VILNIAUS UNIVERSITETAS

INFORMATIKOS IR MATEMATIKOS FAKULTETAS

KOMPIUTERIJOS KATEDRA

Baigiamasis magistro darbas

Viešosios informacijos ir transporto portalo kūrimo ir testavimo metodika

Atliko: 2 kurso studentas

Aurimas Lacitis

Darbo vadovas:

doc. Algimantas Juozapavičius

Vilnius
2006

Turinys

1. SUTARTINIŲ TERMINŲ SĄRAŠAS

Sąvoka	**Aprašymas**
GPS	Pasaulinė pozicionavimo sistema (Global Positioning System)
GSM	Korinio ryšio tinklas (Global System for Mobile Communications)
LAS	Lokacijos serveris (Location Application Server)
Oracle CDM	Oracle informacinių sistemų kūrimo metodika (Oracle custom development method)
TVIMP	Transporto Viešosios Informacijos Mobilus Portalas
TIP	Transporto Informacijos Portalas
VITMOS IS	VITMOS informacinė sistema
VITMOS projektas	VITMOS projektas, kurio vykdymo metu yra kuriama VITMOS IS
WLAN	Bevielis lokaliosios aprėpties tinklas (Wireless Local Area Network)

2. ANOTACIJA

Baigiamajame magistro darbe nagrinėjamos ir aprašomos viešosios informacijos ir transporto portalų kūrimo ir testavimo metodikos. Metodikos aprėpia visą portalo kūrimo ciklą t.y. analitinį procesą, siekiant apibrėžti funkcionalumą bei parinkti tinkamą transporto priemonių buvimo vietos nustatymo technologiją, portalo projektavimo, programavimo ir testavimo procesus paremtus moderniomis instrumentinėmis priemonėmis. Remiantis išnagrinėtomis metodikomis atliktas praktinis darbas – sukurtos tam tikros viešosios informacijos ir transporto portalo funkcijos, kurių viena iš pagrindinių yra transporto priemonės buvimo vietos nustatymas ir pavaizdavimas. Praktinėje dalyje aprašytos sukurtos funkcijos, vidiniai informaciniai srautai, sukurtų priemonių testavimo scenarijai, pagal kuriuos testuojamas portalas ir nustatomos jo veikimo kritinės vietos. Darbo metu yra suprogramuota dalis portalo funkcijų, LAS imitatorius, duomenų mainai tarp portalo ir LAS. Praktinis darbas parodo, kaip naudojant tinkamas metodikas išsprendžiamos transporto portalų, o kartu ir informacinių sistemų kūrimo, kokybės įvertinimo ir perdavimo užsakovui problemos.

Išnagrinėjus teorinius ir praktinius informacinių sistemų kūrimo ir testavimo aspektus, atlikus objektų buvimo vietos nustatymo technologijų lyginamąją analizę, pateiktos baigiamojo darbo išvados ir siūlymai.

3. SUMMARY

Development and testing methods of public information and transport portals are studied and described in the final master work. The methods range whole cycle of the portal development including analysis, which lets you to define a functionality and choose right technology of determination of a transport vehicle position, design and generation, which include testing process. These processes are maintained by modern means. The practical work - developed public information and transport portal, which has main function to determine and show a position of a transport vehicle, is made and tested according to the studied methods. There are described developed functions, internal information flows, testing scenarios, which are used to test the portal and to find the most critical it's working places, in the practical work. There is developed some part of the portal functionality, LAS imitator, data exchange between the portal and LAS. The practical work shows how development, quality rating and system acceptance problems can be solved if system is developed using right information system including information and transport portals development and testing methods.

There are given the conclusion and offerings of the final work after studying theoretical and practical information systems development and testing aspects, making the analysis of different technology of determination of a transport vehicle position.

4. ĮVADAS

4.1 MAGISTRO DARBO APRAŠYMAS

Magistro darbas yra VITMOS projekto dalis, kuriame dalyvauja VU, KTU, VGTU ir Bitė GSM.

VITMOS – viešosios informacijos ir transporto mobilių sprendimų projektas. Projekto programoje numatyta sukurti krovinių ir transporto kompiuterizuoto registravimo bei sekimo sistemą, o taip pat interneto vartus (portalą) . Išanalizavus pagrindines dar plačiai nenaudojamas mobiliųjų komunikacijų galimybes, tokias kaip - pranešimų perdavimą tiesiogiai asmeniui, o ne į fiksuotą vietą, multiterpės informacijos perdavimo iš duomenų saugyklų į mobilius telefonus ir kt., – padaryta išvada, kad transporto viešosios informacijos tvarkymui ir transporto paslaugų organizavimui tikslinga taikyti mobilias komunikacijas.

Transporto viešosios informacijos mobilus interneto portalas (TVIMP) labai padidins šalies viešųjų institucijų informacinių bazių pasiekiamumą. Sukurtas transporto informacijos portalas (TIP) turėtų įgalinti bet kuriuo laiku nustatyti ir stebėti transporto priemonės ar krovinio vietą bei jų judėjimą Lietuvoje ar už jos ribų. TIP portale bus teikiamos tiek komercinės paslaugos, tiek ir suteiktos galimybės viešam naudojimui. Portalas bus skirtas plačiam naudotojų ratui:

- Organizacijoms, renkančioms statistinę informaciją apie Lietuvos kelių eksploatavimą bei transporto srautus.
- Su transportu ir logistika susijusiomis įmonėmis, kurioms yra svarbu gauti informaciją apie vežamus krovinius, keistis informacija su vežėjais ir užsakovais.
- Įmonėms, kurių darbuotojams tenka dirbti nutolusiuose taškuose (pvz., degalines aptarnaujančios įmonės).
- Individualiems asmenims, kuriems yra svarbu gauti informaciją apie artimųjų/pažįstamų buvimo vietą bei judėjimo eigą.

Dalis portalo funcionalumo bus skirta informacijos apie sekamus mobilius objektus tvarkymui, tačiau nemaža portalo dalis bus visiškai savarankiška:

- Skelbimų skyrius
- Transporto naujienos
- Logistikos įmonių katalogas

Šios papildomos paslaugos turi gana didelę paklausą tarp logistikos srityje dirbančių asmenų, todėl padės pritraukti potencialius komerciniu pagrindu teikiamas paslaugas užsisakančius klientus.

Siekiant atitikti daugumos naudotojų poreikius, mokėjimas už TIP portalo paslaugas bus taikomas keliais būdais:

- už transakcijas;
- už laikotarpį;
- mišriai (už laikotarpį ir už tranzakcijas).

Atsiskaitymo būdai bus orientuoti tiek į verslo atstovus, tiek į fizinius asmenis:

- pagal sutartį banko pavedimu už paslaugų paketą;
- padidinto tarifo SMS žinutėmis už atskiras paslaugas.

Toks lankstumas suteiks galimybę teikti paslaugas plačiam naudotojų ratui - nuo didelių logistikos srityje dirbančių įmonių iki individualių vairuotojų.

Portale pateikiama dviejų tipų informacija: stacionari – pateikiamos nuorodos į kitas Interneto svetaines ir portalus, bei dinaminė – pateikiama informacija iš transporto portalo duomenų bazės, kurią vartotojai gali papildyti, bei redaguoti.

Stacionarios informacijos tikslas yra suteikti kuo daugiau reikalingų žinių apie vieną ar kitą veiksnį įtakojantį kelionę. Ši informacija gali būti pateikta transporto portale Internete bei per mobilųjį telefoną. Pagrindinė stacionarios informacijos paskirtis – pateikti kiek įmanoma išsamesnį ir turtingesnį sąrašą nuorodų, atitinkančių vartotojų

reikalavimus. Šių nuorodų sąrašas sudarytas peržiūrėjus visus transporto informacijos puslapius, paminėtus analizės priede. Stacionarios informacijos naudotojai yra visi portalo lankytojai.

Dinaminė informacija yra surenkama iš portalo lankytojų ir registruotų vartotojų ir gali būti pateikta (ar ribojama) atsižvelgiant į vartotojų reikalavimus: t.y. užsakovams pateikiamas užsiregistravusių vežėjų sąrašas, jų pasiūlymai, galimybės pervežti krovinius, informacija apie jų siūlomas paslaugas, turimą transportą ir pan.

Vienas iš pagrindinių TIP reikalavimų – portalo turinys turi būti nesunkiai redaguojamas bei papildomas nauja medžiaga, meniu punktais bei naujais moduliais. TVS turi savo naudotojų registrą bei atskirą duomenų bazę. Vienas iš didžiausių sistemos privalumų yra tai, kad puslapius galima kurti, išnaudojant programavimo kalbos PHP specifiką. Tai yra išskirtinis bruožas, nes šiuo atveju puslapio turinys gali būti dinamiškas ir automatiškai keistis, pasikeitus duomenims pagal kuriuos jis yra kuriamas. Kitas reikalavimas yra TVS bei viso portalo universalumas duomenų bazių valdymo sistemų atžvilgiu. Sistema bus projektuojama taip, kad ją būtų galima naudoti su įvairiomis DBVS. Saugumo užtikrinimui sistemoje negalima naudoti sukurtų atviro kodo turinio valdymo sistemų komponentų arba branduolių, nes dauguma jų turi saugumo spragų. Nors minėtos saugumo spragos yra pakankamai gerai išnagrinėtos, nes jų kodas yra atviras, dalis spragų nėra ištaisytos.

4.2 DARBO TIKSLAI

Magistro darbo tikslas – išnagrinėti ir aprašyti viešosios informacijos ir transporto portalų kūrimo ir testavimo metodikas, išbandyti praktinį jų taikymą parodant, kaip pagal jas yra kuriami, testuojami transporto portalai bei išsprendžiamos jų, o kartu ir informacinių sistemų kūrimo, kokybės įvertinimo ir perdavimo užsakovui problemos. Šis tikslas yra išskaidytas į smulkesnius tikslus:

- Išnagrinėti ir aprašyti Oracle CDM Fastrack informacinių sistemų kūrimo metodiką;
- Išnagrinėti ir aprašyti programinės įrangos testavimo metodiką;
- Išnagrinėti ir palyginti skirtingas judančių objektų buvimo vietos nustatymo

technologijas: GSM, GPS, WLAN, aprašyti jų panaudojimo galymybes VITMOS sistemoje;

- Remiantis Oracle CDM informacinių sistemų kūrimo metodika specifikuoti, suprojektuoti ir suprogramuoti atskiras VITMOS projekto TVIMP ir TIP portalų ir jų turinio valdymo sistemos (toliau vadinama TVS) funkcijas, suprojektuoti duomenų mainus tarp LAS ir TIP, suikurti LAS imitatorių. *Pastaba. TVIMP ir TIP portalai ir jų TVS priklauso VITMOS informacinei sistemai (toliau vadinama - VITMOS IS), todėl kai kuriose šio dokumento dalyse šios dalys apibendrintai bus vadinamos VITMOS IS.*

- Parengti VITMOS IS sistemos testavimo scenarijus, remiantis geriausia testavimo praktika bei metodika aprašyta pripažintų autorių literatūroje,

- Ištestuoti sukurtą programinę įrangą pagal parengtus testavimo scenarijus;

4.3 DARBO REZULTATAI

Magistro darbo metu buvo pasiektas užsibrėžtas tikslas t.y. išnagrinėtos ir aprašytos pasirinktos metodikos, parodyta, kaip pagal jas kuriami ir testuojami portalai, išsprendžiamos transporto portalų, o kartu ir visų informacinių sistemų kūrimo, kokybės įvertinimo ir perdavimo užsakovui problemos.

Atlikus darbą buvo gauti tokie rezultatai:

- Oracle CDM IS kūrimo metodo aprašymas (Rezultatas pateiktas skyriuje „5.1.2“);

- Testavimo metodikos aprašymas (Rezultatas pateiktas skyriuje „5.1.3“);

- Judančių objektų buvimo vietos nustatymo technologijų: GSM, GPS, WLAN lyginamoji analizė ir panaudojimo kuriant VITMOS IS aprašymas (Rezultatas pateiktas skyriuje „5.1.1“);

- VITMOS IS reikalavimų specifikacija: TVIMP ir TIP funkcijų „Moscow“ sąrašas (Rezultatas pateiktas skyriuje „5.2.2.1“);
  TVIMP ir TIP portalų, jų TVS ir LAS imitatoriaus programinė įranga

(Rezultatas pateiktas atskirame priede):

  - duomenų bazės skriptai,
  - programiniai išeities tekstai.

- TVIMP ir TIP portalų ir jų TVS testavimo scenarijai (Rezultatas pateiktas skyriuje „5.2.3.2“);
- Ištestuota sukurta programinė įranga pagal parengtus testavimo scenarijus;
- Išvados (Rezultatas pateiktas skyriuje „6“).

5. ATLIKTI MAGISTRO DARBO DARBAI

5.1 ATLIKTI TEORINĖS DALIES DARBAI

Magistro darbo kūrimo eigoje buvo atlikti šie teorinės dalies darbai:

- Atliktas objektų buvimo vietos nustatymo technologijų GSM, GPS, WLAN palyginimas, aprašytos jų panaudojimo galymybės VITMOS sistemoje;
- Aprašyti Oracle CDM metodikos esminiai punktai, įvardinta kaip Oracle CDM yra panaudojamas VITMOS projekte;
- Aprašyta testavimo metodika;
- Atlikta su magistro darbe atliekamomis užduotimis susijusios literatūros apžvalga.

5.1.1 GSM, GPS, WLAN lyginamoji analizė

Viena svarbesnių VITMOS IS projekto uždavinių – objekto buvimo vietos nustatymo technologija, jos pasirinkimas. Kiekviena objekto vietos nustatymo technologija yra skirtinga, turi tiek teigiamų, tiek neigiamų savybių. Priklausomai nuo vartotojo poreikių reikalingas tinkamas technologijos parinkimas ir panaudojamas konkrečioje analitinėje užduotyje. Viena iš VITMOS projekte atliekamų užduočių yra nustatyti tam tikros kompanijos transporto priemonių buvimo vietą. Objekto buvimo vietos nustatymui gali būti naudojamos įvairios echnologijos: GSM, GPS, WLAN. Norint pasirinkti vieną iš jų yra reikalinga jų analizė.

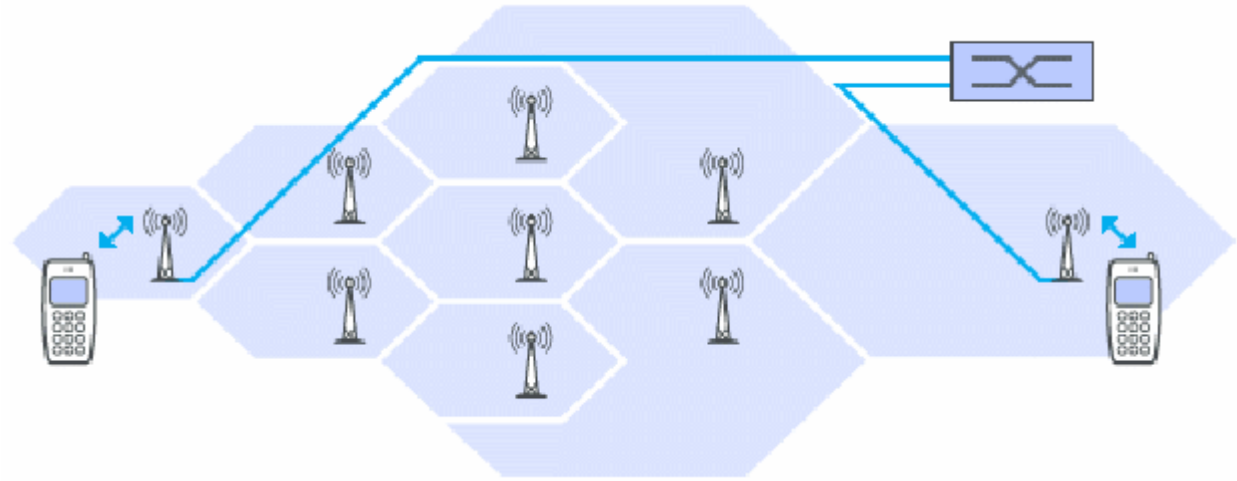
Lyginamosios analizės tikslas – panagrinėti aukščiau minėtas technologijas, nustatyti jų panaudojimo VITMOS informacinėje sistemoje galimybes. Gautas rezultatas – pasirinkta konkreti objekto buvimo vietos nustatymo technologija, tinkama aukščiau minėtai užduočiai atlikti.

5.1.1.1 Duomenų perdavimo technologijų aprašymas

5.1.1.1.1 GSM

GSM – (Global System for Mobile Communication) – globaliai naudojamas komunikavimo mobiliaisiais telefonais standartas. GSM yra šį standartą sukūrusios grupės, įkurtos 1982 metais vardas. Ši grupė buvo įkurta tam, kad sukurti bendrą Europos mobiliųjų telefonų standartą. Kuriant standartą buvo numatyta, kad jis bus naudojamas ir už Europos ribų. GSM turi virš 120 milijonų vartotojų ir yra naudojamas 120 pasaulio šalių.

GSM tinklą sudaro daugybė viena nuo kitos per tam tikrą atstumą nutolusių stočių, kurios priima ir siunčia informaciją mobiliajam įrenginiui tam tikru spinduliu aplink save (6.1.1.1.1 pav.).



Pav. 6.1.1.1.1 – GSM tinklas

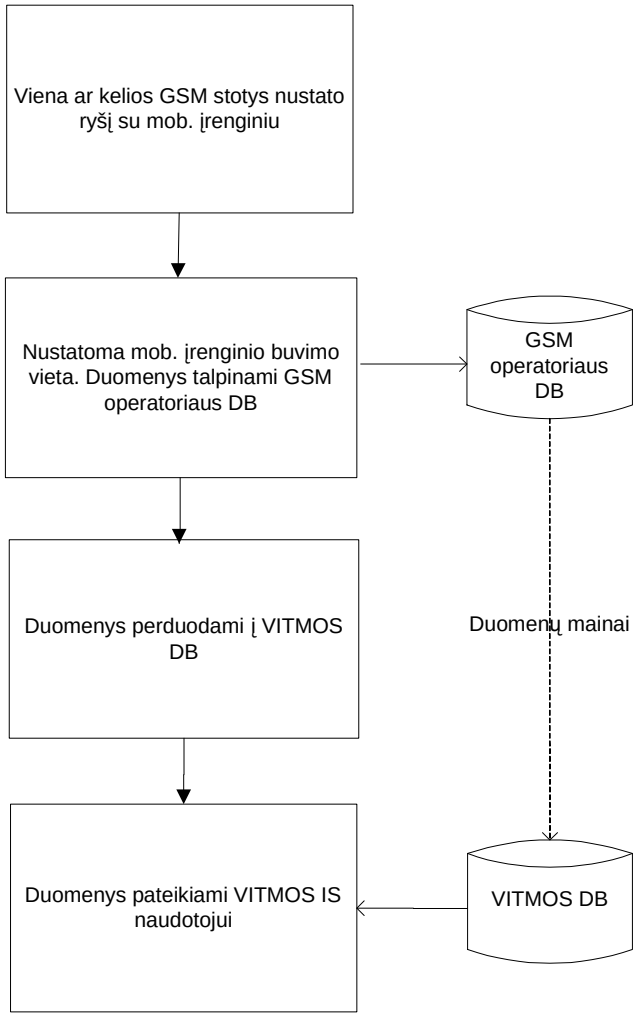
GSM panaudojamas VITMOS projekte.

GSM technologija gali būti panaudota VITMOS projekte nustatant tam tikro objekto buvimo vietą. Kaip ir minėta aukščiau, GSM tinklą sudaro daugybė stočių, prie kurių per tam tikrą atstumą gali prisijungti mobilūs įrenginiai. Mobiliojo įrenginio buvimo vietą galima apytiksliai nustatyti pagal tai, prie kurios stoties yra prisirišęs mobilusis įrenginys. Kadangi mobilusis įrenginys tam tikru momentu gali būti prisirišęs prie kelių stočių, tai padeda tiksliau nustatyti jo buvimo vietą. Naudojant šią

technologiją objekto buvimo vietą geriausiu atveju galima nustatyti keliasdešimties metrų tikslumu.

Žemiau yra pateikta schema, kaip GSM turėtų būti naudojama VITMOS projekto informacinėje sistemoje:

1. Viena arba kelios mobiliojo ryšio stotys tam tikru laiko momentu nustato, kad yra ryšys tarp stoties ir mobiliojo įrenginio, įmontuoto judančiame objekte;
2. Pagal turimus duomenis yra apytiksliai nustatoma mobiliojo įrenginio, kartu ir judančio objekto buvimo vieta tam tikru laiko momentu;
3. Gauti duomenys perduodami VITMOS informacinės sistemos duomenų bazei;
4. Duomenų bazėje apdorojami duomenys ir pateikiami naudotojui.



Pav. 5.1.1.2 GSM panaudojimo VITMOS IS schema

5.1.1.1.2 GPS

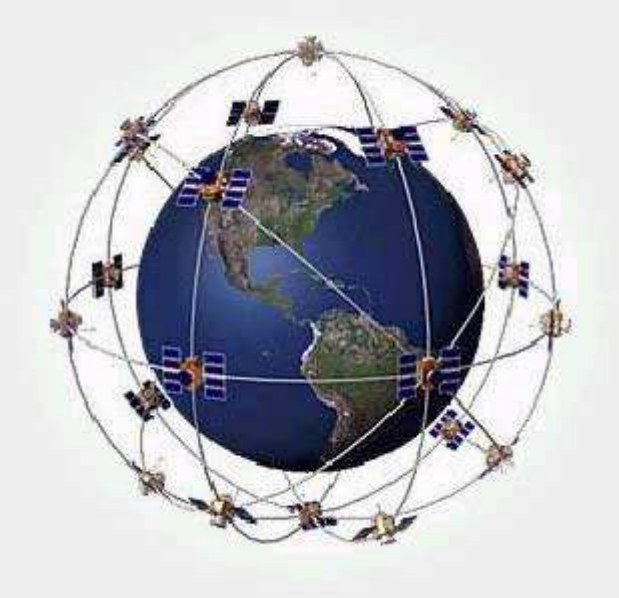
Globali pozicijos nustatymo sistema (GPS - Global Positioning System) buvo pradedama kurti Jungtinių Valstijų Gynybos Departamento dar 1973 metais, siekiant tikslios navigacijos galimybių. Iš pradžių ji buvo vadinama NAVSTAR (Navigation System with Timing and Ranging) ir taip pat buvo naudojama tik kariniais tikslais. Šioje sistemoje 24 Žemės palydovai skrieja šešiomis orbitomis apytiksliai 20000 km aukštyje nuo Žemės centro ir užtikrina pilną mūsų planetos paviršiaus perdengima. Orbitą palydovas įveikia du kartus per 24 valandas. Skriedamas jis pastoviai siunčia radijo signalus su informacija, kada tas signalas buvo išsiųstas ir kitais navigaciniais duomenimis. Palydovu plejada buvo išdėstyta taip, kad iš bet kurio Žemės paviršiaus taško būtų matomi mažiausiai 4 palydovai. GPS imtuvas, esantis automobilyje, lėktuve, laive ar bet kurioje kitoje transporto priemonėje, priima informaciją iš mažiausiai trijų palydovų ir atlikdamas reikalingus skaičiavimus nustato tikslią savo poziciją Žemės paviršiuje. Tam, kad nustatyti pozicija trimatėje erdvėje, reikia priimti signalus bent iš keturių palydovų. Ši sistema pilnai buvo įdiegta jau 1993 metais ir jos realizavimas pareikalavo daugiau nei 15 milijonų JAV dolerių.

GPS sistemą sudaro:

- dirbtinių Žemės palydovų plejada (kosminis segmentas);
- antžeminių sekimo ir valdymo stočių tinklas (valdymo segmentas);
- GPS imtuvas (vartotojo aparatinės priemonės).

Kosminis segmentas

Jį sudaro 26 palydovai (21 pagrindinis ir 5 papildomi), nors kiti šaltiniai teigia, kad net 28, beskriejantys aplink planetą šešiomis orbitomis. Orbitų plokštumos pasvirusios 55° nuo ekvatoriaus plokštumos ir perstumtos viena kitos atžvilgiu 60° pagal ilgumą.



5.1.2.1 pav. – palydovai išsidėstę aplink Žemę.

Orbitų spindulys apie 26 tūkst. km., o apsisukimo periodas - pusė paros (apytiksliai 11 val. 58 min.). Palydove sumontuoti keturi atominiai laikrodžiai, arba kitaip sakant, keturi dažnių standartai, taip pat saulės baterija, orbitų koregavimo varikliai, priėmimo perdavimo aparatūra, kompiuteriai.

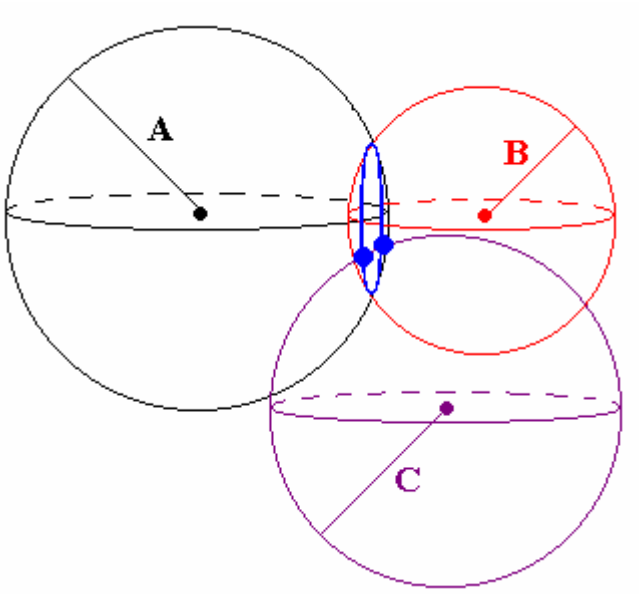
Valdymo segmentas

Jis susideda iš pagrindinės valdymo stoties (Falcon aviacijos bazė, Kolorado valst. JAV), penkių sekimo stočių, išsidėsčiusių taip pat amerikiečių karinėse bazėse įvairiose salose po visą pasaulį ir dar trijų siuntimo stočių. Be to yra visas tinklas valstybinių ir privačių palydovų sekimo stočių, kurios stebi ir tikslina atmosferos ir palydovų judėjimo trajektorijų parametrus. Surenkama informacija apdorojama galingais kompiuteriais ir periodiškai perduodama į palydovus navigacinių pranešimų atnaujinimui ir orbitų koregavimui.

Vartotojų aparatūra

Vartotojo aparatūroje (GPS imtuve) priimamas signalas dekoduojamas. Gautas kodas lyginamas su analogišku kodu, kurį generuoja pats GPS imtuvas. Tai leidžia nustatyti signalo sklidimo iš palydovo trukmę ir taip apskaičiuoti pseudoatstumą. Pagrindinė idėja, GPS imtuvui nustatant koordinates, yra atstumo skaičiavimas iki kelių palydovų, kurių išsidėstymas iš anksto žinomas. Jeigu žinomas atstumas A nuo vieno

palydovo, tai imtuvo koordinačių nustatyti neįmanoma. Jis gali būti bet kuriame sferos su spinduliu A taške. Tegul žinomas atstumas B iki kito palydovo. Tokiu atveju taip pat negalime nustatyti tikslių imtuvo koordinačių, bet galime sakyti, kad jis yra kažkur dviejų sferų susikirtimo apskritime. Atstumas C iki trečio palydovo sumažina neapibrėžtumą iki dviejų taškų. To jau pakaktų nustatant GPS imtuvo koordinates, nes tik vienas iš tų taškų randasi ant Žemės paviršiaus arba arti jo, o antras taškas, melagingas, bus arba gyliai Žemėje, arba labai aukštai virš jos paviršiaus. Taigi, teoriškai trimatei navigacijai užtenka žinoti atstumus tik iki trijų palydovų.



5.1.2.2 pav. GPS imtuvo buvimo vietos nustatymo principas.

GPS panaudojimas VITMOS projekte.

Ši technologija taip pat gali būti sėkmingai panaudota VITMOS projekte nustatant judančio objekto buvimo vietą. Kadangi GPS leidžia nustatyti objekto buvimo vietą labai tiksliai (iki kelių metrų tikslumu), šią technologiją tikslinga naudoti ten, kur svarbus itin tikslus objekto buvimo vietos nustatymas.

Žemiau yra pateikta schema, kaip GPS turėtų būti naudojama VITMOS projekto informacinėje sistemoje:

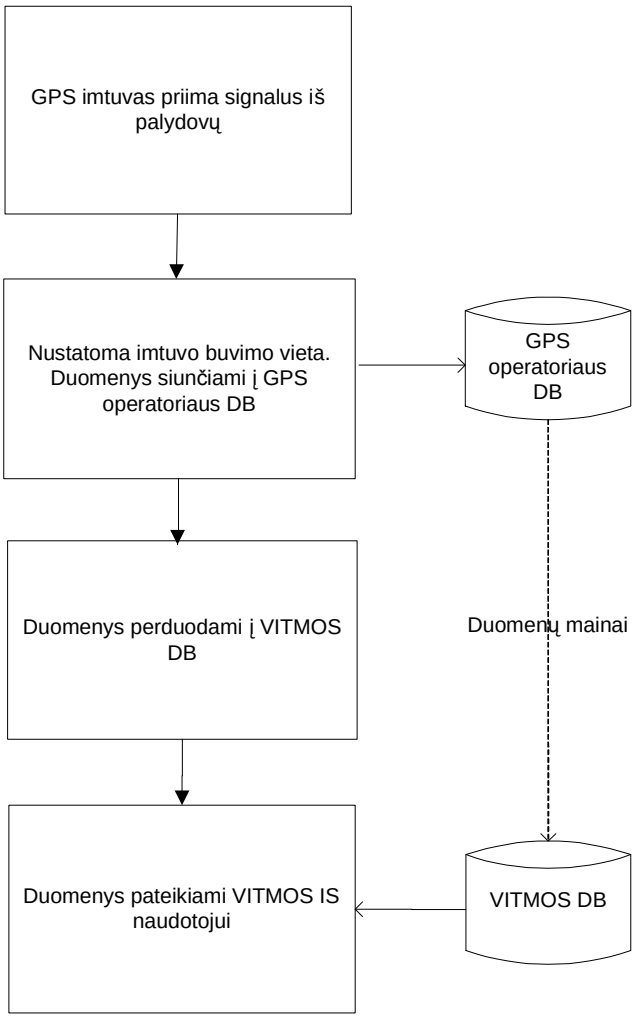
1. judančiame objekte esantis imtuvas priima iš palydomų signalus;

2. prie imtuvo įmontuotas mini kompiuteris juos apdoroja, nustato imtuvo buvimo vietą - koordinates.

3. turimus duomenis išsiunčia į centrinį serverį – VITMOS IS duomenų bazės serverį;

4. duomenų bazės serveryje papildomai apdorojami duomenys;

5. duomenys pateikiami vartotojui.



Pav. 5.1.2.3 GPS panaudojimo VITMOS IS schema

5.1.1.1.3 WLAN – bevieliai vietiniai tinklai

WLAN – bevieliai vietiniai tinklai yra dar viena technologija arba būdas, kuris gali leidžia nustatyti tam tikro judančio objekto buvimo vietą.

WLAN (Wireless Local Area Network) tinklu panaudojimo sritys:

WLAN įranga naudojama verslo įmonių, privačiuose bei viešuose tinkluose informacijai tarp tinklo įrenginių perduoti. Tipinės WLAN taikymų vietos bei sąlygos:

1. Vietos, kur sunku įrengti laidinį ryšį (istoriniai pastatai, laikini statiniai);
2. Laikini tinklai (avarijų likviduotojų tinklai, laikini intarpai vietoj sugedusių stacionarių);
3. Mobilios darbo vietos – ofisai;
4. Nuolatiniai kompiuterių tinklai, kuriuose WLAN įrengiami tam tikram komfortui sukurti;

Ryšio taškuose (*angl. hot spot*) kaip bevielio interneto ryšio zonos, kur daug vartotojų ir jie nuolat kinta (oro uostai, viešbučiai, parodos, konferencijų salės, universitetai).

Tokių tinklų nėra daug, tačiau jų vis daugėja. Didėjant tokių tinklų kiekiui juos sėkmingai galima panaudoti pozicionavimo sistemose tokiose kaip VITMOS.

Pav. 5.1.3.1 – bevielis vietinis tinklas

WLAN panaudojimas VITMOS projekte.

WLAN VITMOS projekte gali būti panaudotas tokiu principu. Judančiame objekte gali būti įmontuotas įrenginys turintis galimybę prisijungti prie bevielio tinklo. Įrenginys turėtų būti sukonfigūruotas taip, kad:

- Jei nėra prisijungęs prie jokio tinklo, bandytų prisijungti (prie tinklo DHCP serverio);

- Jei yra prisijungęs prie tinklo, tam tikrais laiko intervalais internetu jungtųsi prie VITMOS centrinio web serverio;

- Jei ryšys su vietiniu tinklu nutrūktų, įrenginys bandytų vėl prisijungti prie tinklo.

Tokiu būdu veikiantis įrenginys, įmontuotas judančiame objekte, galėtų teikti informaciją apie savo buvimo vietą.

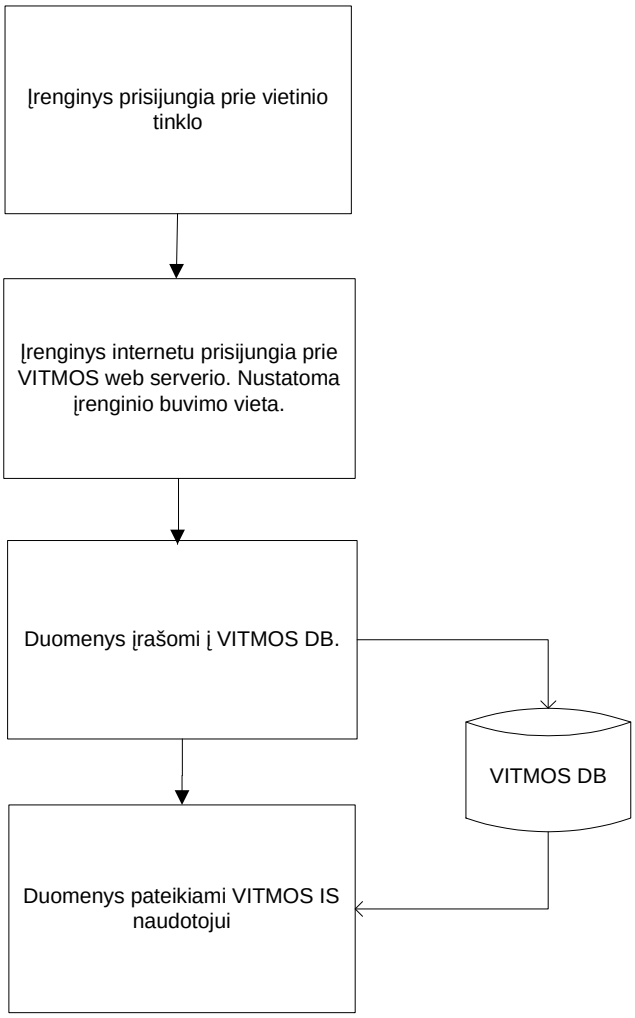
Žemiau yra pateikta schema, kaip WLAN turėtų būti naudojama VITMOS informacinėje sistemoje:

1. judančiame objekte įmontuotas įrenginys prisijungęs prie tinklo jungtųsi prie VITMOS IS web serverio;

2. Web serveryje esanti programinė įranga apdorotų gautą informaciją – nustatytų iš kokio tinklo buvo prisijungta, pagal konkretaus tinklo buvimo vietą būtų apytiksliai nustatyta ir judančio objekto buvimo vieta;

3. informacija apie objekto buvimo vietą tam tikru laiko momentu būtų patalpinta VITMOS duomenų bazėje;

4. informacija pateikiama VITMOS sistemos naudotojui.



Pav. 5.1.3.2 WLAN panaudojimo VITMOS IS schema

5.1.1.2 Technologijų palyginimas ir lyginamosios analizės rezultatai

Technologijų palyginimui yra pasirinkti keturi palyginimo parametrai: vietos nustatymo tikslumas, objekto judėjimo trajektorijos diskretiškumas, geografinis ribotumas ir paslaugų bei įrangos kaina.

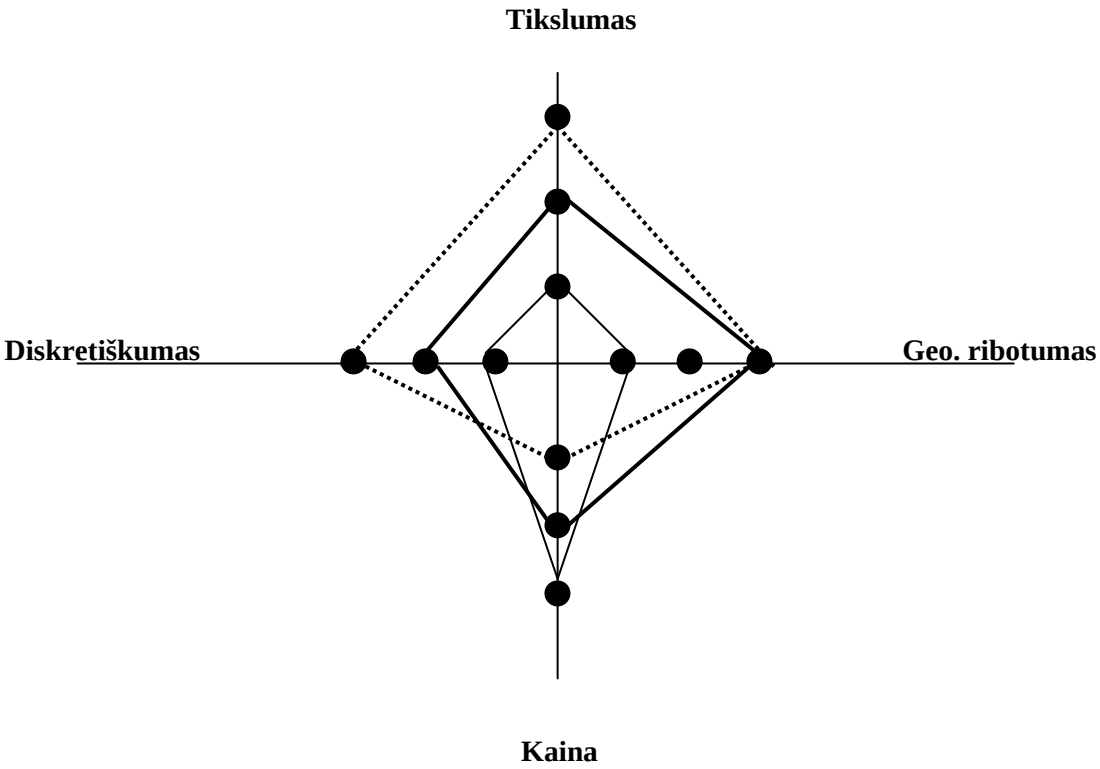
Vietos nustatymo tikslumas – šis parametras nusako, kiek tiksliai yra nustatoma judančio objekto vieta;

Objekto judėjimo trajektorijos diskretiškumas – nusako kokiu dažniu yra galima fiksuoti objekto buvimo vietą;

Geografinis ribotumas – nusako kiek technologija apriboja priklausomai nuo geografinės vietos.

Paslaugų bei įrangos kaina – nusako šiuo metu siūlomą naudojimosi tam tikra technologija paslaugų ir įrangos kainą.

Išnagrinėjus technologijas, jas vertinant pagal aukščiau įvardintus parametrus, galima pateikti galutinius tyrimo rezultatus. Rezultatai pateikti žemiau esačioje schemoje:



GSM ————

GPS ·····

WLAN ————

Šioje schemoje kiekviena ašis vaizduoja tam tikrą parametrą. Kuo ašyje reikšmė yra didesnė, tuo geresni yra technologijos rezultatai atitinkamo parametro atžvilgiu.

Išvada

Pagal aukščiau minėtą konkrečią vieną iš VITMOS IS užduočių – nustatyti judančios transporto priemonės buvimo vietą ir remiantis gautais objektų buvimo vietos nustatymo technologijų palyginimu, pasirinkta GSM technologija. Ji pasirinkta todėl, kad transporto priemonės yra stebimos didelėje teritorije, nėra svarbus itin didelis tikslumas, o šių paslaugų kaina yra kur kas mažesnė.

5.1.2 Oracle CDM Fastrack metodikos aprašymas

Kaip minėta aukščiau, pagal Oracle CDM Fastrack metodiką yra kuriama VITMOS IS.

5.1.2.1 Apie Oracle CDM Fast Track

Oracle CDM (Custom Development Method) Fast Track yra didelių ir vidutinio dydžio sistemų kūrimo metodika, atitinkanti Pasaulyje pripažintus standartus. Pasaulyje yra įvairių sistemų kūrimo metodikų, tačiau šiandieninei rinkai yra reikalinga tokia, kurios pagrindu sistemas būtų galima kurti sparčiai, efektyviai ir mažai rizikingai, nes būtent toks kūrimo būdas užtikrina sistemos kūrėjo sėkmę. Sukūrusi šią metodiką Oracle kompanija patenkino rinkos poreikį – Spartų Sistemų Kūrimą (Rapid Application Development - RAD). Oracle CDM yra sukurtas sudėjus RAD ir Dinaminių Sistemų Kūrimo Metodą (Dynamic Systems Development Method - DSDM). Daugeliu aspektų CDM sutampa su RAD įskaitant laiko matavimo principą, prioritetų parinkimą, prototipų kūrimą, kurie leidžia projekto komandai greitai, sėkmingai, efektyviai, su minimaliu rizikos faktoriumi pateikti klientui produktą.

Oracle CDM Fast Track pritaikytas darbui su Oracle duomenų baze ir jos įrankiais (Oracle Designer), leidžiančiais generuoti duomenų bazės skriptus ir programinės įrangos pirminius išeities tekstus. Toks būdas ypač paspartina sistemos kūrimą, sumažina pasitaikančių klaidų tikimybę. Programinės įrangos kūrimo metu sutaupomas laikas leidžia daugiau laiko skirti analizei ir projektavimui, tai apsaugo nuo skubotų ir ne itin gerai apgalvotų sistemos pagrindų kūrimo veiksmų, nuo kurių labai priklauso tolimesnė projekto eiga ir sėkmė.

Oracle CDM Fast Track metodas neteigia, kad turi būti būtinai naudojami Oracle įrankiai, tokie kaip Oracle Designer. Pasak jo, nenaudojimas šių įrankių šiek tiek apriboja metode aprašytas galimybes.

Tarkim paprastai kuriamam projektui atlikti užtenka 18 mėn., tuo tarpu vykdant pagal CDM Fast Track metodą, projektą galima sukurti per 6 mėn. ir mažiau. Projekto eigoje yra labai didelė rizika – reikalavimų pasikeitimas vykdymo metu. Greitas projekto atlikimas labai sumažiną šios rizikos pasitvirtinimo tikimybę.

Projektai pagal Oracle CDM Fast Track dažniausiai trunka apie 6 mėn. juos dažniausiai atlieka nuo 3 iki 6 žmonių. Jei projektas yra didesnis, jis gali būti tai pat vykdomas pagal CDM Fast Track, jei jis suskaidomas į mažesnes dalis.

Klientai dažnai nelabai supranta, kokie yra jų veiklos reikalavimai kol gyvai nepamato sistemos prototipo. CDM Fast Track numato prototipo kūrimą ankstyvoje projekto fazėje, kuris leidžia klientui susipažinti su būsima sistema ir padeda lengviau suprasti ir apibrėžti savo veiklos procesus ir kaip jie bus realizuoti programinėje įrangoje.

5.1.2.2 Oracle CDM Fast Track apibrėžiamos fazės

Šiame skyriuje yra aprašytos Oracle CDM Fast Track apibrėžiamos fazės:

- Apibrėžimas (Definition);
- Reikalavimų modeliavimas (Requirements Modeling);
- Projektavimas ir generavimas (System Design and Generation);
- Diegimas ir palaikymas (Tranzition to production).

5.1.2.2.1 Apibrėžimas (Definition)

Apibrėžimo fazėje yra kuriama projekto koncepcija, joje aprašomi projekto tikslai, norimi pasiekti rezultatai sukuriamas bendri (high-level) veiklos modeliai, kurie registruojami Oracle Repository duomenų bazėje (vadinamame CASE). Šiame etape yra svarbu suskirstyti apibrėžtus reikalavimus pagal prioritetus, taip pat juos suskirstyti į tam tikras dalis (partition) pagal veiklą. Nustatomos priklausomybės tarp tų dalių bei galimas realizavimo eiliškumas. Šis suskirstymas yra labai svarbus, nes šitaip išskaidžius projektą į atskiras nepriklausomas dalis, jos gali būti vykdomos vienu metu atskirų projekto komandų.

Šioje fazėje apibrėžiamos projektų valdymo planas (projektų valdymą aprašo Oracle metodas PJM) bei darbų kalendorinis planas, pagal kuriuos toliau vykdomi tolimesni darbai.

5.1.2.2.2 Reikalavimų modeliavimas (Requirements Modeling)

Šioje fazėje analizuojami ir smulkiau aprašomi veiklos modeliai. Naudojant Oracle Designer įrankį, galima sėkmingai aprašyti įvairius veiklos modelius. Stambesni loginiai modeliai yra detalizuojami iki tokio lygio modelių, kuriuos jau galima realizuoti fiziniu būdu: duomenų bazės objektais, programinės įrangos moduliais, kurie yra generuojami Oracle Designer pagalba.

Šios fazės metu yra svarbu apsibrėžti visų realizuojamų funkcijų prioritetus, pagal panaudojimo svarbą veikloje. Oracle CDM Fast Track funkcijas pagal prioritrtus skirtsto į 4 grupes:

- P – privalo būti (M – must be);
- T – turi būti (S – should be);
- G – gali būti (C – could be);
- N – neturi būti (W - would not be);

Šis funkcijų suskirstymas yra vadinamas MoSCoW sąrašu (pagal pirmąsias prioritetų raides MSCW). MoSCoW sąrašą yra labai svarbu sukurti todėl, kad dažnai projekto eigoje dėl vienokių ar kitokių priežąsčių nespėjama realizuoti visų funkcijų, todėl jos realizuojamos pagal prioritetus. Esant vėlamimui daugiausia nerealizuotų funkcijų būna su mažiausiu prioritetu.

5.1.2.2.3 Projektavimas ir generavimas (System Design and Generation)

Šios fazės metu yra pilnai sukuriama sistemos programinė įranga. Jei sistema kuriama naudojant Oracle Designer įrankį, programinė įranga arba didžioji jos dalis yra generuojama. Ši fazė dažniausiai vyksta keliomis iteracijomis. Kiekvienos iteracijos pabaigoje klientai yra supažindinami su jos rezultatais – ištestuota programine įranga. Projektavimo ir generavimo fazės metu vyksta pasikeitimai visuose sistemos lygiuose: tiek veiklos, tiek loginiame, tiek fiziniame. Klientui pateikus pakeitimus, pirmiausiai atliekami pakeitimai veiklos lygmenyje – gali būti perbraižomi procesai, perrašomos funkcijos, perkuriamos esybės, po to loginiame – iš pakeistos esybės trasformuojamos į lenteles, atnaujinami moduliai, jų laukai, funkcionalumas. Visi šie veiksmai atliekami

Oracle Designer įrankiu, o duomenys saugomi Oracle Repository duomenų bazėje. Galiausiai pakeitimai atliekami ir fiziniame lygmenyje – iš CASE sugeneruojami duomenų bazės objektai ir programinės įrangos moduliai.

Ši fazė ir kiekviena jos iteracija yra vykdomos tol kol baigiasi jai numatytas laikas pagal ankstesnėse fazėse aprašytą darbų kalendorinį planą. Pasibaigus numatytam laikui, generavimo darbai baigiasi, programinė įranga ištestuojama ir pateikiama klientui. Testavimo metodika aprašyta skyriuje 5.1.3. Vėliau visi programinėje įrangoje atliekami pakeitimai fiksuojami. Tiek klientas tiek projekto komanda mato keitimų chronologiją.

5.1.2.2.4 Diegimas ir palaikymas (Tranzition to production)

Šioje fazėje sukurta sistema įdiegiama klientui. Joje gali būti vykdomi labai nedideli pakeitimai. Sistema aktyviai testuojama bandoma, atliekamas galimas naujų arba nerealizuotų funkcijų realizavimo darbų planavimas. Atliekamas sistemos palaikymas.

5.1.2.3 VITMOS IS pagal Oracle CDM Fast Track

Šiame skyriuje aprašyta, kokios Oracle CDM Fast Track metodo dalys ir priemonės yra panaudotos, kuriant “VITMOS Interneto portalą”. TIP yra kuriamas pagal Oracle CDM metodą: visa sistema aprašoma CASE, naudojami informacinės sistemos kūrimo etapai, apibrėžti minėtame metode. Dėl pasirinktos projekte programavimo kalbos nėra pilnai išnaudojamos metodo galymybės – moduliai nėra generuojami iš CASE.

5.1.3 Testavimo metodikos aprašymas

Šiame skyriuje yra pateikta testavimo metodika - aprašyti testavimo principai, testavimo tikslai, nurodytos pagrindinės klaidos, kurios daromos testuojant. Apibrėžti testavimo etapai ir naudojamos testavimo technikos.

5.1.3.1 Testavimo apžvalga

5.1.3.1.1 Testavimo apibrėžimas

Testavimas – tai procesas, kurio metu siekiama rasti programinės įrangos (toliau - PĮ) klaidas ir įrodyti, kad PĮ tenkina reikalaujamą programinės įrangos kokybę.

Testavimo metu siekiama rasti klaidas testuojamame objekte. Testavimo objektu gali būti – modulis, sistema, dokumentacija. Testavimas turi dvi dalis – verifikavimą ir validavimą.

Verifikavimas – tai veikla, kuria siekiama įsitikinti, kad tam tikra funkcija vykdoma korektiškai (Atsako į klausimą, ar teisingai veikia kuriamas produktas?).

Validavimas – veikla, kuria siekiama įsitikinti, kad funkcija atitinka vartotojo reikalavimus (Atsako į klausimą, ar kuriamas teisingas produktas?).

Testavimu nesiekiama parodyti, kad PĮ veikia teisingai. Priešingai, testuotojas turi pradėti testavimą laikantis prielaidos, kad programa turi klaidų ir stengtis surasti jų kiek galima daugiau.

5.1.3.1.2 Testavimo principai:

1. Testavimas pradedamas smulkiausiame, modulių, lygyje ir plečiamas, kol apima visą sistemą.
2. Išskiriami šie testavimo etapai:
    a. Modulių testavimas – testuojamos smulkiausios sudėtinės sistemos dalys – moduliai;
    b. Sistemos testavimas – testuojama atskirų modulių sąveikatarpusavyje ir ar sistema tenkina specifikacijoje keliamus reikalavimus;
    c. Sistemos integracijos testavimas – testuojamos sistemos sąsajos su kitomis kompiuterinėmis sistemomis.
3. Testavimo metu vykdomi testavimo atvejai. Testavimo atvejis yra sudarytas iš tam tikrų įėjimo duomenų, veiksmų ir tikėtinų rezultatų. Testavimo atveju siekiama patikrinti sistemos atitikimą specifikacijoje keliamiems reikalavimams.
4. Geras testavimo atvejis yra tas, kurio rezultatas yra iki tol nerasta klaida. Daugumai sėkmingas testavimo atvejis asocijuojasi su gautais teisingais testavimo rezultatais, t.y. kai nesurandama klaidų – sėkmingas, kai surandama – nesėkmingas. Remiantis testavimo apibrėžimu, sėkmingas testavimo atvejis yra tas, kuris surado klaidas. Taigi, geras testavimo atvejis, kuris turi didelę tikimybę surasti dar nerastas klaidas.

5. Kiekvienam testavimo atvejui nurodomi tikslūs tikėtini rezultatai. Jeigu testavimo atvejis neturi jų, yra tikimybė, kad klaidingas rezultatas bus interpretuotas kaip teisingas. Todėl būtina tiksliai išsiaiškinti visus galimus programos išėjimo duomenų rinkinius. Kiekvieną testavimo atvejį turi sudaryti dvi dalys – įėjimo duomenų

aprašymas ir tuos duomenis atitinkančių teisingų tikėtinų rezultatų nurodymas. Lyginant tikėtiną ir gautą rezultatus lengviau identifikuojamos klaidos.

6. Testavimo atvejai turi būti parašomi tiek galimiems įėjimo duomenims, tiek ir negalimiems įėjimo duomenims. Pastebima, kad testuojant susikoncentruojama tik ties galimais ir tikėtinais įėjimo duomenimis, t.y. stengiamasi patikrinti, ar programa teisingai interpretuoja galimus duomenis, ar gaunami rezultatai atitinka reikalavimus. Tačiau, nemažai klaidų pastebima, kai su programomis dirbama skirtingomis nei buvo tikėtasi sąlygomis. Todėl papildomai turi būti rašomi testavimo atvejai negalimiems įvedimo duomenims – sąlygoms, kurios turi didesnę klaidų aptikimo tikimybę tam, kad nustatyti kaip programa veiks su nenumatytais įvedimo duomenimis ar nenumatytomis sąlygomis.

7. Tikimybė, kad programoje egzistuoja daugiau klaidų yra proporcinga jau surastų klaidų skaičiui, t.y. modulyje, kuriame buvo surasta sąlyginai daugiau klaidų, galima tikėtis rasti daugiau klaidų nei tame, kuriame rasta mažiau klaidų. Surastas didesnis klaidų kiekis rodo, kad programuotojas neteisingai ar ne išsamiai išsiaiškino reikalavimus, arba neteisingai panaudojo programavimo galimybes ir pan. Be to, kiekvienos klaidos taisymas gali sąlygoti papildomų klaidų atsiradimą, t.y. kuo daugiau klaidų taisoma, tuo didesnė tikimybė surasti naujas klaidas.

8. Sistemos programinės įrangos testavimą atlieka ne jos kūrėjas. Programuotojai neturėtų testuoti savo programų. Programuotojui yra sunku pakeisti mastymą, kad jo programoje gali būti klaidų. Taip pat programa gali turėti klaidų dėl programuotojo blogai suprastų reikalavimų. Bandant testuoti, jis bus tokios pačios nuomonės apie reikalavimus. Be to, kritikuoti savo paties darbą yra labai sunku. Todėl programuotojai savo sukurtų programų negali efektyviai testuoti.

9. Testai turi būti rengiami taip, kad juos galima būtų kartoti. Testų pakartojimas ypač reikalingas programuotojams, kad galėtų pakartoti (sumodeliuoti) testavimo atvejį, kuriame buvo rasta klaida. Taip pat testų pakartojimo galimybė yra viena iš sąlygų vykdyti regresinį testavimą, kai pakartotinai testuojamas išleistas naujas programos leidinys.

10. Testavimas turi būti planuojamas. Kiekvienam testavimo etapui turi būti sudaromas testavimo planas tam, kad pakankamai laiko ir resursų būtų skiriama testavimo užduočių vykdymui, kad testavimo procesas būtų tinkamai kontroliuojamas ir valdomas. Testavimo planavimas turi būti derinamas su projekto valdymo planu. Testavimo grupės vadovas ir projekto vadovas turi bendradarbiauti, kad suderinti darbą. Sudarant testavimo planą turi būti įvertintos testavimo rizikos, t.y. kur gali būti uždelsimas testuojant. Būtina įvertinti, kokio sudėtingumo funkcijas-operacijas atlieka programos, kuris komponentas reikalauja sudėtingesnio testavimo arba kad testuojant komponentą testuotojas turės papildomai išmokti naudotis nauju įrankiu (programine priemone).

11. Testavimo plano negalima sudarinėti remiantis prielaida, kad klaidų nebus rasta. Testavimas yra procesas, kurio tikslas yra surasti klaidas. Negalima sudarinėti testavimo plano, atsižvelgiant vien tik į tai, kokio sudėtingumo funkcijas-operacijas atlieka programos. Papildomai turi būti skiriamas dėmesys tokiems aspektams – ar tai visiškai nauja programa, jeigu tai nėra nauja – kiek klaidų prieš tai vykusio testavimo metu buvo rasta, koks testuotojas jį atliko ir atliks ir t.t. Be to, daugiau laimi tie, kurie pasilieka galimybę koreguoti testavimo planą testavimo eigoje.

12. Testavimo veikla turi būti integruota į programinės įrangos kūrimo ciklą. Testavimo darbai pradedami sistemos projektavimo ir programavimo fazės pradžioje ir baigiami diegimo fazėje, įvykdžius sistemos priėmimo testus ir patvirtinus jų įvykdymo rezultatus.

13. Testavimas - daug kūrybiškumo reikalaujantis darbas. Testavimui siūlomi įvairūs metodai. Tačiau nei vienas jų negarantuoja, kad programa bus ištestuota pilnai – t.y. joje nebeliks klaidų. Kiek klaidų bus rasta ir kiek jų liks nepastebėta, priklauso vien tik nuo testuotojo kūrybiškumo: kokius metodus jis pasirinks, kaip juos suderins tarpusavyje, kokius testavimo atvejus parašys ir pan.

14. Tiktai išsamus testavimas gali parodyti, kad programa yra be klaidų, tačiau jis yra neįmanomas. Negali būti rastos visos klaidos.

5.1.3.1.3 Testavimo tikslai:

- Rasti klaidas ir sistemos trūkumus;
- Patikrinti, ar sistema atitinka reikalavimų specifikacijoje keliamus reikalavimus;
- Patikrinti, ar sistemos dokumentacija atitinka sistemos PĮ veikimą;
- Patikrinti, ar sistema atitinka vartotojų poreikius;
- Išsiaiškinti, kaip programa veikia neįprastomis sąlygomis ir neįprastose situacijose;
- Padidinti programos patikimumą;
- Įvertinti, ar programa yra paruošta naudojimui;
- Įvertinti, sistemos programinės įrangos kokybę.

5.1.3.1.4 Pagrindinės testavimo klaidos:

1. Nerandama svarbių klaidų (Svarbi klaida – ta, kurią gali aptikti vartotojas);
2. Netestuojama sistemos patogumo;
3. Testavimas pradedamas per vėlai (klaidos užfiksuojamos, bet nebėra laiko jų ištaisymui);
4. Visos testavimo pastangos skiriamos funkciniam testavimui;
5. Neteikiama didelė reikšmė konfigūracijos testavimui;
6. Netestuojama dokumentacijos;
7. Instaliavimo procedūrų netestuojama;
8. Dėmesys sukoncentruojamas į testavimo atvejų vykdymą, o ne testavimo atvejų kūrimą;
9. Griežtai laikomasi testavimo plano;
10. Dėmesys sutelkiamas į tai ką sistema turi daryti ir netestuojama ar sistema nedaro to, ko neturi daryti.
11. Testuojama tiktai per vartotojo interfeisą;
12. Neaiškiai aprašomos klaidos.

5.1.3.2 Testavimo technika

Šiame skyriuje apibrėžtos bendros testavimo technikos, funkcinės testavimo technikos ir nefunkcinės testavimo technikos.

5.1.3.2.1 Bendros testavimo technikos

Bendras testavimo technikas sudaro šie testavimo metodai:

1. Teigiamas ir neigiamas testavimas
2. Juodos dėžės testavimas;
3. Baltos dėžės testavimas;
4. Automatizuotas testavimas;
5. Statinis testavimas.

5.1.3.2.1.1 Teigiamas (pozityvus) ir neigiamas (negatyvus) testavimas

Teigiamu (pozityviu) testavimu siekiama patikrinti ar sistema daro tai, ką turi daryti, ar veikia taip, kaip dokumentuota reikalavimų specifikacijoje, ar įvedus „geras/tinkamas" reikšmes sistema elgiasi taip, kaip vartotojas tikisi.

Neigiamas (negatyvus) testavimas parodo, kad sistema nedaro to, ko neturi daryti, ir įvedus „netinkamas" reikšmes sistema elgiasi taip, kaip vartotojas tikisi. Neigiamas testavimas yra naudingas įvertinant reikalavimų specifikaciją – parodo reikalavimų trūkumus.

5.1.3.2.1.2 Juodos dėžės testavimas

Juodos dėžes testavimas – testavimo technika, kai programa yra įsivaizduojama kaip juoda dėžė nevertinant jos struktūros, t.y. visiškai nesidomima vidiniais programos procesais, jos struktūra. Programos testavimo atvejai yra paremti sistemos reikalavimų specifikacija. Testuotojo tikslas - surasti situacijas, kai programa veikia ne taip, kaip nurodyta reikalavimų specifikacijoje. Testavimo duomenys sudaromi remiantis reikalavimų specifikacija. Norint surasti visas klaidas programoje, remiantis vien tik šiuo būdu, reikia atlikti išsamius „įėjimo" testavimus (exhaustive input testing). T.y. testuojami visi įėjimo atvejai. Visi įėjimo atvejai – tai ne visi galimi įėjimo atvejai, bet visi įmanomi įėjimo atvejai.

5.1.3.2.1.3 Baltos dėžės testavimas

Baltos dėžes testavimas – testavimo metu analizuojama vidinė programos struktūra. Testiniai atvejai gaunami iš programos struktūros. Žinios apie programą naudojamos nustatyti papildomus testinius atvejus. Įėjimo duomenys parenkami nagrinėjant programos logiką. „Baltos dėžės" testavimas paremtas išsamiu kelių testavimu (exhausive path testing), t.y. jeigu kiekviena programos šaka įvykdoma bent vieną kartą, tokiu atveju galima sakyti, kad programa ištestuota pilnai. Tačiau toks testavimas turi du trūkumus:

- unikalių loginių kelių kiekis programoje yra labai didelis;
- nors išsamaus kelių testavimo metu visi programos keliai įvykdomi bent po vieną kartą, tačiau tai dar negarantuoja kad programoje neliko klaidų. Pvz.:
    1. Programa gali netenkinti reikalavimų (pvz. atliekamas vienas veiksmas vietoj kito);
    2. Programoje gali trūkti keleto kelių.

Toks testavimas negali išaiškinti klaidų, priklausančių nuo įėjimo duomenų. Baltos dėžės testavimas taikomas ankstyvose testavimo fazėse.

Šiai strategijai priklauso tokie testavimo metodai:

Programos sakinių testavimas – testavimas paremtas reikalavimu, kad kiekvienas programos sakinys būtų įvykdytas bent po vieną kartą. Šis metodas yra brangus, todėl retai naudojamas.

Sprendimų testavimas – testavimo metu įvykdoma tiek testavimo atvejų, kad kiekvienas sprendimas turėtų ‘true’ ir ‘false’ išėjimus bent po vieną kartą. T.y. kiekviena programos šaka būtų peržiūrėta bent po vieną kartą.

Sąlygų testavimas – testavimo metu kiekviena sprendimo sąlyga įvykdoma visiems galimiems rezultatams gauti.

Sprendimų/sąlygų testavimas – testavimui keliami reikalavimai: kiekviena sprendimo sąlyga įvykdoma visiems galimiems rezultatams gauti bent po vieną kartą, kiekvienas sprendimas įvykdomas visiems galimiems rezultatams gauti bent po vieną kartą, taip pat turi būti pratestuotas kiekvienas galimas duomenų įvedimo būdas.

Sudėtinių sąlygų testavimas - taikomas kai programose naudojamos sudėtinės sąlygos. Jam keliami reikalavimai: kiekviena sąlygų kombinacija, lemianti sprendimą, turi būti įvykdoma bent po vieną kartą, taip pat turi būti pratestuotas kiekvienas galimas duomenų įvedimo būdas.

5.1.3.2.1.4 Tiriamasis/klaidų spėjimo testavimas (Error quessing)

Kai kurie žmonės, nenaudodami jokios testavimo metodikos, turi įgūdžius surasti klaidas. Tai paaiškinama tuo, kad šie žmonės praktikos dėka pasiekia tokį lygį, kad gavę naują programą testavimui, pasitelkę savo patirtį ir intuiciją, aptinka tam tikros rūšies klaidas greitai ir efektyviai. Remiamasi intuicija ir patirtimi, kas keista, kas ne taip, kaip turėtų

veikti. Toks klaidų ieškojimo metodas vadinamas tiriamuoju testavimu.

Sudėtinga nustatyti procedūrą, kuri taikoma tiriamajam testavimui, kadangi daugeliu atvejų tai pilnai intuityvus procesas. Pagrindinė idėja - reikia išvardinti galimų problemų sąrašą ir juo remiantis parašyti testavimo atvejus. Kita idėja - pabandyti įsivaizduoti kokias prielaidas galėjo padaryti programuotojas skaitydamas reikalavimus ir pagal prielaidas rašyti testavimo atvejus. Tiriamąjį testavimą verta daryti kai reikia greitai įvertinti produkto kokybę.

5.1.3.2.1.5 Automatizuotas testavimas

Automatizuoto testavimo metu naudojami programinės įrangos įrankiai programos kodo analizavimui. Jie išnagrinėja programos tekstą, bando surasti potencialiai klaidingas sąlygas ir atkreipti į jas testuotojų komandos dėmesį. Testų automatizavimas – programinės įrangos kūrimo procesas, reikalaujantis didelės investicijos. Testavimo įrankiai sumažina reikiamo laiko testavimui ir testavimo kaštus. Pavyzdžiui, klaviatūros bei pelės paspaudimai gali būti užrašyti į failą ir po to automatiškai pakartoti. Apie testavimo sėkmę sprendžiama pagal tarpines ir galutines būsenas/duomenis.

Kodėl reikalingas automatizuotas testavimas?

Programos kūrimo, derinimo, priežiūros metu tenka tą pačią programą ir tuos pačius komponentus testuoti daug kartų po kiekvieno atlikto pakeitimo. Testavimo kartojimas jeigu jis neautomatinis yra labai imlus darbui. Papildomos pastangos automatiniam testavimui visuomet atsiperka.

Kai kuriais atvejais gali būti taikomas ir automatizuotas ir rankinis testavimas, tai funkcionalumo testavimui, vartotojo interfeiso testavimui (jei interfeisas pastovus), laiko ir datos testams. Automatizuotas testavimas geriausias yra paprastiems testams. Sudėtinguose testuose sunku atsekti kur įvyko klaida.

5.1.3.2.1.6 Statinis testavimas

Statinio testavimo technika įvertina sistemos kokybę ir nereikalauja sistemos vykdymo. Statinis testavimas dažniausiai atliekamas reikalavimų modeliavimo fazėje.

Statinio testavimo metodai:

- Desk check (kodo tikrinimas) – kai programuotojai tikrina parašytą programos kodą;

- Walkthrough (peržiūra) – kai projektuotojas ar programuotojas pažingsniui peržiūri dokumentacijos ar kodo dalį tam, kad surasti ir išspręsti projektavimo ir diegimo problemas;
- Inspektion (peržiūra) – kai vartotojas peržiūri sistemos interfeisą tam tikroje fazėje kūrimo proceso.

5.1.3.2.2 Funkcinės testavimo technikos

Funkcinės testavimo technikos yra šios:

- Ekvivalentinis klasių sudalinimas;
- Ribinių reikšmių analizė;
- Atsitiktinis testavimas;
- Perėjimo per būsenas testavimas;
- „Įkyrus/įsikišantis" testavimas;
- Scenarijų testavimas.

5.1.3.2.2.1 Ekvivalentinis klasių sudalinimas (Equivalence class partitioning)

Neįmanoma naudoti išsamaus įėjimo testavimo principo dėl labai didelės duomenų aibės realioje programoje. Išeitis - testuotojas turi apibrėžti mažesnę duomenų aibę. Reikia pasirinkti tokią duomenų aibę, kuriai esant tikimybė rasti daugiausiai klaidų programoje yra didžiausia. Sudarant tokį duomenų rinkinį, reikia atsiminti, kad gerai parinktas testavimo atvejis turi turėti tokias savybes:

- Jis sumažina kitų testavimo atvejų, kurie gali būti sukurti problemos/reikalavimo patikrinimui, kiekį daugiau negu vienetu.
- Jis padengia platų kitų galimų testavimo atvejų rinkinį. T.y. parodoma apie klaidų egzistavimą arba neegzistavimą tam tikroje įėjimo duomenų aibėje.

Pirma savybė nurodo, kad testavimo atvejis turi apimti kuo daugiau pradinių sąlygų tam, kad sumažinti testavimo atvejų kiekį. Antra - reikia padalinti įėjimo duomenų aibę į poaibius, vadinamus ekvivalentinėmis klasėmis (equivalence classes).

Įvedami duomenys ir išvedami rezultatai paskirstomi į atskiras klases, kur visi klasių nariai yra panašūs. Testuotojas turi apibrėžti mažesnę duomenų klasę. Kiekviena iš šių klasių yra

lygiaverčio suskirstymo rezultatas, kur programos elgesys su kiekvienu klasės nariu yra toks pat. Ištestavus vieną reikšmę iš klasės, sakome, kad su kitomis reikšmėmis iš tos klasės gausime tą patį rezultatą. Jeigu randama klaida ekvivalentinėje klasėje, kiti tos klasės testavimo atvejai turėtų aptikti tą pačią klaidą. Testiniai atvejai turi būti parinkti kiekvienai klasei. Reikia padalinti sistemos įėjimų ir išėjimų reikšmes į "ekvivalentiškas aibes" ir iš jų paimti reikšmes.

Ekvivalentinių klasių nustatymas:

Ekvivalentinės klasės nustatomos pagal kiekvieną įėjimo sąlygą (dažniausiai sakinį arba frazę iš sistemos reikalavimų) ir skaidant ją į dvi arba daugiau sub-sąlygų. Tokiu būdu nustatoma du ekvivalentinių klasių tipai:

- teisingos ekvivalentinės klasės, atstovaujančios teisingą duomenų įvedimą;
- neteisingos ekvivalentinės klasės, atstovaujančios visas kitas įmanomas pradines sąlygas (t.y. neteisingą duomenų įvedimą).

Pavyzdžiai:

- Tarkim, reikalavimuose pateikta 'privalo būti' situacija – pirmas identifikatoriaus simbolis turi būti raidė. Šiuo atveju teisinga ekvivalentinė klasė bus – pirma raidė, neteisinga – bet koks kitas simbolis.
- Į laukelį „Asmens kodas" galima įvesti 11 simbolių skaičių. Šiuo atveju teisinga ekvivalentinė klasė bus 11 simbolių, neteisinga – mažiau nei 11 simbolių ir daugiau nei 11 simbolių.

Testavimo atvejų sudarymas:

Panaudojant nustatytas ekvivalentines klases parašomi testavimo atvejai. Šį procesą sudaro tokie žingsniai:

- Visos ekvivalentinės klasės sunumeruojamos;
- Tol, kol visos teisingos ekvivalentinės klasės nėra „padengtos" testavimo atvejais, parašyti naują testavimo atvejį, apimantį kuo daugiau „nepadengtų" teisingų ekvivalentinių klasių.
- Tol, kol visos neteisingos ekvivalentinės klasės nėra „padengtos" testavimo atvejais, parašyti naują testavimo atvejį, „padengiantį" vieną ir tiktai vieną iš „nepadengtų"

neteisingų ekvivalentinių klasių.

5.1.3.2.2.2 Ribinių reikšmių analizė (Boundary – value analysis)

Ribinių reikšmių analizė skiriasi nuo ekvivalentinio suskaldymo dviem aspektais:

- Užuot pasirinkus bet kokią reikšmę iš ekvivalentinės klasės, ribinių reikšmių analizė reikalauja, kad vienas arba daugiau elementų būtų paimta iš kiekvienos ekvivalentinės klasės ribos.
- Užuot kreipus dėmesį tik į įėjimo sąlygas (input space), testavimo atvejai sudaromi atsižvelgiant į rezultatų aibę (result space) t.y. išėjimo ekvivalentines klases.

Atliekant ribinių reikšmių analizę, siūloma vadovautis tokiais kriterijais:

- Jeigu įėjimo sąlygos nurodo reikšmių ribas, turi būti parašomas testavimo atvejis galimoms riboms, o taip pat neteisingoms reikšmėms šalia šių ribų. Pvz.: jeigu užduotas reikšmių intervalas –1.0 - + 1.0, tai testavimo atvejis gali būti sudarytas reikšmėms : -1.0 , 1.0, -1.001 ir 1.001
- Jeigu įėjimo sąlygos nurodo reikšmių kiekį, testas turi būti parašytas minimaliam ir maksimaliam kiekiui, o taip pat reikšmei, kurios numeris yra vienetu mažesnis ir vienetu didesnis. Pvz: Jeigu pradiniai duomenys gali turėti 1-255 įrašus, tai testavimo atvejis turėtų būti parašytas įrašų kiekiui 0,1,255 ir 256.
- Principai 1 ir 2 turi būti naudojamas išėjimo duomenims.
- Jeigu programos įėjimą arba išėjimą sudaro surūšiuotos aibės (pvz. nuoseklus failas, sąrašas, lentele), dėmesys turi būti sukoncentruotas ties pirmu ir paskutiniu elementais.

Ribinių reikšmių analizė yra vienas dažniausiai naudojamų testavimo atvejų sudarymo metodų. Dažnai ekvivalentinis sudalinimas ir ribinių reikšmių analizės metodai apjungiami/naudojami kartu.

5.1.3.2.2.3 Atsitiktinis testavimas (Random testing)

Atsitiktinio testavimo principas – atsitiktiniai parenkami įvedimo duomenys. Atsitiktinis testavimas naudingas tuo, kad įvedimo duomenų neįtakoja nei programuotojas, nei testuotojas. Atsitiktinio testavimo vienas trūkumų yra tai, kad negalima atkartoti atsitiktinai parinktų įvedimo duomenų, taip pat sunku atkurti klaidas, rezultatus gautus atsitiktinio

testavimo metu.

Pavyzdys: Įėjimo duomenys moduliui yra visi teigiami skaičiai tarp 1 ir 100. Taigi atsitiktinai, nesistemingai parinktos įėjimo reikšmės gali būti 55, 24, 3.

5.1.3.2.2.4 Perėjimo per būsenas testavimas (State transition testing)

Šio testavimo tikslas yra nustatyti:

- kokias būsenas gali įgyti sistema;
- kaip pereiti iš vienos būsenos į kitą;
- kokie veiksmai gali įtakoti perėjimą iš vienos būsenos į kitą;
- kokie bus veiksmai perėjus iš vienos būsenos į kitą.

Ši testavimo technika ypač naudinga kai yra „varganai", nepilnai aprašyti sistemos specifikacijos reikalavimai arba kai reikia greitai, nedetaliai peržiūrėti sistemos veikimą.

5.1.3.2.2.5 „Įkyrus/įsikišantis" testavimas

„Įkyraus/įsikišančio" testavimo metu padaromi pakeitimai kode, kai testavimo tikslais reikalingi papildomi laukai interfeise arba tarpiniai pranešimai arba tarpinių rezultatų išvedimas, norint giliau patikrinti sistemos veikimą. Tokie testai turi būti vykdomi testavimo aplinkoje, o ne darbinėje aplinkoje. Šios technikos rizika - perduodama sistema, gali būti ne ta kuri buvo testuota. „Įkyrus/įsikišantis" testavimas gali būti atliekamas tiktai tada, kai:

- „backup'ai" yra saugomi;
- „backup'ai" yra izoliuoti/nepasiekiami darbinei sistemai;
- ankstesnės operacijos gali būti atstatytos iš „backup'ų".

5.1.3.2.2.6 Scenarijų testavimas

Scenarijų (thread) testavimas naudojamas testuoti veiklos procesus, testuojamas sistemos funkcionalumas. Sistemos funkcionalumui testuoti sudaromi testavimo scenarijai pagal reikalavimų specifikaciją. Kuo daugiau testavimo scenarijų yra pratestuojama, tuo išsamesnis yra testavimas. Scenarijams turi būti priskiriami vykdymo prioritetai, kad keičiantis testavimo apimtims būtų galima atmesti mažiau rizikingus testavimo atvejus.

5.1.3.2.3 Nefunkcinės testavimo technikos

Nefunkcinio testavimo technikos yra šios:

- Saugumo testavimas;
- Dokumentacijos ir pagalbos testavimas;
- Našumo testavimas;
- Apkrovimo testavimas;
- Apimties testavimas;
- Stresinis testavimas;
- Instaliavimo testavimas;
- Konfigūracijos testavimas;
- Suderinamumo ir tarpusavio sąveikos testavimas;
- Atsistatymo po klaidos testavimas;
- Patikimumo testavimas;
- Patogumo testavimas.

5.1.3.2.3.1 Saugumo testavimas (Security testing)

Šio testo tikslas yra patikrinti, ar yra užtikrinamas sistemos saugumas. Turi būti patikrinama, kaip sistema yra apsaugota nuo nepageidaujamų vartotojų, t.y ar ji yra prieinama vartotojams, kuriems nėra suteikta teisių ja naudotis. Vartotojui priskyrus tam tikras programos teises, tikrinama ar tik šios teisės yra suteikiamos, ar nėra priskiriama kitų teisių. Saugumo testo metu tikrinamas ir duomenų saugumas, priėjimas prie įvairių resursų. Išskiriami tokie saugumo lygiai:

- Aplikacijos lygio saugumas (testuojama ar vartotojui „prieinamos” sistemos funkcijos/duomenys, prie kurių jie turi teisę „prieiti”);
- Sistemos lygio saugumas (testuojama ar prie aplikacijos/sistemos gali „prieiti” tik tie vartotojai, kurie turi „priėjimo” teises).

Saugumo testo tipai:

- Slaptažodžių testavimas – ar slaptažodis susideda tik iš raidžių, ar tik iš simbolių, ar iš raidžių ir simbolių, ar slaptažodis gali būti žodis; koks turi būti mažiausias ir didžiausias slaptažodžio ilgis;
- Testai su teisingais ir neteisingais slaptažodžiais – ar įvedus teisingą slaptažodį leidžiamas priėjimas prie vartotojui skirtų duomenų/funkcijų, ar įvedus neteisingą slaptažodį priėjimas prie duomenų/funkcijų yra negalimas;
- Slaptažodžių galiojimo testavimas - ar slaptažodis nuolatinis/ilgalaikis, ar turi būti nustatytas slaptažodžio galiojimo laikas, ar pasibaigus galiojimo laikui reikalauja pakeisti slaptažodį;
- Kodavimas (encryption) – testuotojas turi įvertintų ar teisingai veikia užkodavimo ir atkodavimo algoritmai, kur yra koduojami duomenys/pranešimai;
- Naršymas (browsing) – ar naršymo privilegijos užtikrina, kad neautorizuotas naršymas yra negalimas, turi būti apibrėžta prie kokios dalies informacijos gali prieiti neautorizuoti vartotojai;
- Nenumatyti kanalai (trap doors) – patikrinti, ar nėra neapsaugotų įėjimų/"įsilaužimų" į sistemą per nenumatytus kanalus, sumodeliuoti testus, kurie įsilaužtų į sistemą ir stebėti rezultatus;
- Virusai – sumodeliuoti testus kurie užtikrintų, kad sistemos antivirusinės programos apsaugo sistemą nuo virusų ir duomenų praradimo, testuotojai privalo pabandyti „užkrėsti" sistemą vairiais virusais ir patikrinti kaip sistema reaguoja, jei virusas įsiskverbė į sistemą, nustatyti kas buvo sugadinta ir kokio dydžio žala padaryta sistemai.

Tačiau reikia prisiminti, kad netgi detaliai ištestavus saugumą, nėra jokios garantijos, kad sistema yra visiškai saugi.

5.1.3.2.3.2 Dokumentacijos ir pagalbos testavimas (Documentation and help testing)

Dokumentacijos testavimo metu testuojama vartotojui skirta dokumentacija ir pagalba. Testuojant dokumentaciją tikrinama:

- Ar dokumentacija ir pagalba yra tiksli, pilna, aiški?
- Ar dokumentacijoje ir pagalboje aprašytas sistemos (modulio) funkcionalumas yra teisingas?

- Ar dokumentacijoje naudojamos sąvokos ir žymėjimai turi turėti tą pačią prasmę visose vietose, kur jie naudojami?
- Ar nauji dalykai įtraukti į dokumentą? Ar nieko nepraleista?
- Ar nėra gramatikos, stiliaus klaidų?
- Ar to paties tipo dokumentai yra tos pačios struktūros?
- Ar to paties tipo objektai vieno tipo dokumentuose yra aprašomi naudojant tą patį šabloną?
- Ar pagalboje esančios nuorodos yra pasiekiamos?
- Ar dokumentaciją sudaro visi dokumentai, kurie turėtų būti?

Testuojant dokumentaciją testuotojas turi žinoti, kad jis tik testuoja dokumentą, o ne jį rašo. Testuotojo darbas yra aptikti neatitikimus ir klaidas.

„Gera" dokumentacija gali būti kaip vadovas testavimui, taip pat gali būti ir kaip mokymo dokumentas.

Vartotojo dokumentacija yra laikoma išbaigta, kai:

- **Vartotojo dokumentacija yra lengva naudotis.** Dokumentas yra naudingas tada, kai juo remiantis yra lengvas sistemos (modulio) panaudojimas.
- **Sumažina klientų aptarnavimo išlaidas.** Jei vartotojo dokumentacija parašyta neaiškiai ir sunkiai suprantama, tai gali padidėti išlaidos susijusios su užsakovo aptarnavimu. Šį darbą atlieka klientų aptarnavimo skyrius. Jei dokumentas yra parašytas aiškiai ir užsakovui nereikalingos klientų aptarnavimo skyriaus paslaugos, yra iš anksto užkertamas kelias bereikalingoms išlaidoms.
- **Užtikrina patikimumą.** Gerai parašyta vartotojo dokumentacija gali užkirsti kelią klaidoms, kurias padarytų vartotojas gerai nežinantis sistemos (modulio).
- **Didina modulio (sistemos) efektyvų panaudojimą.** Gerai parašyta vartotojo dokumentacija leidžia surasti greitą ir efektyvų kelią norimiems veiksmams atlikti.
- **Pagerina instaliavimą.** Dažnai užsakovas pats instaliuoja nusipirktą produktą arba nusisamdo žmogų, kuris tai atliktų. Kadangi šis žmogus dažniausia nežino būsimos sistemos funkcionalumo aspektų, jam iškyla klausimų susijusių su instaliavimu. Instaliavimo sėkmė priklauso nuo to, kaip aiškiai yra parašyta dokumentacija. Taip

pat dokumentacija turėtų atsispindėti ir reinstaliacijos eigą bei ypatumus.

- **Kokybės kaina (pigumas).** Dokumentacijos kokybė yra labai svarbus dalykas parduodant sistemą (modulį). Ji padeda pardavimo agentams paaiškinti modulį ir jį pademonstruoti. Taip pat dokumentacija dažnai padeda produktų peržiūros metu.
- **Sumažina atsakomybę.** Dokumentacijoje turi būti aprašyta tik tai, ką modulis ar sistema gali atlikti, ir nieko bereikalingo. Jei vartotojo dokumentacijoje bus aprašyta tai, ko sistema (modulis) iš tikrųjų neatlieka, gali iškilti nesusipratimų tarp gamintojų ir vartotojų.

5.1.3.2.3.3 Našumo testavimas (Performance testing)

Našumo testavimo tikslas – ištirti, ar sistema tenkina apibrėžtus veikimo bei efektyvumo reikalavimus. Gali būti matuojamas laiko arba atminties sunaudojimas, procesoriaus apkrovimas, kiti serverio parametrai.

Prieš pradedant našumo testavimą reikia išsiaiškinti:

- kokius įrankius sistemos našumo testavimui galima panaudoti;
- kokius duomenų tipus naudosim;
- su kokiais duomenų kiekiais testuosim;
- ką testuosime/seksime:
    - ar seksime/tikrinsime serverio resursus;
    - ar seksime/tikrinsime tinklo resursus;
    - ar seksime techninės įrangos resursus;
- kokia yra optimali techninė įranga sistemos veikimui.

Atliekant našumo testavimą svarbu yra žinoti, kokie yra keliami reikalavimai sistemos veikimui ir efektyvumui. Specifikacijoje turi būti nurodyta, koks turi būti maksimalus užklausų apdorojimo laikas, ataskaitų sukūrimo greitis. Testo metu tikrinama, kokioms sąlygoms esant jie nevykdomi.

5.1.3.2.3.4 Apkrovimo testavimas (Load testing)

Apkrovimo testavimas – tai našumo testavimo specifinė sritis. Tikslas – ištirti, ar sistema tenkina apibrėžtus veikimo bei efektyvumo reikalavimus esant skirtingam sistemos

apkrovimui. Per skirtingus laiko tarpus pateikiant skirtingus darbo krūvius (prisijungusių vartotojų kiekį, transakcijų kiekį ir pan.) identifikuojami sistemos komponentų silpnosios vietos - ‘bottlenecks‘. Apkrovimo testavimo metu tiriamas sistemos veikimas esant įvairiems darbo krūviams:

- esant normaliam darbo krūviui;
- esant vidutiniam darbo krūviui;
- esant didžiausiam darbo krūviui.

Atliekant apkrovimo testavimą svarbu yra žinoti, kokie yra keliami reikalavimai sistemos veikimui ir efektyvumui esant skirtingam darbo krūviui. Specifikacijoje turi būti nurodyta, koks turi būti maksimalus užklausų apdorojimo laikas, ataskaitų sukūrimo greitis esant skirtingam prisijungusių prie sistemos vartotojų skaičiui ir pan. Testo metu tikrinama, kokioms sąlygoms esant jie nevykdomi.

5.1.3.2.3.5 Apimties testavimas (Volume testing)

Šio testo metu tikrinamas programos veikimas esant dideliam duomenų kiekiui, t.y jo tikslas parodyti, kad programa negali valdyti didelio duomenų kiekio. Į sistemą įvedamas didelis duomenų kiekis ir stebima, kaip atitinkami moduliai juos apdoroja.

Apimties testavimas nuo našumo ar stresinio testavimo skiriasi tuo, kad apimties testavimo metu tikrinama, ar sistema gali apdoroti didelius duomenų kiekius, o našumo ar stresinio testavimo metu - ar tą didelį duomenų kiekį sistema apdoroja per nustatytą konkretų laiką. Pavyzdžiui:

- Apimties testavimo tikslas yra patvirtinti, kad sekretorė gali „suvesti” labai didelį dokumentą;
- Našumo testavimo tikslas yra patvirtinti, kad sekretorė gali „suvesti” 40 žodžius per minutę (jei maksimalus reikalavimas yra 40 žodžių per minutę);
- Stresinis testavimo tikslas yra nustatyti, kokį maksimalų žodžių skaičių sekretorė gali „suvesti” per minutę.

5.1.3.2.3.6 Stresinis testavimas (Stress testing)

Stresinis testavimas neturi būti sutapatinamas su apimties testu. Stresinis testavimas – testuojama sistema virš jos maksimaliai suprojektuoto krūvio. Sistemos spaudimas dažnai

lemia defektų pasirodymą. Stresinio testo metu stengiamasi pateikti kuo daugiau duomenų, užklausų ar vartotojų sistemai per trumpą laiko tarpą. Kadangi šis testas apima laiko faktorių, jis nėra vykdomas visoms programoms. Jis taikomas programoms, dirbančiom su kintančiu apkrovimo srautu, interaktyviom arba realaus laiko programoms. Pavyzdžiui, jei yra reikalaujama, kad sistema palaikytų 10 pertraukimų (interrupts) per sekundę, tai sumodeliavus 20 pertraukimų per sekundę sistema yra „stresinėje būsenoje". Pagrindinis stresinio testavimo tikslas ya „nulaužti" sistemą, nustatyti aplinkybes, kuriomis sistema „nulūžta". Sąlygos stresiniam testavimui:

- Mažai ar nėra serverio atminties;
- Prisijungęs maksimalus skaičius vartotojų;
- Daug vartotojų atlieka tas pačias transakcijas;
- Blogiausia transakcijų apimtis;
- Mažai ar nepakankamai vietos diske.

5.1.3.2.3.7 Instaliavimo testavimas (Installation testing)

Visą instaliavimo programos testavimą galima būtų suskirstyti į tokius etapus:

- **Pirmasis instaliavimas.** Pirmą kartą testuojant instaliavimo programą tikrinama ar įmanoma instaliuoti sistemą normaliomis (pirmas instaliavimas, sistema niekad nebuvo instaliuota kompiuteryje) ir nenormaliomis (nepakankamai vietos diske, kompiuteris netenkina minimalių reikalavimų, kai vartotojas neturi teisių kurti katalogus) sąlygomis. Pirmą kartą testuojant ypatingas dėmesys turi būti kreipiamas į tai, kaip programa sukuria programos paleidimui reikalingą aplinką, , kaip programa nustato reikiamus programos paleidimui konfigūracijos (registry) raktus.
- **Sistemos veikimas.** Tikrinama, kad sistema veikia korektiškai pirmą kartą instaliavus;
- **Sistemos išinstaliavimas.** Jei tam yra sukurta išinstaliavimo programa, tai tuokiu atveju ji turi būti paleista bei jai atlikti visi testai kaip ir instaliavimo programai. Jei tokios programos nėra, sistema išinstaliuojama standartiškai. Šio veiksmo metu būtina kreipti dėmesį į tai, ar visa sistema yra pašalinama iš kompiuterio atminties, ar yra ištrinami visi konfigūracijos raktai.
- **Pakartotinas instaliavimas.** Testuotojas turi atkreipti dėmesį, kaip programa atlieka

visus veiksmus pakartotinio programos instaliavimo metu. Jei dokumentacijoje nėra nurodyta, kad prieš pakartotinį programos naudojamą yra būtinas programos išinstaliavimas, sistemą reikia pabandyti suinstaliuoti be šio veiksmo. Kitu atveju pirma turi būti atliktas programos išinstaliavimas.

5.1.3.2.3.8 Konfigūracijos testavimas (Configuration testing)

Atliekant konfigūracijos testavimą yra tikrinama ar sistemos funkcijos veikia su reikalaujamomis techninės ir programinės įrangos konfigūracijomis. Problema atliekant konfigūracijos testavimą yra ta, kad gali būti daugybė sistemos konfigūracijos kombinacijų. Kiekviena konfigūracija turi būti individualiai pertestuojama. Reikalinga sužinoti iš vartotojo kokiomis konfigūracijomis jie naudosis. Jei yra kuriamas produktas konkrečiam vartotojui, reikia būtina patestuoti su ta konfigūracija, su kuria vartotojas dirba.

5.1.3.2.3.9 Suderinamumo ir tarpusavio sąveikos testavimas (Compatibility and interoperability testing)

Suderinamumo testu patvirtinama, kad suinstaliavus kitą sistema kompiuteryje, testuojamos sistemos funkcionalumas nepakinta, ir atvirkščiai. Testuojant tarpusavio sąveiką tikrinama, kaip sistema komunikuoja su kitomis sistemomis. Pavyzdžiui, ar gali sistemos skaityti duomenis viena kitos, ar suderinami failai.

5.1.3.2.3.10 Atsistatymo po klaidos testavimas (Fault recovery testing)

Šio testavimo metu tikrinama kaip sistema atsistato po klaidos, ar sistema grįžta į gerai žinomą, aiškią būseną po klaidos, ar sistemos dalys veikia korektiškai po atsistatymo. Atsistatymo po klaidos testavimo metu dėmesys sutelkiamas į:

- Persikrovimą (restart) – kai esama sistemos būsena ir veiksmai yra nutraukiami. Testuotojas turi patikrinti, kad visi veiksmai „atstatyti” korektiškai, kad visi įrenginiai yra veikia tinkamai, kad sistema gali vykdyti naujus veiksmus.
- Persijungimą (switchover) – turi būti testuojama sistemos galimybė įjungti naują procesorių.

5.1.3.2.3.11 Patikimumo testavimas (Reliability testing)

Patikimumas ≈ Našumas. Sistemos patikimumas – tai tikimybė, kad sistema veiks be klaidų nurodytomis sąlygomis per nustatytą laiko tarpą (ar pakankamai gera numatomam

naudojimui ir naudojant ją bus pasiektas reikiamas pasitikėjimo lygis). Dažniausiai patikimumas matuojamas naudojant skalę nuo 0 iki 1. Programinė įranga, kurios įvertinimas yra arti 1 yra aukšto patikimumo. Sistemos patikimumas ypač svarbus „gyvybei svarbioms" sistemoms (pvz. sistemoms, pritaikytoms ligoninėms, lėktuvams). Dažnai patikimumas matuojamas vidutiniu laiku iki klaidos (MTBF – mean time between failure).

5.1.3.2.3.12 Patogumo testavimas (Usability testing)

Atliekant patogumo testavimą tikrinama ar sistemos vartotojai veiksmingai, efektyviai, patogiai gali atlikti veiksmus. Patogumo testavimui naudojami šie metodai:

- Atitikimo tikrinimas (conformance checks) – ar atitinka vartotojo interfeiso standartus, technikas;
- Apklausų naudojimas (user – based surveys) – padarius susirinkimą su vartotojais vertinama kiek vartotojas patenkintas sistema:
    - Ar suprantama (kiek pastangų, laiko reikia, kol vartotojas išsiaiškina kaip atlikti norimą veiksmą);
    - Ar greitai išmokstama;
    - Ar pagalba aiški, greitai randama;
    - Ar atlikdamas veiksmus vartotojas nenustemba;
    - Ar vartotojas jaučiasi, kad jis kontroliuoja situaciją.

Rekomenduotina, kad patogumo testus atliktų vartotojai.

Atliekant patogumo testavimą reikią atkreipti dėmesį į:

- Navigavimą (kur galiu eiti, kaip išeiti);
- Funkcionalumą (ar galiu padaryti tai ko noriu, kokiu būdu, tvarka galiu tai padaryti);
- Kontrolę (ar jaučiu, kad kontroliuoju situaciją, sistemą);
- Pagalbą (ar pagalba pasiekiama tada, kai man jos reikia);
- Kalbą (ar suprantu terminologiją);
- Atsakomąja reakciją (ar aš žinau ką sistema daro);
- Nuoseklumą (jei vienam dialoge mygtukai išdėstyti apačioje, tai ir kitam dialoge turi

būti ten pat);

- Klaidos pranešimus (ar aišku, kas atsitiko blogai, kodėl blogai, kaip toliau elgtis);
- Vizualinį aiškumą (ar dizainas aiškus, patrauklus).

5.1.3.3 Testavimo etapai

Testavimo darbai dalinami į etapus:

1. Modulių testavimas;
2. Integracijos testavimas;
3. Sistemos testavimas;
4. Sistemos integracijos testavimas;
5. Priėmimo testavimas.

5.1.3.3.1 Modulių testavimas

Modulių testavimas tikrina smulkiausias sistemos dizaino sudėtines dalis – modulius. Modulio apibrėžimas priklauso nuo kuriamos sistemos ir technologijos, pasirinktos sistemos kūrimui:

- Sistemoje, sukurtoje procedūrinio programavimo kalba, modulis gali būti funkcija ar procedūra, arba artimai susijusių funkcijų grupės ar procedūros;
- Sistemoje, sukurtoje objektinio programavimo kalba, modulis gali būti klasė, objektas ar metodas (funkcijos naudojamos klasėje);
- Grafinėje aplinkoje ar GUI (user interface guidelines) kontekste modulis gali būti programos langas ar rinkinys susijusių elementų programos lange.

Modulių testavimas atliekamas kūrimo (development) aplinkoje. Dažniausiai modulių testavimą atlieka programuotojų komanda. Modulių testavimas atliekamas po kiekvienos modulių kūrimo iteracijos. Modulių testavimui turi būti naudojami fiktyvūs duomenys:

- Jei modulis ar funkcija nemanipuliuoja dideliais duomenų kiekiais, tai nedidelė duomenų aibė gali būti sumodeliuota rankomis;
- Jei modulis manipuliuoja dideliais duomenų kiekiais, tai patartina naudoti realių duomenų kopiją.

Moduliai testuojami remiantis modulių testavimo klausimynais ir projekto reikalavimų

specifikacija (toliau - reikalavimų sąrašu).

Modulių testavimo klausimynas apibrėžia modulių testavimo tvarką.

Testavimo technikos naudojamos modulių testavimui:

- Funkcinis testavimas;
- Statinis testavimas;
- Baltos dėžės testavimas;
- Nefunkcinis testavimas, kur tinka (pvz.: našumo, stresinis, saugumo).

5.1.3.3.2 Integracijos testavimas

Duomenų srautų judėjimo korektiškumas modulių viduje neužtikrina to paties tarp atskirų modulių. Duomenys gali būti prarasti tarp interfeisų, vienas modulis gali paveikti kitą. Integracijos testavimas turėtų būti vykdomas sistemiškai prijungiant vis naujus ir naujus modulius, kol visa sistema ima veikti kaip vientisa struktūra. Integracijos testavimas tikslai:

- Patikrinti ar neatsirado klaidų sujungus/susiejus modulius;
    - Patikrinti ar duomenų perdavimas tarp susijusių/sąveikaujančių modulių yra korektiškas;
    - Patikrinti suderinamumą;
- Sujungti atskirus modulius į veikiančią sistemą, kurią galima būtų testuoti.

Integravimo procesas turi 5 pakopas:

1. Pagrindinis modulis naudojamas kaip draiveris, o vietoje visų tiesiogiai prijungiamų modulių sukuriami stubai;
2. Stubai keičiami tikrais moduliais jungiant arba gilyn, arba platyn;
3. Stubai keičiami tikrais moduliais po vieną;
4. Po kiekvieno pakeitimo vykdomas testas;
5. Vykdomi pakartotiniai (regression) testai.

5.1.3.3.3 Sistemos testavimas

Sistemos testai tikrina visos sistemos kaip visumos darbą. Sistemai testuoti sudaromi testavimo scenarijai. Sistemos testavimo scenarijai apibrėžia veiksmų sekas, kuriomis remiantis testuojama sistema pagal reikalavimų specifikaciją. Sistemos testavimas išskiriamas į funkcinių reikalavimų ir nefunkcinių reikalavimų testavimą. Tipiniai sistemos testai atliekami testuotojų komandos. Sistemos testavimo metu tikrinami:

- Sistemos veiklos funkcionalumo reikalavimai;
- Sistemos nefunkciniai reikalavimai:
    - dokumentacijos bei pagalbos teksto reikalavimai;
    - vidinio saugumo reikalavimai;
    - veikimo bei efektyvumo (našumo) reikalavimai;
    - patogumo reikalavimai;
    - instaliavimo ir konfigūracijos reikalavimai;
    - ir kt.

Sistemos integracijos testai vykdomi testinėje aplinkoje panaudojant testinius duomenis. Sistemos testavimo scenarijai gali būti panaudojami sistemos priėmimo testavimui.

5.1.3.3.4 Sistemos integracijos testavimas

Sistemos integracijos testavimo metu tikrinama sistemos suderinamumas ir jos įtaka kitoms sistemoms (kaip sistema „bendradarbiauja” su kitomis sistemomis, ar testuojama sistema neturi nedaro neigiamo poveikio). Sistemos integracijos testavimas, jei įmanoma, atliekamas realioje aplinkoje pas klientą. Tipiniai sistemos integracijos testai atliekami testuotojų komandos. Sistemos integracijos metu testuojama:

- Reikalavimai sistemos suderinamumui su kitomis sistemomis;
- Aukščiausio lygio veiklos funkcionalumo reikalavimai, kur veiklos funkcijos reikalauja sistemos komunikavimo su kitomis sistemomis;
- Duomenų apdorojimo ir transakcijų vykdymo reikalavimai realioje aplinkoje;
- Reikalavimai sistemos našumui;
- „Buckup’ų” ir atsistatymo reikalavimai;

- Išorinio saugumo reikalavimai.

Sistemos integracijos testai turi būti vykdomi realioje aplinkoje panaudojant realius duomenis. Sistemos integracijos testavimui pakartotinai naudojami sistemos testavimo scenarijai.

5.1.3.3.5 Priėmimo testavimas

Priėmimo testu testuojama veiklos funkcionalumo reikalavimai. Šio testo tikslas yra parodyti vartotojų grupei, kad sistema atitinka vartotojų poreikius, veiklos funkcionalumo reikalavimus ir yra pakankamai gera numatomam naudojimui. Jei sistema ya padaryta vienam konkrečiam vartotojui, vykdomi Priėmimo testai, kurie leidžia vartotojui patikrinti visus reikalavimus. Priėmimo testai yra vykdomi vartotojų padedant testuotojų komandai. Priėmimo testai turi būti vykdomi su realiais duomenimis.

Jei naudotojų bus daug, neįmanoma įvykdyti priėmimo testų su visais. Dažniausiai tuomet naudojamas alfa ir beta testavimas, siekiant rasti klaidas, kurias gali rasti tik tikras naudotojas.

Alfa testai vykdomi pas sistemos gamintojus, stebint sistemos kūrėjams. Beta testai vykdomi pas naudotojus, natūralioje aplinkoje, sistemos kūrėjai čia nedalyvauja. Naudotojas dokumentuoja rastas klaidas, problemas ir siunčia jas sistemos kūrėjams, ko pasekoje daromi sistemos pakeitimai/modifikacijos.

5.1.3.3.6 Regresinis testavimas

Regresinis testavimas – grįžtamasis testavimas (klaidos pataisymas). Regresinio testavimo tikslas – patikrinti, kad sistema neprarado buvusio funkcionalumo, kad pataisytos rastos funkcionalumo klaidos ir kad naujas funkcionalumas veikia korektiškai. Regresinis testavimas - tai ne testavimo etapas, jis gali būti priskirtas prie testavimo technikų ir taikomas visuose testavimo etapuose.

Testuotojas, iš programuotojų gavęs klaidos ištaisymo patvirtinimą, turi patikrinti, ar anksčiau aptiktos klaidos buvo ištaisytos, ar taisant jas nebuvo padarytą naujų klaidų. Tam turi būti atliekamas regresinis testavimas. Jei yra aptinkamos naujos klaidos, klaidos registruojamos ir vėl atiduodama taisyti programuotojams.

Testavimo procesas nėra begalinis. Projekto testavimo vadovas ir (arba) testuotojas nusprendžia, ar yra tikslinga tęsti testavimo procesą. Tam gali būti naudojami testavimo

pabaigimo kriterijai.

5.1.3.4 Klaidų valdymas

5.1.3.4.1 Klaidos ir jų tipai

Programinės įrangos klaida dažnai suprantama kaip neatitikimas tarp specifikacijų ir programos. Bet tai nėra tikslus ir visą laiką tinkantis apibrėžimas. Toks klaidos apibrėžimas yra teisingas tik tada, kai pačios specifikacijos yra tikslios ne dviprasmiškos ir be klaidų. Realiai tokių specifikacijų nebūna. Tad norint tiksliau suprasti klaidos sampratą, ją apibrėšime kaip neatitikimą tarp sistemos ir vartotojo lūkesčių. Klaidos atsiranda visų pirma dėl netikslių, dviprasmiškų reikalavimų gaunamų iš vartotojo, dėl specifikacijų neaiškumo, aplaidumo ir t.t. Klaidos yra labai įvairios ir tam, kad sudarytume efektyvų testą reikia susipažinti su klaidomis ir jų pagrindiniais tipais, kokių galima tikėtis aptikti programose.

Pagrindiniai 13 klaidų tipų:

1. Vartotojo sąsajos klaidos. Visas šias klaidas dažniausiai galime apibrėžti kaip problemas, dėl kurių sunku, nepatogu dirbti su programa. Šias klaidas galima išskirti į keletą kategorijų:
    1.1 Funkcionalumo. Programa turi funkcionalumo problemų jei ji nedaro to, ką ji turi daryti arba kažką padaryti yra sunku, nepatogu, komplikuota.
    1.2 Informatyvumo. Ar pakanka informacijos lange, kad būtų galima be klaidų atlikti norimus veiksmus, ar aišku, ką kokia komanda daro.
    1.3 Komandų struktūra. Ar komandos aiškiai ir suprantamai pateiktos. Ar jos lengvai atskiriamos nuo kitų komandų.
    1.4 Trūkstamos komandos. Ar visos komandos yra? Gal galima pridėti naujų komandų, kurios palengvintų darbą su sistema.
    1.5 Greičio. Greitis yra iteraktyvių programų esmė. Viskas, kas priverčia vartotoją mąstyti, kad programa veikia lėtai yra greičio problema.
    1.6 Informacijos pateikimo klaidos. Dauguma programų rodo, spausdina, išsaugo informaciją. Ar tvarkingai, logiškai išdėstyta ši informacija? Ar informacijos kiekis pakankamas, kad patenkintų vartotojo lūkesčius? Ar ji įskaitoma ir suprantama? Ar programa išsaugo duomenis taip, kad kitos

programos galėtų juos suprasti.

1.7 Stilius. Programa turi ne tik teisingai veikti, bet ir gerai atrodyti (gramatinės klaidos, nesuprantami, neišsamūs pranešimai ir t.t.). Kartais skirtingo stiliaus GUI komponentai verčia vartotoją klaidingai galvoti, kad ir jų funkcionalumas skiriasi, ir atvirkščiai.

2. Klaidų apdorojimas. Programa turi numatyti klaidingų situacijų, nekorektiškų duomenų apdorojimą ir veiksmus tokiems atvejams atsitikus.
3. Ribinių reikšmių klaidos. Klaidos pasireiškiančios su ribinėmis reikšmėmis, būsenomis (ribos gali būti skaičių intervalo, simbolių, reikšmių aibės, laiko, atminties ir t.t.).
4. Skaičiavimų klaidos. Tai aritmetinės klaidos, sudėtingų formulių klaidingas interpretavimas, nepakankamas skaičiavimo tikslumas, neteisingi skaičiavimų algoritmai ir t.t.
5. Pradinės ir vėlesnės būsenos klaidos. Programos, kažkurios funkcijos gali veikti neteisingai, kai pirmą kartą jas paleidžiame. Tai susiję su tuo, kad daugelis programų naudoja pradinės būsenos failus (initialization files). Ir tiktai paleidus programą pirmą kartą pasireiškia klaida, nes antrą kartą programa jau susikuria ir sutvarko pradinės būsenos failus.
6. Kontrolės klaidos. Kontrolės srautas nusako, ką programa prie tam tikrų sąlygų turi daryti toliau. Kontrolės klaidos, kai programa daro sekančius veiksmus ne tuos kuriuos turėtų ar ne tada kada turėtų.
7. Duomenų interpretavimo klaidos. Vienas modulis gali perduoti duomenis kitam (pvz., sukurti juos duomenų bazėje kitam moduliui naudoti) atitinkamiems veiksmams atlikti. Dalis domenų gali būti sugrąžinta atgal ir taip kelis kartus. Šio proceso metu duomenys gali būti išgadinti arba ne taip interpretuoti ir nekorektiškai pakeisti.
8. Lygiagrečių procesų klaidos. Tarkime, turime aibę įvykių kurie gali įvykti atitinkamais laiko momentais tam tikra seka. Programa, sulaukusi tam tikro įvykio, gali laukti kito ir neteisingai ignoruoti kitus įvykius. Arba teisingai apdoroti ne visas galimas įvykių sekas.
9. Apkrovų klaidos. Programos gali nesusitvarkyti su didelėmis apkrovomis

(dideliais duomenų, lygiagrečių procesų kiekiais ir t.t.).

10. Kompiuterinės įrangos naudojimo klaidos. Programos gali ignoruoti iš kitų įrenginių ateinančias klaidas, bandyti naudoti užimtus arba nesančius įrenginius. Programos turi apdoroti visas galimas kitų įrenginių klaidas ir informuoti vartotoją apie tai.
11. Versijų kontrolės klaidos. Naudojimas vienos versijos programos dalies su kitos versijos likusia dalimi (seni ar naujesni, nesulinkuoti DLL, exe failai). Pagalbinių programų ne tos versijos ir t.t.
12. Dokumentacijos klaidos. Dokumentacija nėra programinė įranga, bet tai yra dalis jos. Neteisinga, skurdi dokumentacija, gali versti vartotoją manyti, kad programa neveikia, ar veikia neteisingai.
13. Testavimo klaidos. Programos klaidų neaptikimas yra testavimo klaidos. Dažnai tokios klaidos pasireiškia dėl informacijos, laiko trūkumo, klaidingos informacijos (Neišsami, klaidinga dizaino dokumentacija, specifikacijos ir t.t.).

5.1.4 Literatūros apžvalga

Prieš pradedant sistemos kūrimą buvo perskaityta daug literatūros straipsnių, susijusių su atliekamu darbu. Tai literatūra aprašanti informacinių sistemų bei turinio valdymo sistemų kūrimo metodologijas, konkrečių informacinių technologijų taikymus, testavimo metodikas ir kt. Nagrinėtos literatūros sąrašas pateiktas paskutiniame skyriuje.

5.2 ATLIKTI PRAKTINĖS DALIES DARBAI

Siekiant pasiekti užsibrėžtų tikslų ir pasiekti reikiamus rezultatus, magistro darbo metu buvo atlikti šie praktinės dalies darbai:

- Atlikta VITMOS IS reikalavimų analizė
- Pagal apibrėžtus reikalavimus parengta VITMOS IS testavimo scenarijai;
- Sukurta VITMOS IS programinė įranga (tam tikra dalis, tame tarpe LAS

imitatorius, realizuoti duomenų mainai tarp TIP ir LAS),

- Ištestuota sukurta programinė įranga;

Minėti darbai atlikti tam tikrų veiklos fazių metu:

- Apibrėžimas;
- Reikalavimų modeliavimas;
- Projektavimas ir generavimas;
- Diegimas ir palaikymas.

Žemiau pateikti rezultatai yra išskaidyti būtent pagal šias fazes.

Šioje lentelėje yra pateikta, kokie darbai kokioje fazėje yra atlikti.

Lentelė 1 Darbo rezultatai fazėse

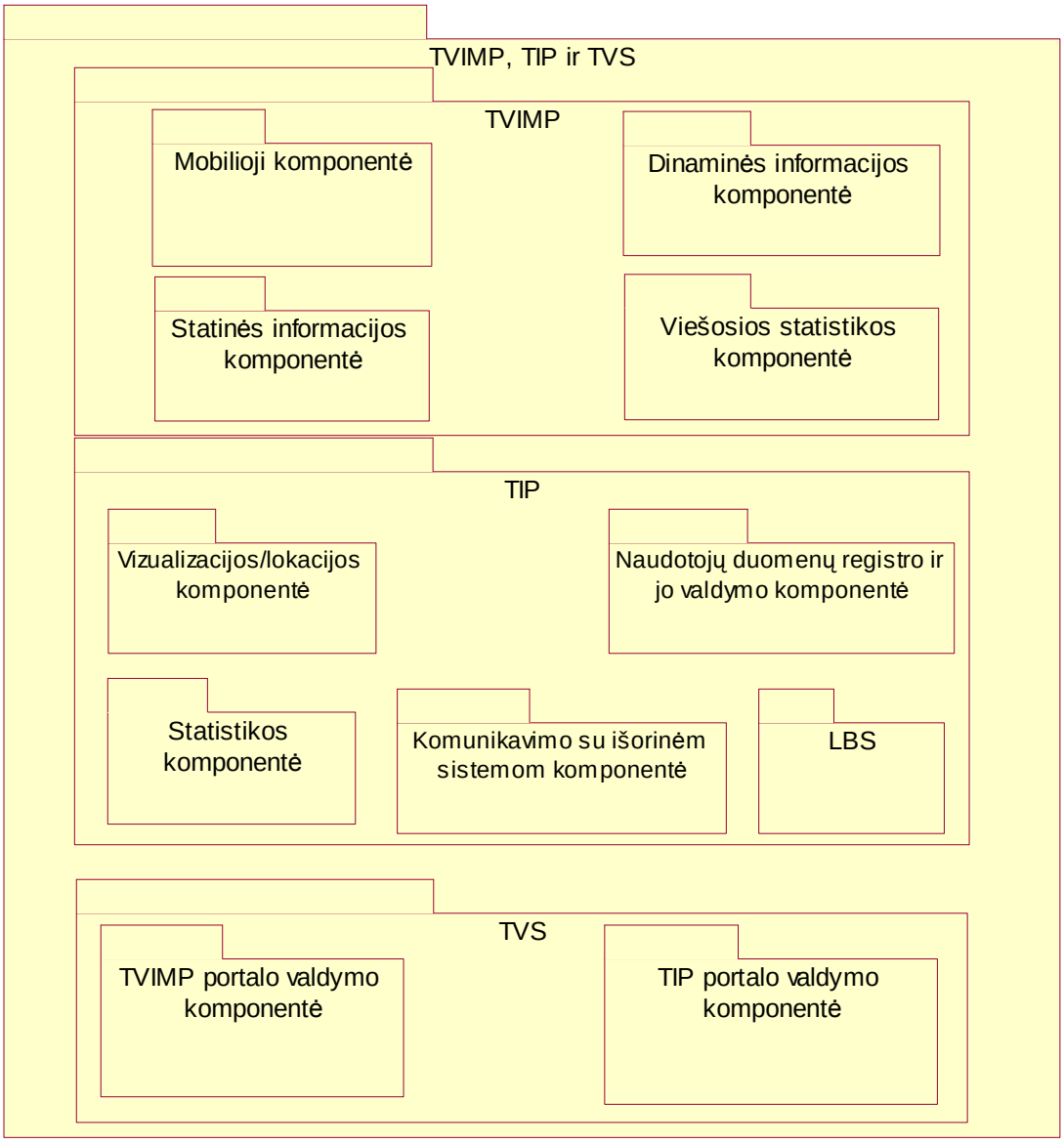
Eil. Nr.	**Fazė**	**Atlikti darbai**
1	Apibrėžimas	Atlikta VITMOS IS reikalavimų analizė
2	Reikalavimų modeliavimas	Atlikta VITMOS IS reikalavimų analizė
3	Projektavimas ir generavimas	Pagal apibrėžtus reikalavimus parengta VITMOS IS testavimo scenarijai
		Sukurta VITMOS IS programinė įranga (tam tikra dalis, tame tarpe LAS imitatorius, realizuoti duomenų mainai tarp TIP ir LAS)
		Ištestuota sukurta programinė įranga
4	Diegimas ir palaikymas	Įdiegta testinė versija į testinį serverį

5.2.1 Apibrėžimas

5.2.1.1 Sistemos apibrėžimas

5.2.1.1.1 Bendra sistemos struktūra ir dalys

Žemiau yra pateiktos TVIMP ir TIP bei jų TVS dalys ir komponentės.



Paveikslėlis 1 - TVIMP ir TIP bei jų TVS dalys ir komponentės

TIP portalas – tai posistemė, skirta informacijos apie transporto priemones pateikimui. Portalo vartotojai galės ne tik gauti reikiamą informaciją, bet naudotis papildomų modulių

suteikiamomis funkcijomis. Ši sistema skirta portalo registruotiems portalo vartotojams. Jie gali dirbti su papildomais portalo moduliais bei posistemėmis. Pavyzdžiui portalo lankytojas galės ne tik skaityti naujienas bei įvairius kitus straipsnius, bet ir prisijungęs prie portalo stebėti savo trasporto priemonės judėjimą ar gauti kokią kitą reikiamą informaciją, kurios pateikimą užtikrins papildomi sistemos moduliai bei posistemės. Kokias funkcijas realizuoja ši posistemė yra aprašyta skyriuje „5.2.2.1“

TIP komponentės:

- statistikos komponentė – pateikia statistinius duomenis apie transporto ir krovinių judėjimą;
- vizualizacijos/lokacijos komponentė – vykdo užklausas ir pateikia tekstinę ir vizualią informaciją iš lokacijos serverio (LAS) komponentės apie objektų lokaciją;
- naudotojų duomenų registro ir jo valdymo komponentė – registruoja portalo naudotojus, leidžia registruotiems vartotojams įvesti/pateikti duomenis apie savo kompanijos sekamas transporto priemones, gabenamus krovinius ir pan;
- komunikavimo su išorinėmis sistemomis komponentė – komunikavimo sąsaja tarp portalo ir kitų informacinių sistemų;
- LBS (location based service) komponentė diegia ir teikia įvairiarūšes, su naudotojo vietos nustatymu susijusiąs paslaugas.

TVIMP portalas (transporto viešosios informacijos mobilus portalas) skirtas visiems portalo lankytojams. Tai posistemė atliekanti viešosios informacijos pateikimo portalo lankytojams funkcijas. Kokias funkcijas realizuoja ši posistemė yra aprašyta skyriuje „5.2.2.1“.

TVIMP komponentės:

- mobilioji komponentė – suteikia galimybę naršyti portalą mobiliuoju telefonu, kitomis priemonėmis, naudojant WWW naršyklę arba WAP tipo naršyklę;
- statinės informacijos komponentė – surenka ir automatiškai atnaujina statinius duomenis (dokumentus, nuorodas ir t.t.);
- dinaminės informacijos komponentė – pateikia dinaminę informaciją sukauptą portale:

naujienas, pranešimus/skelbimus iš transporto kompanijų, paslaugas ir pan;

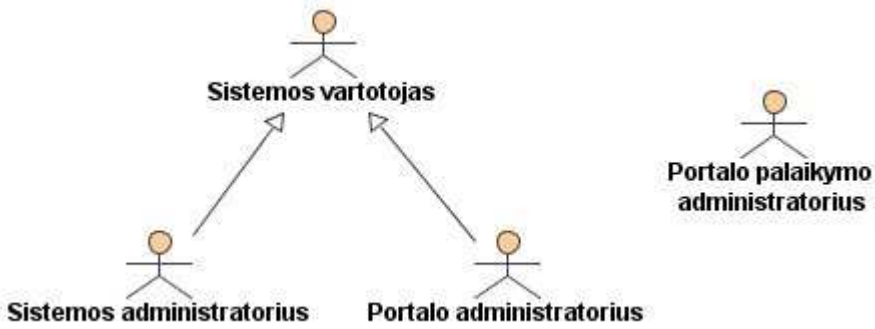
- viešosios statistikos komponentė – pateikia viešą apibendrintą statistiką apie transporto srautus, bendrą kelių naudojimą ir pan.

TVS – tai portalo turinio valdymo sistema, skirta portalo bei pačios turinio valdymo sistemos administratoriui. TVS skirta tiek jos pačios vartotojų administravimui tiek portalo valdymui. Dalis TVS funkcionalumo yra skirta išorinių posistemių bei modulių prijungimui prie portalo. Vienas iš svarbiausių reikalavimų šiai sistemai – galimybė papildyti portalą papildomais moduliais arba smulkiomis posistemėmis, kurios veiktų portalo aplinkoje.

TVS komponentės:

- TVIMP portalo valdymo komponentė – skirta TVIMP portalo modulių ir vartotojų valdymui;
- TIP portalo valdymo komponentė - skirta TIP portalo modulių ir vartotojų valdymui;

5.2.1.2 Sistemos vartotojai



Paveikslėlis 2 sistemos vartotojai

Sistemos administratorius – vartotojas, administruojantis turinio valdymo sistemą.

Portalo administratorius – vartotojas, dirbantis su turinio valdymo sistema. Jis administruoja portalą.

"Sistemos administratorius" bei "Portalo administratorius" bendrai vadinami "Sistemos vartotojais".

Portalo palaikymo administratorius – programuotojas, kuriantis bei diegiantis papildomus portalo modulius, diegiamus į turinio valdymo sistemą.

5.2.2 Reikalavimų modeliavimas

5.2.2.1 Sistemos funkcijų hierarchija (MoSCoW sąrašas)

Remiantis “Oracle CDM fast track” – informacinių sistemų kūrimo metodologija, kaip aprašyta skyriuje 5.1.2.2.2, visos sistemos funkcijos suskirstomos į prioritetus pagal svarbą. Pagal priioritetus jas galima skirstyti į 4 grupes: P - privalo būti, T – turi būti, G – gali būti, N – neturėtų būti. (Angliškai M – must be, S – should be, C – could be, W – would not be). Kadangi labai dažnai nėra spėjama realizuoti visų funkcijų laiku, todėl prioritetas skiriamas svarbiausioms iš jų.

Atlikus TVIMP ir TIP reikalavimų analizę, buvo sudaryta tokia funkcijų hierarchija – funkcijų sąrašas, nustatyti funkcijų prioritetai.

Funkcijų sąrašas pateiktas priede „A.1“.

5.2.2.2 Sistemos modulių sąrašas

Funkcijos yra realizuojamos žemiau pateiktuose moduliuose.

Lentelė 1 TVIMP modulių sąrašas

Modulio kodas	Modulio pavadinimas
M_TVIMP_101	Portalo vartotojų grupės registravimo/redagavimo/šalinimo/peržiūros forma
M_TVIMP_102	Portalo vartotojo registravimo/redagavimo/peržiūros forma
M_TVIMP_103	Portalo meniu punkto registravimo/redagavimo/peržiūros forma
M_TVIMP_104	Portalo modulio registravimo/redagavimo/peržiūros forma
M_TVIMP_105	Portalo puslapio registravimo/redagavimo/peržiūros forma
M_TVIMP_106	Portalo puslapių šablono registravimo/redagavimo/peržiūros forma
M_TVIMP_107	Portalo grupės teisių į modulį registravimo/redagavimo/peržiūros forma

M_TVIMP_108	Portalo vartotojo teisių į modulį registravimo/redagavimo/peržiūros forma
M_TVIMP_109	Portalo TVS vartotojo registravimo/redagavimo/peržiūros forma
M_TVIMP_110	Portalo dokumentų atnaujinimo nustatymų redagavimo forma
M_TVIMP_111	Portalo nuorodų atnaujinimo nustatymų redagavimo forma
M_TVIMP_112	Portalo naujienų registravimo/redagavimo/peržiūros forma
M_TVIMP_113	Portalo skelbimų registravimo/redagavimo/peržiūros forma
M_TVIMP_201	Vartotojų grupių sąrašo peržiūros forma
M_TVIMP_202	Vartotojų sąrašo peržiūros forma
M_TVIMP_203	Meniu sąrašo peržiūros forma
M_TVIMP_204	Modulių sąrašo peržiūros forma
M_TVIMP_205	Puslapių sąrašo peržiūros forma
M_TVIMP_206	Vartotojų grupės teisių į modulį sąrašo peržiūros forma
M_TVIMP_207	Vartotojų teisių į modulį sąrašo peržiūros forma
M_TVIMP_208	TVS vartotojų sąrašo peržiūros forma
M_TVIMP_209	Portalo dokumentų peržiūros forma
M_TVIMP_210	Portalo nuorodų peržiūros forma
M_TVIMP_211	Portalo naujienų sąrašo peržiūros forma
M_TVIMP_212	Portalo dokumentų sąrašo peržiūros forma
M_TVIMP_213	Portalo skelbimų sąrašo peržiūros forma
M_TVIMP_214	Peržiūrėti viešosios statistikos informaciją apie transporto srautus, bendrą kelių naudojimą

Lentelė 2 TIP modulių sąrašas

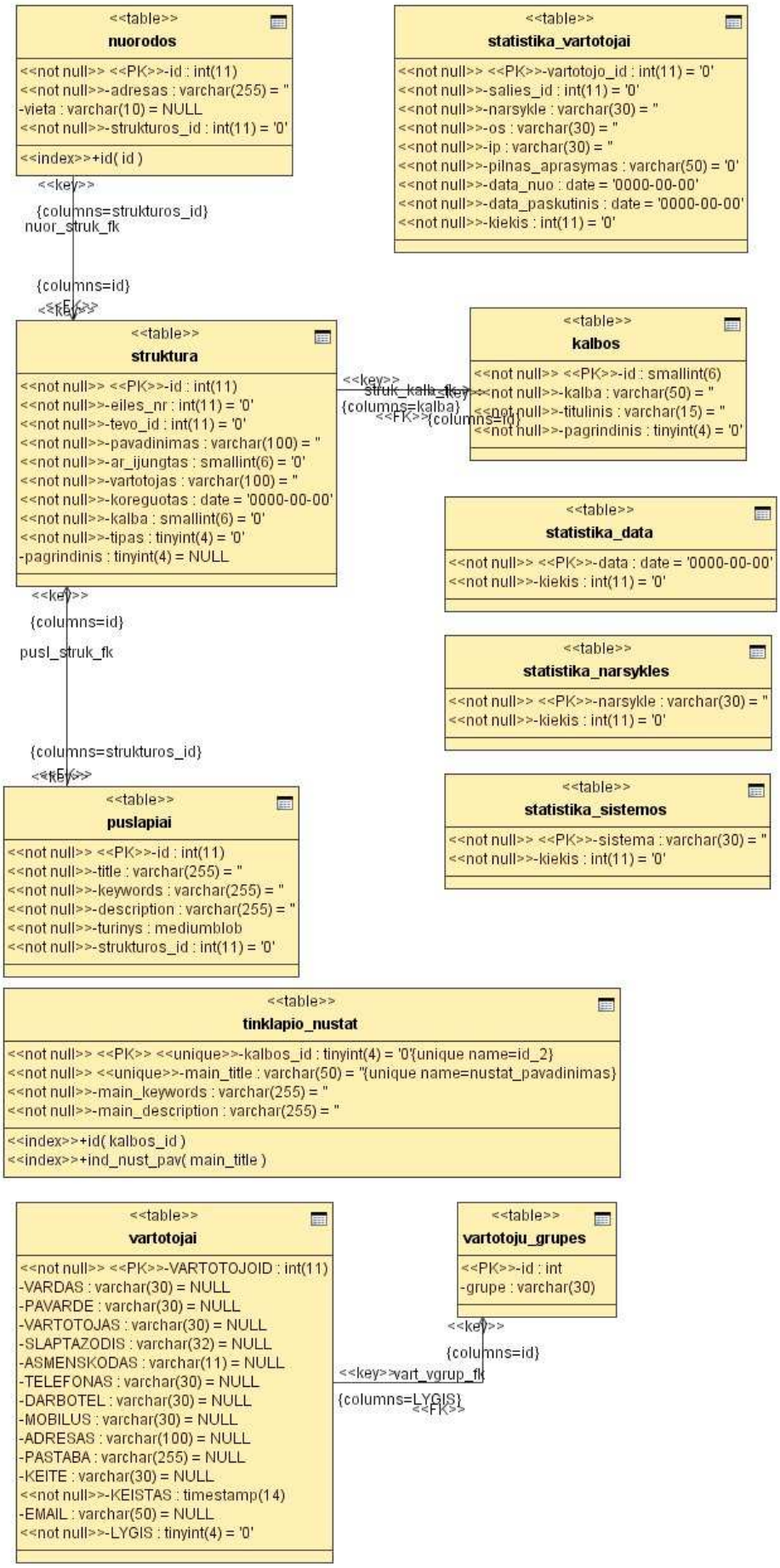
Modulio kodas	Modulio pavadinimas
M_TIP_101	Šalies registravimo/redagavimo/šalinimo/peržiūros forma
M_TIP_102	Miesto registravimo/redagavimo/šalinimo/peržiūros forma
M_TIP_103	Krovinio registravimo/redagavimo/šalinimo/peržiūros forma
M_TIP_104	Krovinių priėmėjo registravimo/redagavimo/šalinimo/peržiūros forma
M_TIP_105	Vairuotojo registravimo/redagavimo/šalinimo/peržiūros forma
M_TIP_106	Kliento registravimo/redagavimo/šalinimo/peržiūros forma
M_TIP_107	Maršruto registravimo/redagavimo/šalinimo/peržiūros forma
M_TIP_108	Objekto sekimo registravimo/redagavimo/šalinimo/peržiūros forma
M_TIP_201	Šalių sąrašo peržiūros forma
M_TIP_202	Miestų sąrašo peržiūros forma
M_TIP_203	Krovinių sąrašo peržiūros forma
M_TIP_204	Krovinių priėmėjų sąrašo peržiūros forma
M_TIP_205	Vairuotojų sąrašo peržiūros forma
M_TIP_206	Klientų sąrašo peržiūros forma
M_TIP_207	Maršrutų sąrašo peržiūros forma
M_TIP_208	Sekamo objekto buvimo vietos peržiūros forma
M_TIP_209	Sekamų objektų sekimų sąrašo peržiūros forma

5.2.3 Projektavimas ir generavimas

5.2.3.1 TVIMP ir TIP portalų bei jų TVS projektavimas ir programavimas

Duomenų bazės modelio sudarymas

Pagal analizės metu pateiktus reikalavimus buvo suprojektuotas pradinis duomenų bazės modelis. Duomenų bazės struktūra kuriama taip, kad portalo straipsnius ir nuorodas būtų galima atvaizduoti neriboto dydžio medžio struktūra. Žemiau yra pateikta tam tikros sistemos dalies duomenų bazės struktūra.


<<table>>
nuorodos
<<not null>> <<PK>>-id : int(11)
<<not null>>-adresas : varchar(255) = ''
-vieta : varchar(10) = NULL
<<not null>>-strukturos_id : int(11) = '0'
<<index>>+id(id)
<<key>>
{columns=strukturos_id}
nuor_struk_fk
{columns=id}
<<FK>>
<<table>>
statistika_vartotojai
<<not null>> <<PK>>-vartotojo_id : int(11) = '0'
<<not null>>-salies_id : int(11) = '0'
<<not null>>-narsykle : varchar(30) = ''
<<not null>>-os : varchar(30) = ''
<<not null>>-ip : varchar(30) = ''
<<not null>>-pilnas_aprasymas : varchar(50) = '0'
<<not null>>-data_nuo : date = '0000-00-00'
<<not null>>-data_paskutinis : date = '0000-00-00'
<<not null>>-kiekis : int(11) = '0'
<<table>>
struktura
<<not null>> <<PK>>-id : int(11)
<<not null>>-eiles_nr : int(11) = '0'
<<not null>>-tevo_id : int(11) = '0'
<<not null>>-pavadinimas : varchar(100) = ''
<<not null>>-ar_ijungtas : smallint(6) = '0'
<<not null>>-vartotojas : varchar(100) = ''
<<not null>>-koreguotas : date = '0000-00-00'
<<not null>>-kalba : smallint(6) = '0'
<<not null>>-tipas : tinyint(4) = '0'
-pagrindinis : tinyint(4) = NULL
<<key>>
{columns=kalba}
<<FK>>
<<table>>
kalbos
<<not null>> <<PK>>-id : smallint(6)
<<not null>>-kalba : varchar(50) = ''
<<not null>>-titulinis : varchar(15) = ''
<<not null>>-pagrindinis : tinyint(4) = '0'
<<table>>
statistika_data
<<not null>> <<PK>>-data : date = '0000-00-00'
<<not null>>-kiekis : int(11) = '0'
<<table>>
statistika_narsykles
<<not null>> <<PK>>-narsykle : varchar(30) = ''
<<not null>>-kiekis : int(11) = '0'
<<table>>
statistika_sistemos
<<not null>> <<PK>>-sistema : varchar(30) = ''
<<not null>>-kiekis : int(11) = '0'
<<key>>
{columns=id}
pusl_struk_fk
{columns=strukturos_id}
<<FK>>
<<table>>
puslapiai
<<not null>> <<PK>>-id : int(11)
<<not null>>-title : varchar(255) = ''
<<not null>>-keywords : varchar(255) = ''
<<not null>>-description : varchar(255) = ''
<<not null>>-turinys : mediumblob
<<not null>>-strukturos_id : int(11) = '0'
<<table>>
tinklapio_nustat
<<not null>> <<PK>> <<unique>>-kalbos_id : tinyint(4) = '0'{unique name=id_2}
<<not null>> <<unique>>-main_title : varchar(50) = ''{unique name=nustat_pavadinimas}
<<not null>>-main_keywords : varchar(255) = ''
<<not null>>-main_description : varchar(255) = ''
<<index>>+id(kalbos_id)
<<index>>+ind_nust_pav(main_title)
<<table>>
vartotojai
<<not null>> <<PK>>-VARTOTOJOID : int(11)
-VARDAS : varchar(30) = NULL
-PAVARDE : varchar(30) = NULL
-VARTOTOJAS : varchar(30) = NULL
-SLAPTAZODIS : varchar(32) = NULL
-ASMENSKODAS : varchar(11) = NULL
-TELEFONAS : varchar(30) = NULL
-DARBOTEL : varchar(30) = NULL
-MOBILUS : varchar(30) = NULL
-ADRESAS : varchar(100) = NULL
-PASTABA : varchar(255) = NULL
-KEITE : varchar(30) = NULL
<<not null>>-KEISTAS : timestamp(14)
-EMAIL : varchar(50) = NULL
<<not null>>-LYGIS : tinyint(4) = '0'
<<table>>
vartotoju_grupes
<<PK>>-id : int
-grupe : varchar(30)
<<key>>
{columns=id}
vart_vgrup_fk
{columns=LYGIS}
<<FK>>

Paveikslėlis 3 – DB schema

5.2.3.2 Testavimo scenarijai

Šiame skyriuje apibrėžiami testavimo scenarijai, kurie naudojami atliekant funkcinį programinės įrangos testavimą sistemos testavimo metu. Šie scenarijai taip pat naudojami vertinant sukurtos sistemos kokybę. Jie padeda spręsti ne tik kokybės užtikrinimo, bet ir sistemos perdavimo jos naudotojui problemas. Testavimo scenarijai parengiami bei sistema yra testuojama projektavimo ir generavimo fazės metu.

Testavimo scenarijai sudaryti naudojantis sumodeliuotais veiklos procesais, kurie sudaryti iš sistemos funkcijų. Kiekvienam testavimo scenarijui aprašyti testavimo žingsniai.

Sistemos funkcionalumo testavimo scenarijai

5.2.3.2.1.1 Transporto viešosios informacijos mobilaus portalo funkcionalumo testavimas

Transporto viešosios informacijos mobilaus portalo testavimo scenarijai yra pateikti priede „A.2“.

5.2.3.2.1.2 Transporto informacijos portalo funkcionalumo testavimas

Transporto informacijos portalo testavimo scenarijai yra pateikti priede „A.3“

5.2.3.2.1.3 Transporto informacijos portalo sąsajos su LAS testavimas

Lentelė 2 TIP_LAS1.Vykdyti duomenų mainus tarp LAS ir TIP

Proceso NR:	**TIP_LAS1**
Proceso pavadinimas:	**TIP_LAS1.Vykdyti duomenų mainus tarp LAS ir TIP**
Scenarijaus NR:	TIP_LAS1_1
Scenarijaus pavadinimas:	Priimti, apdoroti ir įrašyti duomenis iš LAS

5.2.3.2.2 Detalus testavimas pagal testavimo scenarijus

Parengti testavimo scenarijai yra detalizuojami. Detaliuose scenarijuose yra įvardinami:

- proceso numeris ir pavadinimas;
- scenarijaus numeris ir pavadinimas;
- papildomi testavimo reikalavimai;
- modulis, kuris yra testuojamas;
- veiksmai, kuriuos reikia atlikti norint įvykdyti scenarijų;
- scenarijaus įgyvendinimo atvejai.

Sistemos kokybei tikrinti yra pasirinktas funkcionalumo testavimo būdas.

5.2.3.2.2.1 Transporto viešosios informacijos mobilaus portalo funkcionalumo testavimas

TVIMP1. Administruoti transporto viešosios informacijos portalo vartotojus ir jų grupes

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos mobilaus portalo vartotojus ir jų grupes**
	Scenarijaus NR:	TVIMP1_1
	Scenarijaus pavadinimas:	Registruoti portalo vartotojų grupę
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos administratoriaus teisėmis
Modulis		M_TVIMP_1 – Portalo vartotojų grupės registravimo forma
Veiksmas		1. Paleisti modulį M_TVIMP_1 2. Užpildyti grupės pavadinimą ir grupės aprašymą 3. Išsaugoti (Paspausti OK)

Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP1_1_1	Reikia registruoti portalo vartotojų grupę	1. Užregistruota portalo vartotojų grupė 2. Išsaugoti jo atributai pavadinimas ir grupės aprašymas

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos mobilaus portalo vartotojus ir jų grupes**
	Scenarijaus NR:	TVIMP1_2
	Scenarijaus pavadinimas:	Peržiūrėti portalo vartotojų grupių sąrašą
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos administratoriaus teisėmis
Modulis		M_TVIMP_201 – Vartotojų grupių sąrašo peržiūros forma
Veiksmas		1. Paleisti modulį M_TVIMP_201 2. Atlikti sąrašo filtravimą pagal visus esamus filtravimo laukus

Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP1_2_1	Reikia surasti užregistruotą portalo vartotojų grupę	1. Sąraše surasta ieškoma portalo vartotojų grupė

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos mobilaus portalo vartotojus ir jų grupes**
	Scenarijaus NR:	TVIMP1_3
	Scenarijaus pavadinimas:	Redaguoti portalo vartotojų grupės duomenis
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos administratoriaus teisėmis
Modulis		M_TVIMP_101 – Portalo vartotojų grupės registravimo/redagavimo/šalinimo/peržiūros forma

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos mobilaus portalo vartotojus ir jų grupes**
Veiksmas	2. Surasti redaguojamą vartotojų grupę (modulis M_TVIMP_201) 3. Paleisti modulį M_TVIMP_101 4. Redaguoti laukus 5. Išsaugoti	
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP1_3_1	Reikia redaguoti portalo vartotojų grupę	6. Paredaguota portalo vartotojų grupė

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos mobilaus portalo vartotojus ir jų grupes**
	Scenarijaus NR:	TVIMP1_4
	Scenarijaus pavadinimas:	Registruoti portalo vartotoją
Papildomi reikalavimai	• Prisijungti prie turinio valdymo sistemos administratoriaus teisėmis	
Modulis	M_TVIMP_102 – Portalo vartotojo registravimo/redagavimo/šalinimo/peržiūros forma	
Veiksmas	1. Paleisti modulį M_TVIMP_102 2. Užpildyti privalomus laukus 3. Išsaugoti (Paspausti OK)	
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP1_4_1	Reikia užregistruoti portalo vartotoją	1. Užregistruotas portalo vartotojas 2. Išsaugoti jo privalomi atributai

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos mobilaus portalo vartotojus ir jų grupes**
	Scenarijaus NR:	TVIMP1_5
	Scenarijaus pavadinimas:	Peržiūrėti portalo vartotojų sąrašą
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos administratoriaus teisėmis
Modulis		M_TVIMP_202 – Vartotojų sąrašo peržiūros forma
Veiksmas		1. Paleisti modulį M_TVIMP_202 2. Atlikti sąrašo filtravimą pagal visus esamus filtravimo laukus
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP1_5_1	Reikia surasti užregistruotą portalo vartotoją	1. Sąraše surastas ieškomas portalo vartotojas

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos portalo vartotojus ir jų grupes**
	Scenarijaus NR:	TVIMP1_6
	Scenarijaus pavadinimas:	Redaguoti portalo vartotojo duomenis
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos administratoriaus teisėmis
Modulis		M_TVIMP_102 – Portalo vartotojų grupės registravimo/redagavimo/šalinimo/peržiūros forma
Veiksmas		1. Surasti redaguojamą vartotoją (modulis M_TVIMP_202) 2. Paleisti modulį M_TVIMP_102 3. Redaguoti laukus 4. Išsaugoti
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos portalo vartotojus ir jų grupes**
TVIMP1_6_1	Reikia redaguoti portalo vartotoją	1. Paredaguotas portalo vartotojas

Proceso NR:		**TVIMP1**
Proceso pavadinimas:		**TVIMP1. Administruoti transporto viešosios informacijos portalo vartotojus ir jų grupes**
	Scenarijaus NR:	TVIMP1_7
	Scenarijaus pavadinimas:	Registruoti TVS vartotoją
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos administratoriaus teisėmis
Modulis		M_TVIMP_109 – Portalo TVS vartotojo registravimo/redagavimo/peržiūros forma
Veiksmas		1. Paleisti modulį M_TVIMP_109 2. Užpildyti privalomus laukus 3. Išsaugoti (Paspausti OK)
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP1_7_1	Reikia užregistruoti portalo TVS vartotoją	1. Užregistruotas portalo TVS vartotojas 2. Išsaugoti jo privalomi atributai

TVIMP2.Administruoti portalo modulius, meniu punktus, puslapius

Proceso NR:		**TVIMP2**
Proceso pavadinimas:		**TVIMP2.Administruoti portalo modulius, meniu punktus, puslapius**
	Scenarijaus NR:	TVIMP2_1
	Scenarijaus pavadinimas:	Registruoti meniu punktą
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos

Proceso NR:		**TVIMP2**
Proceso pavadinimas:		**TVIMP2.Administruoti portalo modulius, meniu punktus, puslapius**
Modulis		M_TVIMP_103 – Portalo meniu punkto registravimo/redagavimo/peržiūros forma
Veiksmas		1. Paleisti modulį M_TVIMP_103 2. Užpildyti privalomus laukus 3. Išsaugoti (Paspausti OK)
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP2_1_1	Reikia registruoti meniu punktą	1. Užregistruotas meniu punktas 2. Išsaugoti jo atributai

Proceso NR:		**TVIMP2**
Proceso pavadinimas:		**TVIMP2.Administruoti portalo modulius, meniu punktus, puslapius**
	Scenarijaus NR:	TVIMP2_2
	Scenarijaus pavadinimas:	Registruoti modulį
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos
Modulis		M_TVIMP_104 – Portalo modulio registravimo/redagavimo/peržiūros forma
Veiksmas		1. Paleisti modulį M_TVIMP_104 2. Užpildyti privalomus laukus 3. Išsaugoti (Paspausti OK)
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP2_2_1	Reikia registruoti modulį	1. Užregistruotas portalo modulis 2. Išsaugoti jo atributai

Proceso NR:		**TVIMP2**
Proceso pavadinimas:		**TVIMP2.Administruoti portalo modulius, meniu punktus, puslapius**
	Scenarijaus NR:	TVIMP2_3
	Scenarijaus pavadinimas:	Registruoti puslapio šabloną
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos
Modulis		M_TVIMP_106 – Portalo puslapių šablono registravimo/redagavimo/peržiūros forma
Veiksmas		1. Paleisti modulį M_TVIMP_106 2. Užpildyti privalomus laukus 3. Išsaugoti (Paspausti OK)
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TVIMP2_3_1	Reikia registruoti puslapio šabloną	1. Užregistruotas puslapio šabloną 2. Išsaugoti jo atributai

Proceso NR:		**TVIMP2**
Proceso pavadinimas:		**TVIMP2.Administruoti portalo modulius, meniu punktus, puslapius**
	Scenarijaus NR:	TVIMP2_4
	Scenarijaus pavadinimas:	Registruoti puslapį
Papildomi reikalavimai		• Prisijungti prie turinio valdymo sistemos
Modulis		M_TVIMP_105 – Portalo puslapio registravimo/redagavimo/peržiūros forma
Veiksmas		1. Paleisti modulį M_TVIMP_105 2. Užpildyti privalomus laukus 3. Išsaugoti (Paspausti OK)
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**

Proceso NR:		**TVIMP2**
Proceso pavadinimas:		**TVIMP2.Administruoti portalo modulius, meniu punktus, puslapius**
TVIMP2_4_1	Reikia registruoti puslapį	1. Užregistruotas puslapis 2. Išsaugoti jo atributai

5.2.3.2.2.2 Transporto informacijos portalo funkcionalumo testavimas

Proceso NR:		**TIP2**
Proceso pavadinimas:		**TIP2.Administruoti pervežamų objektų sekimą**
	Scenarijaus NR:	TIP2_1
	Scenarijaus pavadinimas:	Registruoti kliento objekto sekimą
Papildomi reikalavimai		• Prisijungti prie TIP
Modulis		M_TIP_108 – Objekto sekimo registravimo/redagavimo/šalinimo/peržiūros forma
Veiksmas		1. Paleisti modulį M_TIP_108 2. Užpildyti privalomus laukus 3. Išsaugoti (Paspausti OK)
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TIP2_1_1	Reikia registruoti objekto sekimą	1. Užregistruotas objekto sekimas 2. Išsaugoti jo atributai

TIP2.Administruoti pervežamų objektų sekimą

Proceso NR:		**TIP2**
Proceso pavadinimas:		**TIP2.Administruoti pervežamų objektų sekimą**
	Scenarijaus NR:	TIP2_2
	Scenarijaus pavadinimas:	Peržiūrėti sekamų objektų sąrašą
Papildomi reikalavimai		• Prisijungti prie TIP
Modulis		M_TIP_209 – Sekamų objektų sekimų sąrašo peržiūros forma

Proceso NR:		**TIP2**
Proceso pavadinimas:		**TIP2.Administruoti pervežamų objektų sekimą**
Veiksmas	1. Paleisti modulį M_TIP_209 2. Filtruoti sąrašą pagal visus filtravimo laukus 3. Iškviesti modulį iš formos: • Iškviesti redagavimo formą (M_TIP_108); • Iškviesti sekimo peržiūros formą (M_TIP_208);	
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TIP2_1_1	Reikia surasti objekto sekimą ir iškviesti redagavimo formą	1. Surastas objekto sekimas ir iškviestas surasto sekimo redagavimo forma
TIP2_1_2	Reikia surasti objekto sekimą ir iškviesti sekimo peržiūros formą	1. Surastas objekto sekimas ir iškviesta surasto sekimo peržiūros forma

Proceso NR:		**TIP2**
Proceso pavadinimas:		**TIP2.Administruoti pervežamų objektų sekimą**
	Scenarijaus NR:	TIP2_3
	Scenarijaus pavadinimas:	Redaguoti kliento objekto sekimo duomenis
Papildomi reikalavimai	• Prisijungti prie TIP	
Modulis	M_TIP_108 – Objekto sekimo registravimo/redagavimo/šalinimo/peržiūros forma	
Veiksmas	1. Surasti objekto sekimą (M_TIP_209) 2. Paleisti modulį M_TIP_108 3. Redaguoti laukus 4. Išsaugoti	
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**

Proceso NR:		**TIP2**
Proceso pavadinimas:		**TIP2.Administruoti pervežamų objektų sekimą**
TIP2_1_1	Reikia redaguoti objekto sekimo duomenis	1. Paredaguoti ir išsaugoti objekto sekimo duomenis

Proceso NR:		**TIP2**
Proceso pavadinimas:		**TIP2.Administruoti pervežamų objektų sekimą**
	Scenarijaus NR:	TIP2_4
	Scenarijaus pavadinimas:	Peržiūrėti sekamo objekto buvimo vietą
Papildomi reikalavimai		• Prisijungti prie TIP
Modulis		M_TIP_208 – Sekamo objekto buvimo vietos peržiūros forma
Veiksmas		1. Surasti objekto sekimą (M_TIP_209) 2. Paleisti modulį M_TIP_208 3. Peržiūrėti sekimo duomenis
Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TIP2_1_1	Reikia peržiūrėti sekamo objekto buvimo vietą ir judėjimo duomenis	1. Peržiūrėta sekamo objekto buvimo vieta ir judėjimo duomenys

5.2.3.2.2.3 Transporto informacijos portalo sąsajos su LAS testavimas

TIP_LAS1.Vykdyti duomenų mainus tarp LAS ir TIP

Proceso NR:		**TIP_LAS1**
Proceso pavadinimas:		**TIP_LAS1.Vykdyti duomenų mainus tarp LAS ir TIP**
	Scenarijaus NR:	TIP_LAS1_1
	Scenarijaus pavadinimas:	Priimti, apdoroti ir įrašyti duomenis iš LAS

Proceso NR:	**TIP_LAS1**
Proceso pavadinimas:	**TIP_LAS1.Vykdyti duomenų mainus tarp LAS ir TIP**
Papildomi reikalavimai	• Parengti testinius duomenis (objekto koordinates) • Testinius duomenis įrašyti į LAS serverio imitatorių • Sukonfigūruoti portalą, kad duomenis apie sekamą objektą imtų iš LAS imitatoriaus • Prisijungti prie TIP
Modulis	M_TIP_208 – Sekamo objekto buvimo vietos peržiūros forma
Veiksmas	1. Surasti objekto sekimą (M_TIP_209) 2. Paleisti modulį M_TIP_208 3. Peržiūrėti sekimo duomenis. 4. Sulyginti gautus sekimo duomenis su testiniais duomenimis, pateiktais LAS imitatoriui

Testavimo scenarijaus variantai		
Varianto NR	**Sąlygos prieš veiksmą**	**Tikėtinas rezultatas**
TIP_LAS1_1_1	Reikia patikrinti duomenų mainus su LAS panaudojant LAS imitatorių	1. Gauti sekimo duomenys sutampa su testiniais duomenimis, pateiktais LAS imitatoriui

6. IŠVADOS

- Kuriant pagal Oracle CDM Fastrack informacinių sistemų kūrimo metodiką, buvo galima kokybiškai valdyti sistemos kūrimo procesą - aiškiai atskirti kiekvieną IS kūrimo fazę ir joms priklausančias užduotis, apibrėžti kiekvienos fazės gyvavimo laiką. Metodikos reikalavimas aiškiai ir detaliai įvardinti veiklos procesus ir sistemos funkcijas suteikiant joms prioritetus leido tiek kuriamos sistemos galutiniam vartotojui, tiek kūrėjui tiksliai suvokti jos funkcionalumą bei veikimo galimybes, sutarti kokios funkcijos yra privalomos ir kokios neprivalomos. Projektavimo ir programavimo darbų atlikimas pagal metodikos reikalavimus pagreitino darbų atlikimą, sumažino klaidų kiekį.

- Remiantis pasirinktos testavimo metodikos siūlomu funkcionalumo testavimu ir sukurtais testavimo scenarijais tiksliai apibrėžiant, pagal ką sistema yra testuojama ir kokie yra tikėtini kiekvieno sistemos veikimo proceso rezultatai, leido dar geriau ir iki smulkių detalių suprasti jos veikimą. Kūrėjui ir sistemos galutiniam vartotojui sutarus, kad šie scenarijai yra tinkami sistemos priėmimui, atliktas testavimas. Pagal testavimo teigiamus rezultatus nebeliko abejonių dėl sistemos kokybės ir tinkamumo.

- Pasirinktų kūrimo ir testavimo metodikų taikymas įgalino ne tik efektyviai valdyti sistemos kūrimo procesą, bet ir leido išspręsti santykių su galutiniu sistemos vartotoju formuojant užduotis ir perduodant rezultatus problemas.

- Pasirinkti vieną ar kitą IS kūrimo ir testavimo metodiką reikia atsižvelgiant į kuriamos sistemos funkcijų kiekį, sudėtingumą, programų kūrimo technologijas, interfeisų įvairovę, kokybės ir kitus reikalavimus.

7. LITERATŪROS SĄRAŠAS:

[Bur03] I. Burstein. *Practical Software Testing*. Springer-Verlag, New York, 2003.

[Cha06] J. Chateriee. *Working with Pop Ups and Browser Windows in JavaScript.* URL:http://www.devarticles.com/c/a/JavaScript/Working-with-Pop-Ups-and-Browser-Windows-in-JavaScript/. 2006.

[DR99] J. S. Dumas, J. C. Redish**.** *A Practical Guide to Usability Testing.* Intellect Books, Exeter, 1999.

[Ger06] A. Gervasio. *Abstract Classes in PHP: Introducing the Key Concepts.* URL: http://www.devshed.com/c/a/PHP/Abstract-Classes-in-PHP-Introducing-the-Key-Concepts/ . 2006.

[Goo03] D. Goodman. *JavaScript & DHTML Cookbook.* O' Reilly Media, Sebastopol, 2003.

[Hud05] P. Hudson. *Php in a Nutshell.* O' Reilly Media, Sebastopol, 2005.

[Ivp04] *Reikalavimai Valstybės inormacinių sistemų specifikacijoms.* Informacinės visuomenės plėtros komitetas prie LR Vyriausybės 2004 m. spalio 15 d. įsakymas Nr. T-131. Valstybės žinios, 2004 , Nr. 155-5679.

[Ivp04] *Valstybės informacinių sistemų kūrimo metodika.* Informacinės visuomenės plėtros komitetas prie LR Vyriausybės 2004 m. spalio 15 d. įsakymas Nr. T-131. Valstybės žinios, 2004, Nr. 155-5679.

[Ler03] R. Lerdorf. *PHP Pocket Reference, 2nd Edition.* O' Reilly Media, Sebastopol, 2003.

[LW04] D. Lane, H. E. Williams. *Web Database Applications with PHP and MySQL, 2nd Edition.* O' Reilly Media, Sebastopol, 2004.

[Mys06] MySQL AB. *MySQL 5.0 Reference Manual: 3 Tutorial.* URL:http://dev.mysql.com/doc/refman/5.0/en/tutorial.html . 2006.

[Pet04] E. Petkevičius. *GPS technologija ir panaudojimas (1 dalis).* URL: http://www.elektronika.lt/theory/theme/160/118/. 1,1 MB, 2004.09.10.

[Php06] The PHP Group *PHP Manual: Function Reference.* URL:http://www.php.net/manual/en/funcref.php. 2006.05.08

[Pos06] PostNuke. *What is PostNuke.* URL:http://www.postnuke.com . 2006.

[San00] M. Sanders. *Oracle Custom Development Method Fast Track Method*. California, 2000.

[Ull03] L. Ullman. *Php and Mysql for Dynamic Web Sites: Visual Quickpro Guide.* Peachpit Press, Berkeley, 2003.

[Vee02] E. van Veenendall. *The Testing Parcitioner*. UTN Publishers, Den Bosch, 2002.

A PRIEDAI

A.1 VITMOS IS FUNKCIJŲ MOSCOW SĄRAŠAS

Lentelė 3 VITMOS IS funkcijų MoSCoW sąrašas

Funkcijos kodas	Funkcijos pavadinimas	Prioritetas
TIP_1	**Transporto informacijos portalo funkcijos**	
TIP_1_1	**Tvarkyti portalo duomenis**	
TIP_1_1_1	Registruoti šalį	P
TIP_1_1_2	Redaguoti šalį	P
TIP_1_1_3	Šalinti šalį	P
TIP_1_1_4	Registruoti miestą	P
TIP_1_1_5	Redaguoti miestą	P
TIP_1_1_6	Šalinti miestą	P
TIP_1_1_7	Registruoti krovinį	P
TIP_1_1_8	Redaguoti krovinį	P
TIP_1_1_9	Šalinti krovinį	P
TIP_1_1_10	Registruoti krovinių priėmėją	P
TIP_1_1_11	Redaguoti krovinių priėmėją	P
TIP_1_1_12	Šalinti krovinių priėmėją	P
TIP_1_1_13	Registruoti vairuotoją	T
TIP_1_1_14	Redaguoti vairuotoją	T
TIP_1_1_15	Šalinti vairuotoją	T
TIP_1_1_16	Registruoti klientą	P
TIP_1_1_17	Redaguoti klientų duomenis	P
TIP_1_1_18	Šalinti klientų duomenis	P
TIP_1_1_19	Registruoti maršrutą	P

TIP_1_1_20	Redaguoti maršruto duomenis	P
TIP_1_1_21	Šalinti maršrutą	P
TIP_1_1_22	Registruoti kliento objekto sekimą	P
TIP_1_1_23	Redaguoti kliento objekto sekimo duomenis	P
TIP_1_1_24	Šalinti kliento objekto sekimo duomenis	P
TIP_1_2	**Peržiūrėti portalo duomenis**	
TIP_1_2_1	Peržiūrėti šalių sąrašą	T
TIP_1_2_2	Peržiūrėti miestų sąrašą	T
TIP_1_2_3	Peržiūrėti krovinių sąrašą	P
TIP_1_2_4	Peržiūrėti krovinių priėmėjų sąrašą	P
TIP_1_2_5	Peržiūrėti vairuotojų sąrašą	T
TIP_1_2_6	Peržiūrėti maršrutų sąrašą	P
TIP_1_2_7	Peržiūrėti sekamo objekto buvimo vietą	P
TIP_1_2_8	Peržiūrėti sekamų objektų sąrašą	P
TVIMP_2	**Transporto viešosios informacijos mobilaus portalo funkcijos**	
TVIMP_2_1	**Tvarkyti portalo duomenis**	
TVIMP_2_1_1	**Tvarkyti portalo sisteminius duomenis**	
TVIMP_2_1_1_1	Registruoti meniu punktą	P
TVIMP_2_1_1_2	Redaguoti meniu punktą	P
TVIMP_2_1_1_3	Šalinti meniu punktą	P
TVIMP_2_1_1_4	Registruoti modulį	P
TVIMP_2_1_1_5	Redaguoti modulį	P
TVIMP_2_1_1_6	Šalinti modulį	P
TVIMP_2_1_1_7	Registruoti puslapį	P
TVIMP_2_1_1_8	Redaguoti puslapį	P
TVIMP_2_1_1_9	Šalinti puslapį	P
TVIMP_2_1_1_10	Registruoti puslapio šabloną	P
TVIMP_2_1_1_11	Redaguoti puslapio šabloną	P

TVIMP_2_1_1_12	Šalinti puslapio šabloną	P
TVIMP_2_1_1_13	Registruoti portalo vartotoją	P
TVIMP_2_1_1_14	Redaguoti portalo vartotojo duomenis	P
TVIMP_2_1_1_15	Šalinti portalo vartotoją	P
TVIMP_2_1_1_16	Sukurti vartotojo teises į modulį	P
TVIMP_2_1_1_17	Redaguoti vartotojo teises į modulį	P
TVIMP_2_1_1_18	Šalinti vartotojo teises į modulį	P
TVIMP_2_1_1_19	Registruoti portalo vartotojų grupę	P
TVIMP_2_1_1_20	Redaguoti portalo vartotojų grupės duomenis	P
TVIMP_2_1_1_21	Šalinti portalo vartotojų grupę	P
TVIMP_2_1_1_22	Sukurti vartotojų grupės teises į modulį	T
TVIMP_2_1_1_23	Redaguoti vartotojų grupės teises į modulį	T
TVIMP_2_1_1_24	Šalinti vartotojų grupės teises į modulį	T
TVIMP_2_1_1_25	Registruoti TVS vartotoją	P
TVIMP_2_1_1_26	Redaguoti TVS vartotojo duomenis	P
TVIMP_2_1_1_27	Šalinti TVS vartotoją	P
TVIMP_2_1_1_28	Keisti TVS vartotojo slaptažodį	P
TVIMP_2_1_1_29	Tvarkyti dokumentų atnaujinimo modulio nustatymus	T
TVIMP_2_1_1_30	Tvarkyti nuorodų atnaujinimo modulio nustatymus	T
TVIMP_2_1_1_31	Automatiškai atnaujinti portalo statinę informaciją - dokumentus	P
TVIMP_2_1_1_32	Automatiškai atnaujinti portalo statinę informaciją - nuorodas	P
TVIMP_2_1_1_33	Automatiškai surinkti ir pateikti informaciją iš kitų portalų apie kelius	N
TVIMP_2_1_2	**Tvarkyti portalo duomenis**	P
TVIMP_2_1_2_1	Registruoti naujieną	P
TVIMP_2_1_2_2	Redaguoti naujieną	P
TVIMP_2_1_2_3	Šalinti naujieną	P
TVIMP_2_1_2_4	Registruoti skelbimą	P
TVIMP_2_1_2_5	Redaguoti skelbimą	P

TVIMP_2_1_2_6	Šalinti skelbimą	P
TVIMP_2_2	**Peržiūrėti portalo duomenis**	
TVIMP_2_2_1	**Peržiūrėti portalo sisteminius duomenis**	
TVIMP_2_2_1_1	Peržiūrėti meniu punktų sąrašą	P
TVIMP_2_2_1_2	Peržiūrėti meniu punkto duomenis	G
TVIMP_2_2_1_3	Peržiūrėti modulių sąrašą	P
TVIMP_2_2_1_4	Peržiūrėti modulio duomenis	G
TVIMP_2_2_1_5	Peržiūrėti puslapių sąrašą	P
TVIMP_2_2_1_6	Peržiūrėti puslapio duomenis	G
TVIMP_2_2_1_7	Peržiūrėti portalo vartotojų sąrašą	P
TVIMP_2_2_1_8	Peržiūrėti portalo vartotojo duomenis	G
TVIMP_2_2_1_9	Peržiūrėti vartotojų teisių į modulius sąrašą	P
TVIMP_2_2_1_10	Peržiūrėti vartotojo teisės į modulį duomenis	G
TVIMP_2_2_1_11	Peržiūrėti portalo vartotojų grupių sąrašą	P
TVIMP_2_2_1_12	Peržiūrėti portalo vartotojų grupės duomenis	G
TVIMP_2_2_1_13	Peržiūrėti vartotojų grupių teisių į modulius sąrašą	P
TVIMP_2_2_1_14	Peržiūrėti vartotojų grupės teisės į modulį duomenis	G
TVIMP_2_2_1_15	Peržiūrėti TVS vartotojų sąrašą	P
TVIMP_2_2_1_16	Peržiūrėti TVS vartotojo duomenis	G
TVIMP_2_2_2	**Peržiūrėti portalo duomenis**	
TVIMP_2_2_2_1	Peržiūrėti portalo statinę informaciją - dokumentus	P
TVIMP_2_2_2_2	Peržiūrėti portalo statinę informaciją - nuorodas	P
TVIMP_2_2_2_3	Peržiūrėti naujienų sąrašą	P
TVIMP_2_2_2_4	Peržiūrėti naujieną	P
TVIMP_2_2_2_5	Peržiūrėti skelbimų sąrašą	P
TVIMP_2_2_2_6	Peržiūrėti skelbimą	P
TVIMP_2_2_2_7	Peržiūrėti viešosios statistikos informaciją apie transporto srautus, bendrą kelių naudojimą	P
TVIMP_2_2_2_8	Peržiūrėti portalo lankomumo statistiką	G

TVIMP_2_2_2_9	Peržiūrėti informaciją, automatiškai surinktą iš kitų portalų, apie kelius	N

A.2 TRANSPORTO VIEŠOSIOS INFORMACIJOS MOBILAUS PORTALO TESTAVIMO SCENARIJAI

Lentelė 4 - Transporto viešosios informacijos mobilaus portalo testavimo scenarijai

Proceso NR:	**TVIMP1**
Proceso pavadinimas:	**TVIMP1. Administruoti transporto viešosios informacijos portalo vartotojus ir jų grupes**
Scenarijaus NR:	TVIMP1_1
Scenarijaus pavadinimas:	Registruoti portalo vartotojų grupę
Scenarijaus NR:	TVIMP1_2
Scenarijaus pavadinimas:	Peržiūrėti portalo vartotojų grupių sąrašą
Scenarijaus NR:	TVIMP1_3
Scenarijaus pavadinimas:	Redaguoti portalo vartotojų grupės duomenis
Scenarijaus NR:	TVIMP1_4
Scenarijaus pavadinimas:	Registruoti portalo vartotoją
Scenarijaus NR:	TVIMP1_5
Scenarijaus pavadinimas:	Peržiūrėti portalo vartotojų sąrašą
Scenarijaus NR:	TVIMP1_6
Scenarijaus pavadinimas:	Redaguoti portalo vartotojo duomenis
Scenarijaus NR:	TVIMP1_7
Scenarijaus pavadinimas:	Registruoti TVS vartotoją
Proceso NR:	**TVIMP2**
Proceso pavadinimas:	**TVIMP2.Administruoti portalo modulius, meniu punktus, puslapius**
Scenarijaus NR:	TVIMP2_1
Scenarijaus pavadinimas:	Registruoti meniu punktą
Scenarijaus NR:	TVIMP2_2
Scenarijaus pavadinimas:	Registruoti modulį
Scenarijaus NR:	TVIMP2_3

Scenarijaus pavadinimas:	Registruoti puslapio šabloną
Scenarijaus NR:	TVIMP2_4
Scenarijaus pavadinimas:	Registruoti puslapį
Proceso NR:	**TVIMP3**
Proceso pavadinimas:	**TVIMP3.Administruoti vartotojų grupių ir vartotojų teises į modulius**
Scenarijaus NR:	TVIMP3_1
Scenarijaus pavadinimas:	Sukurti vartotojų grupės teises į modulį
Scenarijaus NR:	TVIMP3_2
Scenarijaus pavadinimas:	Peržiūrėti vartotojų grupių teisių į modulius sąrašą
Scenarijaus NR:	TVIMP3_3
Scenarijaus pavadinimas:	Redaguoti vartotojų grupės teises į modulį
Scenarijaus NR:	TVIMP3_4
Scenarijaus pavadinimas:	Sukurti vartotojo teises į modulį
Scenarijaus NR:	TVIMP3_5
Scenarijaus pavadinimas:	Peržiūrėti vartotojų teisių į modulius sąrašą
Scenarijaus NR:	TVIMP3_6
Scenarijaus pavadinimas:	Redaguoti vartotojo teises į modulį
Proceso NR:	**TVIMP4**
Proceso pavadinimas:	**TVIMP4.Administruoti portalo statinę informaciją**
Scenarijaus NR:	TVIMP4_1
Scenarijaus pavadinimas:	Tvarkyti dokumentų atnaujinimo modulio nustatymus
Scenarijaus NR:	TVIMP4_2
Scenarijaus pavadinimas:	Tvarkyti nuorodų atnaujinimo modulio nustatymus
Scenarijaus NR:	TVIMP4_3
Scenarijaus pavadinimas:	Automatiškai atnaujinti portalo statinę informaciją – dokumentus
Scenarijaus NR:	TVIMP4_4
Scenarijaus pavadinimas:	Automatiškai atnaujinti portalo statinę informaciją – nuorodas

Proceso NR:	**TVIMP5**
Proceso pavadinimas:	**TVIMP5.Administruoti portalo dinaminę informaciją**
Scenarijaus NR:	TVIMP5_1
Scenarijaus pavadinimas:	**Registruoti naujieną**
Scenarijaus NR:	TVIMP5_2
Scenarijaus pavadinimas:	**Peržiūrėti naujienų sąrašą**
Scenarijaus NR:	TVIMP5_3
Scenarijaus pavadinimas:	**Redaguoti naujieną**
Scenarijaus NR:	TVIMP5_4
Scenarijaus pavadinimas:	**Registruoti skelbimą**
Scenarijaus NR:	TVIMP5_5
Scenarijaus pavadinimas:	**Peržiūrėti skelbimų sąrašą**
Scenarijaus NR:	TVIMP5_6
`Scenarijaus pavadinimas:	**Redaguoti skelbimą**

A.3 TRANSPORTO INFORMACIJOS PORTALO TESTAVIMO SCENARIJAI

Lentelė 5 - Transporto informacijos portalo testavimo scenarijai

Proceso NR:	**TIP1**
Proceso pavadinimas:	**TIP1.Administruoti vykdomus pervežimus.**
Scenarijaus NR:	TIP1_1
Scenarijaus pavadinimas:	Registruoti šalį
Scenarijaus NR:	TIP1_2
Scenarijaus pavadinimas:	Registruoti miestą
Scenarijaus NR:	TIP1_3
Scenarijaus pavadinimas:	Registruoti klientą
Scenarijaus NR:	TIP1_4
Scenarijaus pavadinimas:	Registruoti krovinį

Scenarijaus NR:	TIP1_5
Scenarijaus pavadinimas:	Registruoti krovinių priėmėją
Scenarijaus NR:	TIP1_6
Scenarijaus pavadinimas:	Registruoti vairuotoją
Scenarijaus NR:	TIP1_7
Scenarijaus pavadinimas:	Registruoti maršrutą
Proceso NR:	**TIP2**
Proceso pavadinimas:	**TIP2.Administruoti pervežamų objektų sekimą**
Scenarijaus NR:	TIP2_1
Scenarijaus pavadinimas:	Registruoti kliento objekto sekimą
Scenarijaus NR:	TIP2_2
Scenarijaus pavadinimas:	Peržiūrėti sekamų objektų sąrašą
Scenarijaus NR:	TIP2_3
Scenarijaus pavadinimas:	Redaguoti kliento objekto sekimo duomenis
Scenarijaus NR:	TIP2_4
Scenarijaus pavadinimas:	Peržiūrėti sekamo objekto buvimo vietą