užklausų duomenų bazėms sistema (užklausos formuluojamos SQL kalba); DIASPORA, išskirstyta užklausų Web serveriams sistema (užklausos formuluojamos specialia deklaratyviaja kalba); Ginga, užklausų išskirstytoms internetinėms duomenų bazėms sistemai.

Trm84. Valdymo sistema (angl. *control system*) – techninė sistema, skirta kokiems nors įrenginiams, kokiems nors, dažniausiai, technologiniams procesams arba kokioms nors techninėms sistemoms valdyti. Valdymo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Esant nekompiuterizuotai valdymo sistemių valdymą gali atliliki operatorius (rankinis valdymas), tačiau dažniausiai tą daro kokia nors aparatūra. Šiandien beveik visos valdymo sistemos esti bent jau iš dalies kompiuterizuotos. Pavyzdžiai: EPICS, moksliniams instrumentams (greitintuvams, teleskopams ir kt.) valdyti skirta sistema; SCADA, cheminiams ir transporto procesams valdyti skirtų sistemų specialios architektūros sistemų kategorija; Ch Control System Toolkit, laisvai platinama instrumentinė sistema, skirta tolydaus ir diskretinio laiko valdymo sistemoms kurti.

Trm85. Vartotojo reikalavimai (angl. *user requirements*) – Reikalavimai, aprašantys sistemą dalykinės srities specialistų požiūriu. Vartotojo reikalavimai aprašo verslo sistemos procesus, informacines, skaičiuojamąsias, komunikavimo ir galbūt kokias nors kitas paslaugas, kurias privalo teikti dalykinės srities specialistams verslo sistemą palaikanti sistema, verslo objektų informaciniu modeliavimu būdus, verslo paslapčių apsaugos reikalavimus, kvalifikacinius dalykinės srities specialistų reikalavimus, jų darbo vietų reikalavimus ir verslo užduočių našumo reikalavimus. Kitaip tariant, jie nusako vartotojų operacinius poreikius <Trm34>. Paprastai vartotojo reikalavimai yra aprašomi dokumente „Operacinių poreikių specifikacija“ <Trm35>.

Trm86. Verslo nuostata (angl. *business policy*) – nurodymas ar nurodymų rinkinys, kuriuo privalu vadovautis visiems organizacijos darbuotojams,

pavyzdžiui, "Su klientu visada reikia elgtis mandagiai ir pagarbiai" arba "Nusiskundimus ir skundus reikia nagrinėti išsamiai ir skubos tvarka". Verslo nuostatos turi būti detalizuojamos konkrečiomis verslo taisyklėmis, t. y. darbo procedūromis, aprašytomis darbuotojų pareigybiniėse instrukcijose. Tačiau daugelyje organizacijų verslo nuostatos arba apskritai nėra suformuluotos išreikštiniu būdu, arba jos nėra nuleistos į pareigybinių instrukcijų lygmenį, arba nėra griežtai laikomasi pareigybinių instrukcijų. Todėl analitikui tenka aiškintis tikrąsias verslo nuostatas ir taisykles kitais būdais. Be to, kartais, ypač vykdant nesąžiningą verslą, yra vadovaujamasi visiems darbuotojams žinomomis, bet oficialiai nepripažystamomis verslo nuostatomis, pavyzdžiui, "Stenkis parduoti pasenusias prekes" ar "Lentynoje nurodyk mažesnę prekės kainą, negu už ją teks mokėti kasoje". Programų sistemų inžinierių etikos kodeksas reikalauja, kad programų sistemose nebūtų realizuojamos nesąžiningo verslo nuostatos bei taisyklos.

Trm87. Verslo reikalavimai (angl. *business requirements*) – Reikalavimai, aprašantys kokia turi būti verslo sistema. Verslo reikalavimai aprašo verslo sistemos misiją, rinkos strategiją, verslo vykdymo strategiją, verslo sistemos viziją, tai vizijai įgyvendinti skirtą tikslų medį, verslo nuostatas, verslo taisykles, verslo objektus, verslo dalyvių įgaliojimus. Jie taip pat susieja verslo tikslus su darbo vietomis ir nustato verslo našumo reikalavimus. Paprastai verslo reikalavimai yra aprašomi dokumente „Verslo poreikių specifikacija“<Trm88>.

Trm88. Verslo poreikių specifikacija (angl. *business requirements, vision and scope document, analysis document*) – Dokumentas, skirtas verslo lygmens reikalavimams <Trm87> aprašyti.

Trm89. Verslo taisyklės (angl. *business rules*) – Verslo taisyklės yra suprantamos labai įvairiai. Šiame vadovėlyje jos apima bendrasias verslo nuostatas, verslą reguliuojančią teisės aktų nuostatas, verslo naudojamų standartų nuostatas, verslo objektų ribojimus, įvykių, situacijų bei iniciatyvų apdorojimo ir nurodymų (direktyvų) vykdymo laiko ribojimus.

Trm90. Verslo transakcija (angl. *business transaction*) – Verslo sistemos veiksmai, atliekami vykdant kokią nors verslo užduotį <Trm81>. Tais veiksmais klientui yra parduodama kokia nors prekė arba suteikiama kokia nors paslauga, pavyzdžiui, rezervuojančios viešbutis, parduodamas bilietas į teatrą, išduodama pažymą apie gautas pajamas, suteikiama paskola, patikrinama sveikata ir t.t.

Trm91. Verslo vykdymo strategija (angl. *business strategy*) – būdas, kaip panaudojant esamas verslo kompetencijas ir jo turimus išteklius, panaudoti neišnaudotas verslo galimybes, sustiprinti silpnąsias verslo puses, pašalinti esamus verslo sistemos trūkumus ir šitaip išspręsti esamas verslo problemas ir išvengti jam grėsiančių grėsmių. Verslo tobulinimo strategijos dar yra vadinamos korporacijos lygmens strategijomis, nes jos nusako kaip pertvarkyti visos organizacijos darbą. Skiriamos trys pagrindiniai korporacijos lygmens strategijų tipai: verslo plėtros strategijos, stabilaus verslo strategijos ir verslo likvidavimo ar atnaujinimo strategijos. Verslas gali būti plečiamas keliais būdais: sutelkiant visą dėmesį sėkmingiausiai paslaugų ar gaminių rūšiai ir didinant jų pateikties rinkai apimtis; perimant jeigos logistikos, išeigos logistikos ar abiejų jų kontrolę, pavyzdžiui, įsigyjant reikiamas žaliavas gaminančias gamyklas arba/ir pagamintus gaminius pardavinėjantį prekybos tinklą (tai vadinama

vertikaliaja verslo integracija); perimant varžovų kontroliuojamas įmones (tai vadinama horizontaliaja verslo integracija), skverbiantis į naujas rinkas (koncentrinė plėtra) arba į naujas veiklos sritis (konglomeratinė plėtra). Verslo plėtra taip pat gali būti vykdoma jį internalizuojant arba decentralizuojant. Stabilaus verslo strategijos nenumato jokių šuolių. Jomis siekiama išlaikyti esamas verslo operacijų apimtis. Stabilaus augimo strategijos (tarkime, 5% per metus) taip pat yra priskiriamos prie stabilaus verslo strategijų. Verslo likvidavimo strategijoms priskiriamos strategijos, numatančios ilgą laiką gamintų gaminių ar teiktų paslaugų keitimą naujais gaminiais ar paslaugomis (pasitraukimo strategija), palaipsni perėjimą iš vieno verslo į kitą (žaidimo pabaigos strategija), verslo pardavimą tikslu įsigyti naują verslą (pasipelnymo strategija) ar bankrotą. Verslo atnaujinimo strategijoms priskiriamos strategijos, numatančios kardinalų išlaidų mažinimą, verslo restruktūrizavimą, verslo dalies likvidavimą ar pardavimą, kardinalų verslo pertvarkymą (verslo reinžinerija) ar esminį laipsnišką jo tobulinimą.

Trm92. Verslo tobulinimo strategija (angl. *business improvement strategy*) – būdas, kaip remiantis esama verslo kompetencija ir jo turimais ištekliais, panaudoti neišnaudotas verslo galimybes, sustiprinti silpnąsias verslo puses, pašalinti esamus verslo sistemos trūkumus ir šitaip išspręsti esamas verslo problemas bei išvengti jam gręsiančių gręsmių. Verslo tobulinimo strategija gali būti naudojama tiek atliekant vienkartinį (revoliucinį) verslo pertvarkymą (jo reinžineriją), tiek ir tobulinant jį palaipsniui. Verslo tobulinimo strategija yra detalizuojama siekiamų tikslų medžiu. Tikslų medis lokalizuojama siekiamus tikslus funkciniuose organizacijos padaliniuose ir nuleidžia juos į funkcinį lygmenį, t. y. išreiškia funkcinio lygmens strategijomis.

Trm93. Vertikalusis maketas (angl. *vertical prototype*) – Maketas <Trm30>, kuris yra kuriamas ne interfeiso, bet ir kokių nors kitų funkciinių ar nefunkcinių reikalavimų įgyvendinamumui analizuoti arba šiemis reikalavimams patikslinti ir papildyti. Tokie maketai vadinami vertikaliaisiai, nes jie realizuojama tam tikrą visus architektūros sluoksnius apimančią programų sistemos išpjovą. Vertikaliesiems maktams kurti specialiai pritaikytų maketavimo priemonių nėra. Todėl tokius maktus tenka kurti rankiniu būdu.

Trm94. Žinių tvarkymo sistema (angl. *knowledge management system*) – speciali išteklių tvarkymo sistemų rūšis. Sistema kaupia organizacijoje naudojamas žinias, jas sistemina, tvarko ir suteikia galimybes naudotis jomis kolektyviai. Be kita ko, žinių tvarkymo sistemos padeda sužinoti, kas organizacijoje yra ekspertai vienais ar kitais jos vykdomo verslo klausimais. Žinių tvarkymo sistemos gali būti nekompiuterizuotos, iš dalies kompiuterizuotos arba visiškai kompiuterizuotos. Terminas žinių tvarkymo sistema pradėtas vartoti tik pradėjus kurti kompiuterizuotas žinių tvarkymo sistemas ir kol kas jo turinys nėra griežtai apibrėžtas. Nekompiuterizuotos žinių tvarkymo sistemos praktiškai nebuvvo kuriamos, nes, iki įsigalint žiniomis grindžiamai ekonomikai, jų poreikis buvo labai nedidelis. Pavyzdžiai: ISYS, korporacijoms skirta paieškos mašina, atliekanti paiešką intranete, interneto tinklalapiuose ir portaluose; RightNow®, korporacijų žinių bazėms kurti ir prieigai (įskaitant balso technologijas) prie tokų bazių skirta įranga; SearchInform, paieškos

mašina skirta dokumentų tekstu paieškai korporaciniuose tinkluose; DEKSI KM, korporacijos žinių tvarkymo sistema; powerKNOW, korporacijos žinių tvarkymo sistema (žinių saugojimas, sklaida ir paieška).